



User Guide for Version 2

AWS Command Line Interface



AWS Command Line Interface: User Guide for Version 2

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

.....	ix
About the AWS CLI	1
About AWS CLI version 2	1
Maintenance and support for SDK major versions	2
About Amazon Web Services	2
About the examples	2
Additional documentation and resources	4
AWS CLI documentation and resources	4
Other AWS SDKs	4
Get started	5
Prerequisites	6
Create an IAM or IAM Identity Center administrative account	6
Next steps	7
Install/Update	7
AWS CLI install and update instructions	8
Troubleshooting AWS CLI install and uninstall errors	21
Next steps	22
Past releases	22
Troubleshooting AWS CLI install and uninstall errors	40
Next steps	40
Build and install from source	40
Why build from source?	41
Quicksteps	41
Step 1: Setup all requirements	44
Step 2: Configuring the AWS CLI source installation	48
Step 3: Building the AWS CLI	54
Step 4: Installing the AWS CLI	55
Step 5: Verifying the AWS CLI installation	57
Workflow examples	57
Troubleshooting AWS CLI install and uninstall errors	60
Next steps	60
Amazon ECR Public/Docker	61
Prerequisites	61
Deciding between Amazon ECR Public and Docker Hub	61

Run the official images	62
Notes on interfaces and backwards compatibility of the official images	63
Use specific versions and tags	63
Update to the latest official image	64
Share host files, credentials, environment variables, and configuration	65
Shorten the docker run command	71
Setup	74
Gather your credential information for programmatic access	74
Setting up new configuration and credentials	75
Using existing configuration and credentials files	84
Configure the AWS CLI	85
Configuration and credentials precedence	85
Additional topics in this section	86
Configuration and credential file settings	87
Format of the configuration and credential files	87
Where are configuration settings stored?	95
Using named profiles	96
Set and view configuration settings using commands	97
Setting new configuration and credentials command examples	100
Supported config file settings	103
Environment Variables	121
How to set environment variables	122
AWS CLI supported environment variables	123
Command line options	133
How to use command line options	133
AWS CLI supported global command line options	134
Common uses of command line options	139
Command completion	139
How it works	140
Configuring command completion on Linux or macOS	140
Configuring command completion on Windows	144
Retries	145
Available retry modes	146
Configuring a retry mode	148
Viewing logs of retry attempts	149
Use an HTTP proxy	150

Using the examples	151
Authenticating to a proxy	152
Using a proxy on Amazon EC2 instances	152
Troubleshooting	153
Endpoints	153
Set endpoint for a single command	154
Set global endpoint for all AWS services	154
Set to use FIPs endpoints for all AWS services	155
Set to use dual-stack endpoints for all AWS services	156
Set service-specific endpoints	157
Endpoint configuration and settings precedence	200
Authentication and access credentials	201
Configuration and credential precedence	202
Additional topics in this section	203
IAM Identity Center authentication	203
Configure automatic token refresh	204
Configure legacy non-refreshable	211
Using an IAM Identity Center profile	217
Short-term credentials	220
IAM roles	221
Prerequisites	221
Overview of using IAM roles	221
Configuring and using a role	223
Using MFA	225
Cross-account roles and external ID	226
Specifying a role session name for easier auditing	227
Assume role with web identity	227
Clearing cached credentials	229
IAM users	229
Step 1: Create your IAM user	230
Step 2: Get your access keys	230
Configure the AWS CLI	231
Use credentials for Amazon EC2 instance metadata	233
Prerequisites	233
Configuring a profile for Amazon EC2 metadata	234
External credentials	235

Use the AWS CLI	238
Get Help	238
The built-in AWS CLI help command	239
AWS CLI reference guide	244
API documentation	244
Troubleshooting errors	245
Additional help	245
Command Structure	245
Command structure	245
Wait commands	246
Specify Parameter Values	248
Common Parameter Types	249
Quotes with Strings	254
Parameters from Files	258
Generate a CLI Skeleton Template	261
Shorthand Syntax	273
Auto-prompt	275
How it works	276
Auto-prompt features	276
Auto-prompt modes	279
Configure auto-prompt	280
Control Command Output	280
Sensitive output	280
Server-side vs client-side output options	281
Output Format	282
Pagination	291
Filter output	297
Return Codes	321
Wizards	322
How it works	323
Aliases	324
Prerequisites	324
Step 1: Creating the alias file	324
Step 2: Creating an alias	326
Step 3: Calling an alias	329
Alias repository examples	331

Resources	332
Code examples	333
Guided command examples	333
DynamoDB	334
Amazon EC2	338
S3 Glacier	356
IAM	363
Amazon S3	368
Amazon SNS	387
Command examples	389
Actions and scenarios	390
Bash script examples	6417
Actions and scenarios	6418
Security	6688
Data Protection	6688
Data encryption	6689
Identity and Access Management	6690
Audience	6690
Authenticating with identities	6691
Managing access using policies	6694
How AWS services work with IAM	6697
Troubleshooting AWS identity and access	6697
Compliance Validation	6699
Resilience	6700
Infrastructure Security	6700
Enforcing a minimum TLS version	6701
Troubleshoot errors	6702
General troubleshooting to try first	6702
Check your AWS CLI command formatting	6703
Check the AWS Region your AWS CLI command is using	6703
Confirm that you're running a recent version of the AWS CLI	6704
Use the --debug option	6704
Enable and review the AWS CLI command history logs	6710
Confirm that your AWS CLI is configured	6710
Command not found errors	6711
The "aws --version" command returns a different version than you installed	6714

The "aws --version" command returns a version after uninstalling the AWS CLI	6715
The AWS CLI processed a command with an incomplete parameter name	6716
Access denied errors	6717
Invalid credentials and key errors	6718
Signature does not match errors	6720
SSL certificate errors	6721
Invalid JSON errors	6722
Additional resources	6724
Migration guide	6725
New features and changes	6725
AWS CLI version 2 new features	6725
Breaking changes between AWS CLI version 1 and AWS CLI version 2	6727
Migration instructions	6734
Replacing version 1 with version 2	6735
Side-by-side install	6735
Uninstall	6737
Troubleshooting AWS CLI install and uninstall errors	6740
Document History	6741
AWS Glossary	6746

What is the AWS Command Line Interface?

The AWS Command Line Interface (AWS CLI) is an open source tool that enables you to interact with AWS services using commands in your command-line shell. With minimal configuration, the AWS CLI enables you to start running commands that implement functionality equivalent to that provided by the browser-based AWS Management Console from the command prompt in your terminal program:

- **Linux shells** – Use common shell programs such as [bash](#), [zsh](#), and [tcsch](#) to run commands in Linux or macOS.
- **Windows command line** – On Windows, run commands at the Windows command prompt or in PowerShell.
- **Remotely** – Run commands on Amazon Elastic Compute Cloud (Amazon EC2) instances through a remote terminal program such as PuTTY or SSH, or with AWS Systems Manager.

All IaaS (infrastructure as a service) AWS administration, management, and access functions in the AWS Management Console are available in the AWS API and AWS CLI. New AWS IaaS features and services provide full AWS Management Console functionality through the API and CLI at launch or within 180 days of launch.

The AWS CLI provides direct access to the public APIs of AWS services. You can explore a service's capabilities with the AWS CLI, and develop shell scripts to manage your resources. In addition to the low-level, API-equivalent commands, several AWS services provide customizations for the AWS CLI. Customizations can include higher-level commands that simplify using a service with a complex API.

About AWS CLI version 2

The AWS CLI version 2 is the most recent major version of the AWS CLI and supports all of the latest features. Some features introduced in version 2 are not backported to version 1 and you must upgrade to access those features. There are some "breaking" changes from version 1 that might require you to change your scripts. For a list of breaking changes in version 2, see [Migrate from AWS CLI version 1 to version 2](#).

The AWS CLI version 2 is available to install only as a bundled installer. While you might find it in package managers, these are unsupported and unofficial packages that are not produced

or managed by AWS. We recommend that you install the AWS CLI from only the official AWS distribution points, as documented in this guide.

To install the AWS CLI version 2, see [the section called “Install/Update”](#).

To check the currently installed version, use the following command:

```
$ aws --version
aws-cli/2.15.30 Python/3.11.6 Linux/5.10.205-195.807.amzn2.x86_64 botocore/1.18.6
```

For version history, see the [AWS CLI version 2 Changelog](#) on *GitHub*.

Maintenance and support for SDK major versions

For information about maintenance and support for SDK major versions and their underlying dependencies, see the following in the [AWS SDKs and Tools Reference Guide](#):

- [AWS SDKs and tools maintenance policy](#)
- [AWS SDKs and tools version support matrix](#)

About Amazon Web Services

Amazon Web Services (AWS) is a collection of digital infrastructure services that developers can leverage when developing their applications. The services include computing, storage, database, and application synchronization (messaging and queuing). AWS uses a pay-as-you-go service model. You are charged only for the services that you—or your applications—use. Also, to make AWS more approachable as a platform for prototyping and experimentation, AWS offers a free usage tier. On this tier, services are free below a certain level of usage. For more information about AWS costs and the Free Tier, see [AWS Free Tier](#). To obtain an AWS account, open the [AWS home page](#) and then choose **Create an AWS Account**.

About the AWS CLI examples

The AWS Command Line Interface (AWS CLI) examples in this guide are formatted using the following conventions:

- **Prompt** – The command prompt uses the Linux prompt and is displayed as (\$). For commands that are Windows specific, C:\> is used as the prompt. Do not include the prompt when you type commands.
- **Directory** – When commands must be executed from a specific directory, the directory name is shown before the prompt symbol.
- **User input** – Command text that you enter at the command line is formatted as **user input**.
- **Replaceable text** – Variable text, including names of resources that you choose, or IDs generated by AWS services that you must include in commands, is formatted as *replaceable text*. In multiple-line commands or commands where specific keyboard input is required, keyboard commands can also be shown as replaceable text.
- **Output** – Output returned by AWS services is shown under user input, and is formatted as `computer output`.

The following **aws configure** command example demonstrates user input, replaceable text, and output:

1. Enter **aws configure** at the command line, and then press **Enter**.
2. The AWS CLI outputs lines of text, prompting you to enter additional information.
3. Enter each of your access keys in turn, and then press **Enter**.
4. Then, enter an AWS Region name in the format shown, press **Enter**, and then press **Enter** a final time to skip the output format setting.
5. The final **Enter** command is shown as replaceable text because there is no user input for that line.

```
$ aws configure
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
Default region name [None]: us-west-2
Default output format [None]: ENTER
```

The following example shows a simple command with output. To use this example, enter the full text of the command (the highlighted text after the prompt), and then press **Enter**. The name of the security group, *my-sg*, is replaceable to your desired security group name. The JSON document, including the curly braces, is output. If you configure your CLI to output in text or table format, the output will be formatted differently. [JSON](#) is the default output format.

```
$ aws ec2 create-security-group --group-name my-sg --description "My security group"
{
  "GroupId": "sg-903004f8"
}
```

Additional documentation and resources

AWS CLI documentation and resources

In addition to this user guide, the following are valuable online resources for the AWS CLI.

- [AWS CLI version 2 reference guide](#)
- [AWS CLI code examples repository](#)
- [AWS CLI GitHub repository](#) You can view and fork the source code for the AWS CLI on GitHub. Join the community of users on GitHub to provide feedback, request features, and submit your own contributions.
- [AWS CLI alias examples repository](#) You can view and fork AWS CLI alias examples on GitHub.
- [AWS CLI version 2 Changelog](#)

Other AWS SDKs

Depending on your use case, you might want to choose one of the AWS SDKs or the AWS Tools for PowerShell:

- [AWS Tools for PowerShell](#)
- [AWS SDK for Java](#)
- [AWS SDK for .NET](#)
- [AWS SDK for JavaScript](#)
- [AWS SDK for Ruby](#)
- [AWS SDK for Python \(Boto\)](#)
- [AWS SDK for PHP](#)
- [AWS SDK for Go](#)
- [AWS Mobile SDK for iOS](#)
- [AWS Mobile SDK for Android](#)

Get started with the AWS CLI

This chapter provides steps to get started with version 2 of the AWS Command Line Interface (AWS CLI) and provides links to the relevant instructions.

1. [Complete all prerequisites](#) - To access AWS services with the AWS CLI, you need at minimum an AWS account and IAM credentials. To increase the security of your AWS account, we recommend that you do not use your root account credentials. You should create a user with least privilege to provide access credentials to the tasks you'll be running in AWS.
2. Install or gain access to the AWS CLI using one of the following methods:
 - **(Recommended)** [the section called "Install/Update"](#).
 - [the section called "Past releases"](#). Installing a specific version is primarily used if your team aligns their tools to a specific version.
 - [the section called "Build and install from source"](#). Building the AWS CLI from GitHub source is a more in-depth method that is primarily used by customers who work on platforms that we do not directly support with our pre-built installers.
 - [the section called "Amazon ECR Public/Docker"](#).
 - Access the AWS CLI version 2 in the AWS console from your browser using AWS CloudShell. For more information, see the [AWS CloudShell User Guide](#).
3. [After you have access to the AWS CLI, configure your AWS CLI with your IAM credentials for first time use.](#)

Troubleshooting installer or configure errors

If you have issues after installing, uninstalling, or configuring the AWS CLI, see [Troubleshoot errors](#) for troubleshooting steps.

Topics

- [Prerequisites to use the AWS CLI version 2](#)
- [Install or update to the latest version of the AWS CLI](#)
- [Install past releases of the AWS CLI version 2](#)
- [Build and install the AWS CLI from source](#)
- [Run the AWS CLI from the official Amazon ECR Public or Docker images](#)

- [Set up the AWS CLI](#)

Prerequisites to use the AWS CLI version 2

To access AWS services with the AWS CLI, you need an AWS account and IAM credentials. When running AWS CLI commands, the AWS CLI needs to have access to those AWS credentials. To increase the security of your AWS account, we recommend that you do not use your root account credentials. You should create a user with least privilege to provide access credentials to the tasks you'll be running in AWS.

Topics

- [Create an IAM or IAM Identity Center administrative account](#)
- [Next steps](#)

Create an IAM or IAM Identity Center administrative account

Before you can configure

To create an administrator user, choose one of the following options.

Choose one way to manage your administrator	To	By	You can also
In IAM Identity Center (Recommended)	Use short-term credentials to access AWS. This aligns with the security best practices . For information about best practices , see Security best	Following the instructions in Getting started in the <i>AWS IAM Identity Center User Guide</i> .	Configure programmatic access by Configuring the AWS CLI to use AWS IAM Identity Center in the <i>AWS Command Line Interface User Guide</i> .

Choose one way to manage your administrator	To	By	You can also
	practices in IAM in the <i>IAM User Guide</i> .		
In IAM (Not recommended)	Use long-term credentials to access AWS.	Following the instructions in Creating your first IAM admin user and user group in the <i>IAM User Guide</i> .	Configure programmatic access by Managing access keys for IAM users in the <i>IAM User Guide</i> .

Next steps

After creating an AWS account and IAM credentials, to use the AWS CLI you can do one of the following:

- [Install the latest release](#) of the AWS CLI version 2 on your computer.
- [Install a past release](#) of the AWS CLI version 2 on your computer.
- Access the AWS CLI version 2 from your computer [using a Docker image](#).
- Access the AWS CLI version 2 in the AWS console from your browser using AWS CloudShell. For more information see the [AWS CloudShell User Guide](#).

Install or update to the latest version of the AWS CLI

This topic describes how to install or update the latest release of the AWS Command Line Interface (AWS CLI) on supported operating systems. For information on the latest releases of AWS CLI, see the [AWS CLI version 2 Changelog](#) on GitHub.

To install a past release of the AWS CLI, see [the section called "Past releases"](#). For uninstall instructions, see [Uninstall](#).

Important

AWS CLI versions 1 and 2 use the same `aws` command name. If you previously installed AWS CLI version 1, see [Migrate from AWS CLI version 1 to version 2](#).

Topics

- [AWS CLI install and update instructions](#)
- [Troubleshooting AWS CLI install and uninstall errors](#)
- [Next steps](#)

AWS CLI install and update instructions

For installation instructions, expand the section for your operating system.

Linux

Install and update requirements

- You must be able to extract or "unzip" the downloaded package. If your operating system doesn't have the built-in `unzip` command, use an equivalent.
- The AWS CLI uses `glibc`, `groff`, and `less`. These are included by default in most major distributions of Linux.
- We support the AWS CLI on 64-bit versions of recent distributions of CentOS, Fedora, Ubuntu, Amazon Linux 1, Amazon Linux 2, Amazon Linux 2023, and Linux ARM.
- Because AWS doesn't maintain third-party repositories, we can't guarantee that they contain the latest version of the AWS CLI.

Install or update the AWS CLI

Warning

If this is your first time updating on Amazon Linux, to install the latest version of the AWS CLI, you must uninstall the pre-installed `yum` version using the following command:


```
$ sudo yum remove awscli
```

After the yum installation of the AWS CLI is removed, follow the below Linux install instructions.

To update your current installation of AWS CLI, download a new installer each time you update to overwrite previous versions. Follow these steps from the command line to install the AWS CLI on Linux.

The following are quick installation steps in a single copy and paste group based on whether you use 64-bit Linux or Linux ARM that provide a basic installation. For guided instructions, see the steps that follow.

Linux x86 (64-bit)

 **Note**

(Optional) The following command block downloads and installs the AWS CLI without first verifying the integrity of your download. To verify the integrity of your download, use the below step by step instructions.

To install the AWS CLI, run the following commands.

```
$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
```

To update your current installation of the AWS CLI, add your existing symlink and installer information to construct the `install` command using the `--bin-dir`, `--install-dir`, and `--update` parameters. The following command block uses an example symlink of `/usr/local/bin` and example installer location of `/usr/local/aws-cli`.

```
$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install --bin-dir /usr/local/bin --install-dir /usr/local/aws-cli --update
```


Linux ARM

Note

(Optional) The following command block downloads and installs the AWS CLI without first verifying the integrity of your download. To verify the integrity of your download, use the below step by step instructions.

To install the AWS CLI, run the following commands.

```
$ curl "https://awscli.amazonaws.com/awscli-exe-linux-aarch64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
```

To update your current installation of the AWS CLI, add your existing symlink and installer information to construct the `install` command using the `--bin-dir`, `--install-dir`, and `--update` parameters. The following command block uses an example symlink of `/usr/local/bin` and example installer location of `/usr/local/aws-cli`.

```
$ curl "https://awscli.amazonaws.com/awscli-exe-linux-aarch64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install --bin-dir /usr/local/bin --install-dir /usr/local/aws-cli --
update
```

Guided installation steps

1. Download the installation file in one of the following ways:

Linux x86 (64-bit)

- **Use the `curl` command** – The `-o` option specifies the file name that the downloaded package is written to. The options on the following example command write the downloaded file to the current directory with the local name `awscliv2.zip`.

```
$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
"awscliv2.zip"
```

- **Downloading from the URL** – To download the installer with your browser, use the following URL: https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip

Linux ARM

- **Use the curl command** – The `-o` option specifies the file name that the downloaded package is written to. The options on the following example command write the downloaded file to the current directory with the local name `awscliv2.zip`.

```
$ curl "https://awscli.amazonaws.com/awscli-exe-linux-aarch64.zip" -o
"awscliv2.zip"
```

- **Downloading from the URL** – To download the installer with your browser, use the following URL: <https://awscli.amazonaws.com/awscli-exe-linux-aarch64.zip>

2. (Optional) Verifying the integrity of your downloaded zip file

If you chose to manually download the AWS CLI installer package `.zip` in the above steps, you can use the following steps to verify the signatures by using the GnuPG tool.

The AWS CLI installer package `.zip` files are cryptographically signed using PGP signatures. If there is any damage or alteration of the files, this verification fails and you should not proceed with installation.

- Download and install the `gpg` command using your package manager. For more information about GnuPG, see the [GnuPG website](#).
- To create the public key file, create a text file and paste in the following text.

```
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBF2Cr7UBEADJZHcgus0Jl7ENSyumXh85z0TRV0xJorM2B/JL0kH0yigQ1uUG
ZMLhENaG0bYatdrKP+3H911vK050pXwn0/R7fB/FSTouki4ciIx50uLlnJZIxSzx
PqG10mkxImLNBGwoi6Lto0LYxqHN2iQtzlwTVmq9733zd3XfcXrZ3+Lb1HAgEt5G
TfnxEKJ8soPLyWmwDH6HWCnjZ/aIQRBTIQ05uVeEoYxSh6w0ai7ss/KveoSNBbYz
gbdzoqI2Y8cgH2bnfgp3DSasaLZEdCSsIsK1u05CinE7k2qZ7KgKAUIcT/cR/grk
C6VwsnDU00UCideXcQ8WeHutqvgZH1JgKDbznoIzeQHJD238GEu+eKhRHcz8/jeG
94zkcqJ0z3KbZGYMiTh277Fvj9zzvZsbMBCedV1BTg3Tqgvdx4bdkhf5cH+7NtW0
lrFj6UwAsGukBTA0xC01/dnSmZhJ7Z1KmEWilro/g0rjt0xqRQut1IqG22TaqoPG
fYVN+en3ZwbT97kcgZDwqbuykNt64oZwC4XKCa3mprEGC3IbJTBFqg1XmZ719ywg
EEUJY01b2XrSuPwm139beWdKM8kzr10jn10m6+lpTRCBfo0wa9F8YZRhHPAkWkKX
XDe0GpWrj4oh0x0d2GWkyV5xyN14p2tQ0Cd00Dmz80yUTgRpPVQut0EHXQARAQAB
```

```
tCFBV1MgQ0xJIFR1YW0gPGF3cy1jbG1AYW1hem9uLmNvbT6JA1QEEwEIAD4CGwMF
CwkIBwIGFQoJCAAsCBBYCAwECHgECF4AWIQT7Xbd/1cEYuAURraimMQrMRnJHXAUC
ZMKcEgUJCSEf3QAKCRCmMQrMRnJHXci1D/4vior9J5tB+icri5WbDudS3ak/ve4q
XS6ZLm5S81+CBxy5aLQUlyFhuaaEHDC11fG780duxatzeHENASYVo3mmKNwrCBza
NJaeaWKLGT0MKwBSP5aa3dva8P/4oUP9GsQn0uWoXwNDWfrMbNI8gn+jC/3MigW
vD3fu6zCOWWLITNv2SJoQ1wILmb/uGfha68o4iTB0vcftVRuao6DyqF+CrHX/0j0
k1EDQFMY9M4tsYT7X8NwfI8Vmc89nzpvL9fwda44WwpKIw1FBZP8S0sgDx2xDsxv
L8kM2Gt0iH0cHqF0+V7xtTKZylo1iDbJKhu80Kc+YC/TmozD8oeGU2rEFxfLegwS
zT9N+jB38+dqaP9pRDSi45iGqyA8yavVBabpL0IQ9jU6eIV+kmcjIjcun/Uo8SjJ
0xQAsm41rxPaKV6vJUn10wVNuhSkKk8mzN01SZwu7Hua6rdcCaGeB8uJ44AP3QzW
BNnrjtoN6A1N0D2wFmfE/YL/rHPxU1XwPntubYB/t3rXFL7ENQ00QH0KVXgRC1ey
sHMglg46c+nQLRzVTshjDjmtzv9rcV9RKR0PetEggzCoD89veDA9jPR2Kw6RYkS
XzYm2fEv16/HRNYt7hJzneFqRIjHW5qAgSs/bcaRWpAU/QQzzJPVKCQNr4y0weyq
B8HctGjfod0p1A==
=gdMc
-----END PGP PUBLIC KEY BLOCK-----
```

For reference, the following are the details of the public key.

```
Key ID:          A6310ACC4672475C
Type:           RSA
Size:          4096/4096
Created:       2019-09-18
Expires:      2024-07-26
User ID:       AWS CLI Team <aws-cli@amazon.com>
Key fingerprint: FB5D B77F D5C1 18B8 0511 ADA8 A631 0ACC 4672 475C
```

- c. Import the AWS CLI public key with the following command, substituting *public-key-file-name* with the file name of the public key you created.

```
$ gpg --import public-key-file-name
gpg: /home/username/.gnupg/trustdb.gpg: trustdb created
gpg: key A6310ACC4672475C: public key "AWS CLI Team <aws-cli@amazon.com>"
imported
gpg: Total number processed: 1
gpg:             imported: 1
```

- d. Download the AWS CLI signature file for the package you downloaded. It has the same path and name as the .zip file it corresponds to, but has the extension .sig. In the following examples, we save it to the current directory as a file named `awscliv2.sig`.

Linux x86 (64-bit)

For the latest version of the AWS CLI, use the following command block:

```
$ curl -o awscliv2.sig https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip.sig
```

For a specific version of the AWS CLI, append a hyphen and the version number to the filename. For this example the filename for version *2.0.30* would be `awscli-exe-linux-x86_64-2.0.30.zip.sig` resulting in the following command:

```
$ curl -o awscliv2.sig https://awscli.amazonaws.com/awscli-exe-linux-x86_64-2.0.30.zip.sig
```

For a list of versions, see the [AWS CLI version 2 Changelog](#) on *GitHub*.

Linux ARM

For the latest version of the AWS CLI, use the following command block:

```
$ curl -o awscliv2.sig https://awscli.amazonaws.com/awscli-exe-linux-aarch64.zip.sig
```

For a specific version of the AWS CLI, append a hyphen and the version number to the filename. For this example the filename for version *2.0.30* would be `awscli-exe-linux-aarch64-2.0.30.zip.sig` resulting in the following command:

```
$ curl -o awscliv2.sig https://awscli.amazonaws.com/awscli-exe-linux-aarch64-2.0.30.zip.sig
```

For a list of versions, see the [AWS CLI version 2 Changelog](#) on *GitHub*.

- e. Verify the signature, passing both the downloaded `.sig` and `.zip` file names as parameters to the `gpg` command.

```
$ gpg --verify awscliv2.sig awscliv2.zip
```

The output should look similar to the following.

```
gpg: Signature made Mon Nov  4 19:00:01 2019 PST
gpg:          using RSA key FB5D B77F D5C1 18B8 0511 ADA8 A631 0ACC 4672
475C
gpg: Good signature from "AWS CLI Team <aws-cli@amazon.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: FB5D B77F D5C1 18B8 0511 ADA8 A631 0ACC 4672 475C
```

Important

The warning in the output is expected and doesn't indicate a problem. It occurs because there isn't a chain of trust between your personal PGP key (if you have one) and the AWS CLI PGP key. For more information, see [Web of trust](#).

3. Unzip the installer. If your Linux distribution doesn't have a built-in unzip command, use an equivalent to unzip it. The following example command unzips the package and creates a directory named `aws` under the current directory.

```
$ unzip awscliv2.zip
```

Note

When updating from a previous version, the `unzip` command prompts to overwrite existing files. To skip these prompts, such as with script automation, use the `-u` update flag for `unzip`. This flag automatically updates existing files and creates new ones as needed.

```
$ unzip -u awscliv2.zip
```

4. Run the install program. The installation command uses a file named `install` in the newly unzipped `aws` directory. By default, the files are all installed to `/usr/local/aws-cli`, and a symbolic link is created in `/usr/local/bin`. The command includes `sudo` to grant write permissions to those directories.

```
$ sudo ./aws/install
```

You can install without `sudo` if you specify directories that you already have write permissions to. Use the following instructions for the `install` command to specify the installation location:

- Ensure that the paths you provide to the `-i` and `-b` parameters contain no volume name or directory names that contain any space characters or other white space characters. If there is a space, the installation fails.
- `--install-dir` or `-i` – This option specifies the directory to copy all of the files to.

The default value is `/usr/local/aws-cli`.

- `--bin-dir` or `-b` – This option specifies that the main `aws` program in the install directory is symbolically linked to the file `aws` in the specified path. You must have write permissions to the specified directory. Creating a symlink to a directory that is already in your path eliminates the need to add the install directory to the user's `$PATH` variable.

The default value is `/usr/local/bin`.

```
$ ./aws/install -i /usr/local/aws-cli -b /usr/local/bin
```

Note

To update your current installation of the AWS CLI, add your existing symlink and installer information to construct the `install` command with the `--update` parameter.

```
$ sudo ./aws/install --bin-dir /usr/local/bin --install-dir /usr/local/aws-cli --update
```

To locate the existing symlink and installation directory, use the following steps:

1. Use the `which` command to find your symlink. This gives you the path to use with the `--bin-dir` parameter.

```
$ which aws
/usr/local/bin/aws
```

2. Use the `ls` command to find the directory that your symlink points to. This gives you the path to use with the `--install-dir` parameter.

```
$ ls -l /usr/local/bin/aws
lrwxrwxrwx 1 ec2-user ec2-user 49 Oct 22 09:49 /usr/local/bin/aws -> /usr/
local/aws-cli/v2/current/bin/aws
```

5. Confirm the installation with the following command.

```
$ aws --version
aws-cli/2.15.30 Python/3.11.6 Linux/5.10.205-195.807.amzn2.x86_64 botocore/2.4.5
```

If the `aws` command cannot be found, you might need to restart your terminal or follow the troubleshooting in [Troubleshoot errors](#).

macOS

Install and update requirements

- We support the AWS CLI on macOS versions 10.9 and later. For more information, see [macOS support policy updates for the AWS CLI v2](#) on the *AWS Developer Tools Blog*.
- Because AWS doesn't maintain third-party repositories, we can't guarantee that they contain the latest version of the AWS CLI.

Install or update the AWS CLI

If you are updating to the latest version, use the same installation method that you used in your current version. You can install the AWS CLI on macOS in the following ways.

GUI installer

The following steps show how to install the latest version of the AWS CLI by using the standard macOS user interface and your browser.

1. In your browser, download the macOS pkg file: <https://awscli.amazonaws.com/AWSCLIV2.pkg>
2. Run your downloaded file and follow the on-screen instructions. You can choose to install the AWS CLI in the following ways:

- **For all users on the computer (requires sudo)**
 - You can install to any folder, or choose the recommended default folder of `/usr/local/aws-cli`.
 - The installer automatically creates a symlink at `/usr/local/bin/aws` that links to the main program in the installation folder you chose.
- **For only the current user (doesn't require sudo)**
 - You can install to any folder to which you have write permission.
 - Due to standard user permissions, after the installer finishes, you must manually create a symlink file in your `$PATH` that points to the `aws` and `aws_completer` programs by using the following commands at the command prompt. If your `$PATH` includes a folder you can write to, you can run the following command without `sudo` if you specify that folder as the target's path. If you don't have a writable folder in your `$PATH`, you must use `sudo` in the commands to get permissions to write to the specified target folder. The default location for a symlink is `/usr/local/bin/`.

```
$ sudo ln -s /folder/installed/aws-cli/aws /usr/local/bin/aws
$ sudo ln -s /folder/installed/aws-cli/aws_completer /usr/local/bin/
aws_completer
```

Note

You can view debug logs for the installation by pressing **Cmd+L** anywhere in the installer. This opens a log pane that enables you to filter and save the log. The log file is also automatically saved to `/var/log/install.log`.

3. To verify that the shell can find and run the `aws` command in your `$PATH`, use the following commands.

```
$ which aws
/usr/local/bin/aws
$ aws --version
aws-cli/2.15.30 Python/3.11.6 Darwin/23.3.0 botocore/2.4.5
```

If the `aws` command cannot be found, you might need to restart your terminal or follow the troubleshooting in [Troubleshoot errors](#).

Command line installer - All users

If you have `sudo` permissions, you can install the AWS CLI for all users on the computer. We provide the steps in one easy to copy and paste group. See the descriptions of each line in the following steps.

```
$ curl "https://awscli.amazonaws.com/AWSCLIV2.pkg" -o "AWSCLIV2.pkg"
$ sudo installer -pkg AWSCLIV2.pkg -target /
```

Guided installation instructions

1. Download the file using the `curl` command. The `-o` option specifies the file name that the downloaded package is written to. In this example, the file is written to `AWSCLIV2.pkg` in the current folder.

```
$ curl "https://awscli.amazonaws.com/AWSCLIV2.pkg" -o "AWSCLIV2.pkg"
```

2. Run the standard macOS installer program, specifying the downloaded `.pkg` file as the source. Use the `-pkg` parameter to specify the name of the package to install, and the `-target /` parameter for which drive to install the package to. The files are installed to `/usr/local/aws-cli`, and a symlink is automatically created in `/usr/local/bin`. You must include `sudo` on the command to grant write permissions to those folders.

```
$ sudo installer -pkg ./AWSCLIV2.pkg -target /
```

After installation is complete, debug logs are written to `/var/log/install.log`.

3. To verify that the shell can find and run the `aws` command in your `$PATH`, use the following commands.

```
$ which aws
/usr/local/bin/aws
$ aws --version
aws-cli/2.15.30 Python/3.11.6 Darwin/23.3.0 botocore/2.4.5
```

If the `aws` command cannot be found, you might need to restart your terminal or follow the troubleshooting in [Troubleshoot errors](#).

Command line - Current user

1. To specify which folder the AWS CLI is installed to, you must create an XML file with any file name. This file is an XML-formatted file that looks like the following example. Leave all values as shown, except you must replace the path `/Users/myusername` in line 9 with the path to the folder you want the AWS CLI installed to. *The folder must already exist, or the command fails.* The following XML example, named `choices.xml`, specifies the installer to install the AWS CLI in the folder `/Users/myusername`, where it creates a folder named `aws-cli`.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
  <array>
    <dict>
      <key>choiceAttribute</key>
      <string>customLocation</string>
      <key>attributeSetting</key>
      <string>/Users/myusername</string>
      <key>choiceIdentifier</key>
      <string>default</string>
    </dict>
  </array>
</plist>
```

2. Download the pkg installer using the `curl` command. The `-o` option specifies the file name that the downloaded package is written to. In this example, the file is written to `AWSCLI2.pkg` in the current folder.

```
$ curl "https://awscli.amazonaws.com/AWSCLI2.pkg" -o "AWSCLI2.pkg"
```

3. Run the standard macOS installer program with the following options:
 - Specify the name of the package to install by using the `-pkg` parameter.
 - Specify installing to a *current user only* by setting the `-target` parameter to `CurrentUserHomeDirectory`.
 - Specify the path (relative to the current folder) and name of the XML file that you created in the `-applyChoiceChangesXML` parameter.

The following example installs the AWS CLI in the folder `/Users/myusername/aws-cli`.

```
$ installer -pkg AWSCLIV2.pkg \  
            -target CurrentUserHomeDirectory \  
            -applyChoiceChangesXML choices.xml
```

4. Because standard user permissions typically don't allow writing to folders in your `$PATH`, the installer in this mode doesn't try to add the symlinks to the `aws` and `aws_completer` programs. For the AWS CLI to run correctly, you must manually create the symlinks after the installer finishes. If your `$PATH` includes a folder you can write to and you specify the folder as the target's path, you can run the following command without `sudo`. If you don't have a writable folder in your `$PATH`, you must use `sudo` for permissions to write to the specified target folder. The default location for a symlink is `/usr/local/bin/`. Replace `folder/installed` with the path to your AWS CLI installation.

```
$ sudo ln -s /folder/installed/aws-cli/aws /usr/local/bin/aws  
$ sudo ln -s /folder/installed/aws-cli/aws_completer /usr/local/bin/  
aws_completer
```

After installation is complete, debug logs are written to `/var/log/install.log`.

5. To verify that the shell can find and run the `aws` command in your `$PATH`, use the following commands.

```
$ which aws  
/usr/local/bin/aws  
$ aws --version  
aws-cli/2.15.30 Python/3.11.6 Darwin/23.3.0 botocore/2.4.5
```

If the `aws` command cannot be found, you might need to restart your terminal or follow the troubleshooting in [Troubleshoot errors](#).

Windows

Install and update requirements

- We support the AWS CLI on Microsoft-supported versions of 64-bit Windows.
- Admin rights to install software

Install or update the AWS CLI

To update your current installation of AWS CLI on Windows, download a new installer each time you update to overwrite previous versions. AWS CLI is updated regularly. To see when the latest version was released, see the [AWS CLI version 2 Changelog](#) on *GitHub*.

1. Download and run the AWS CLI MSI installer for Windows (64-bit):

<https://awscli.amazonaws.com/AWSCLIV2.msi>

Alternatively, you can run the `msiexec` command to run the MSI installer.

```
C:\> msiexec.exe /i https://awscli.amazonaws.com/AWSCLIV2.msi
```

For various parameters that can be used with `msiexec`, see [msiexec](#) on the *Microsoft Docs* website. For example, you can use the `/qn` flag for a silent installation.

```
C:\> msiexec.exe /i https://awscli.amazonaws.com/AWSCLIV2.msi /qn
```

2. To confirm the installation, open the **Start** menu, search for `cmd` to open a command prompt window, and at the command prompt use the `aws --version` command.

```
C:\> aws --version
aws-cli/2.15.30 Python/3.11.6 Windows/10 exe/AMD64 prompt/off
```

If Windows is unable to find the program, you might need to close and reopen the command prompt window to refresh the path, or follow the troubleshooting in [Troubleshoot errors](#).

Troubleshooting AWS CLI install and uninstall errors

If you come across issues after installing or uninstalling the AWS CLI, see [Troubleshoot errors](#) for troubleshooting steps. For the most relevant troubleshooting steps, see [the section called "Command not found errors"](#), [the section called "The "aws --version" command returns a different version than you installed"](#), and [the section called "The "aws --version" command returns a version after uninstalling the AWS CLI"](#).

Next steps

After you successfully install the AWS CLI, you can safely delete your downloaded installer files. After completing the steps in [the section called “Prerequisites”](#) and installing the AWS CLI, you should perform a [the section called “Setup”](#).

Install past releases of the AWS CLI version 2

This topic describes how to install the past releases of the AWS Command Line Interface version 2 (AWS CLI) on supported operating systems. For information on the AWS CLI version 2 releases, see the [AWS CLI version 2 Changelog](#) on GitHub.

AWS CLI version 2 installation instructions:

Linux

Installation requirements

- You know which release of the AWS CLI version 2 you'd like to install. For a list of versions, see the [AWS CLI version 2 Changelog](#) on *GitHub*.
- You must be able to extract or "unzip" the downloaded package. If your operating system doesn't have the built-in unzip command, use an equivalent.
- The AWS CLI version 2 uses `glibc`, `groff`, and `less`. These are included by default in most major distributions of Linux.
- We support the AWS CLI version 2 on 64-bit versions of recent distributions of CentOS, Fedora, Ubuntu, Amazon Linux 1, Amazon Linux 2 and Linux ARM.
- Because AWS doesn't maintain third-party repositories, we can't guarantee that they contain the latest version of the AWS CLI.

Installation instructions

Follow these steps from the command line to install the AWS CLI on Linux.

We provide the steps in one easy to copy and paste group based on whether you use 64-bit Linux or Linux ARM. See the descriptions of each line in the steps that follow.

Linux x86 (64-bit)

Note

(Optional) The following command block downloads and installs the AWS CLI without first verifying the integrity of your download. To verify the integrity of your download, use the below step by step instructions.

For a list of versions, see the [AWS CLI version 2 Changelog](#) on *GitHub*.

To install the AWS CLI, run the following commands.

To specify a version, append a hyphen and the version number to the filename. For this example the filename for version *2.0.30* would be `awscli-exe-linux-x86_64-2.0.30.zip` resulting in the following command:

```
$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64-2.0.30.zip" -o
  "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
```

To update your current installation of the AWS CLI, add your existing symlink and installer information to construct the `install` command using the `--bin-dir`, `--install-dir`, and `--update` parameters. The following command block uses an example symlink of */usr/local/bin* and example installer location of */usr/local/aws-cli*.

```
$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64-2.0.30.zip" -o
  "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install --bin-dir /usr/local/bin --install-dir /usr/local/aws-cli --
update
```

Linux ARM

Note

(Optional) The following command block downloads and installs the AWS CLI without first verifying the integrity of your download. To verify the integrity of your download, use the below step by step instructions.

For a list of versions, see the [AWS CLI version 2 Changelog](#) on *GitHub*.

To install the AWS CLI, run the following commands.

To specify a version, append a hyphen and the version number to the filename. For this example the filename for version *2.0.30* would be `awscli-exe-linux-aarch64-2.0.30.zip` resulting in the following command:

```
$ curl "https://awscli.amazonaws.com/awscli-exe-linux-aarch64-2.0.30.zip" -o
  "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
```

To update your current installation of the AWS CLI, add your existing symlink and installer information to construct the `install` command using the `--bin-dir`, `--install-dir`, and `--update` parameters. The following command block uses an example symlink of `/usr/local/bin` and example installer location of `/usr/local/aws-cli`.

```
$ curl "https://awscli.amazonaws.com/awscli-exe-linux-aarch64-2.0.30.zip" -o
  "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install --bin-dir /usr/local/bin --install-dir /usr/local/aws-cli --
update
```

1. Download the installation file in one of the following ways:

Linux x86 (64-bit)

- **Use the `curl` command** – The `-o` option specifies the file name that the downloaded package is written to. The options on the following example command write the downloaded file to the current directory with the local name `awscliv2.zip`.

To specify a version, append a hyphen and the version number to the filename. For this example the filename for version *2.0.30* would be `awscli-exe-linux-x86_64-2.0.30.zip` resulting in the following command:

```
$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64-2.0.30.zip" -o
  "awscliv2.zip"
```

For a list of versions, see the [AWS CLI version 2 Changelog](#) on *GitHub*.

- **Downloading from the URL –**

In your browser, download your specific version of the AWS CLI by appending a hyphen and the version number to the filename.

```
https://awscli.amazonaws.com/awscli-exe-linux-x86_64-version.number.zip
```

For this example the filename for version *2.0.30* would be `awscli-exe-linux-x86_64-2.0.30.zip` resulting in the following link: https://awscli.amazonaws.com/awscli-exe-linux-x86_64-2.0.30.zip

Linux ARM

- **Use the `curl` command –** The `-o` option specifies the file name that the downloaded package is written to. The options on the following example command write the downloaded file to the current directory with the local name `awscliv2.zip`.

To specify a version, append a hyphen and the version number to the filename.

For this example the filename for version *2.0.30* would be `awscli-exe-linux-aarch64-2.0.30.zip` resulting in the following command:

```
$ curl "https://awscli.amazonaws.com/awscli-exe-linux-aarch64-2.0.30.zip" -o  
  "awscliv2.zip"  
unzip awscliv2.zip  
sudo ./aws/install
```

- **Downloading from the URL –**

In your browser, download your specific version of the AWS CLI by appending a hyphen and the version number to the filename.

```
https://awscli.amazonaws.com/awscli-exe-linux-aarch64-version.number.zip
```

For this example the filename for version *2.0.30* would be `awscli-exe-linux-aarch64-2.0.30.zip` resulting in the following link: <https://awscli.amazonaws.com/awscli-exe-linux-aarch64-2.0.30.zip>

2. (Optional) Verifying the integrity of your downloaded zip file

If you chose to manually download the AWS CLI installer package .zip in the above steps, you can use the following steps to verify the signatures by using the GnuPG tool.

The AWS CLI installer package .zip files are cryptographically signed using PGP signatures. If there is any damage or alteration of the files, this verification fails and you should not proceed with installation.

- a. Download and install the gpg command using your package manager. For more information about GnuPG, see the [GnuPG website](#).
- b. To create the public key file, create a text file and paste in the following text.

```
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBF2Cr7UBEADJZHcgus0J17ENSyumXh85z0TRV0xJorM2B/JL0kH0yigQ1uUG
ZMLhEnAG0bYatdrKP+3H911vK050pXwn0/R7fB/FSTouki4ciIx50uLlnJZIxSzx
PqG10mkxImLnbGwoi6Lto0LYxqHN2iQtz1wTVmq9733zd3XfcXrZ3+Lb1HAGeT5G
TfNxEKJ8soPLyWmwDH6HWcnjZ/aIQRBTIQ05uVeEoYxSh6w0ai7ss/KveoSNBbYz
gbdzoqI2Y8cgH2nbfpg3DSasaLZEdCSsIsK1u05CinE7k2qZ7KgKAUIcT/cR/grk
C6VwsnDU00UCideXcQ8WeHutqvgZH1JgKDbznoIzeQHJD238GEu+eKhRHcz8/jeG
94zkcgJ0z3KbZGYMiTh277Fvj9zzvZsbMBCedV1BTg3TqgvDX4bdkhf5cH+7NtW0
1rFj6UwAsGukBTA0xC01/dnSmZhJ7Z1KmEWilro/g0rjt0xqRQut1IqG22TaqoPG
fYVN+en3ZwbT97kcgZDwqbuykNt64oZwC4XKCa3mprEGC3IbJTBFqg1XmZ719ywG
EEUJY01b2XrSuPwml39beWdKM8kzr10jnl0m6+lpTRCBfo0wa9F8YZRhHPAkWkKX
XDe0GpWRj4oh0x0d2GWkyV5xyN14p2tQ0Cd00Dmz80yUTgRpPVQUt0EHXQARAQAB
tCFBV1MgQ0xJIFRlYW0gPGF3cy1jbG1AYW1hem9uLmNvbT6JAlQEEwEIAD4CGwMF
CwkIBwIGFQoJCAAsCBBYCAwECHgECF4AWIQT7Xbd/1cEYuAURraimMQrMRnJHXAUc
ZMKcEgUJCSEf3QAKCRCmMQrMRnJHXCIld/4vior9J5tB+icri5WbDudS3ak/ve4q
XS6ZLm5S81+CBxy5aLQUlyFhuuaEHDC11fG780duxatzeHENASYVo3mmKNwrCBza
NJaeaWKLGT0MKwBSP5aa3dva8P/4oUP9GsQn0uWoXwNDwfrMbNI8gn+jC/3MigW
vD3fu6zC0WwLITNv2SJoQ1wILmb/uGfha68o4iTB0vcftVRua06DyqF+CrHX/0j0
k1EDQFMY9M4tsYT7X8NwfI8Vmc89nzpVl9fwda44WwpKIw1FBZP8S0sgDx2xDsxv
L8kM2Gt0iH0cHqF0+V7xtTKZy1o1iDbJKhu80Kc+YC/TmozD8oeGU2rEfxFLegwS
zT9N+jB38+dqaP9pRDsi45iGqyA8yavVBabpL0IQ9jU6eIV+kmcjIjcun/Uo8SjJ
0xQAsm41rxPaKV6vJUn10wVnuhSkKk8mzN01SZwu7Hua6rDCaGeB8uJ44AP3QzW
BNnrjtoN6A1N0D2wFmFE/YL/rHPxU1XwPntubYB/t3rXFL7ENQ00QH0KVXgRC1ey
sHMglg46c+nQLRzVTshjDjmtzvH9rcV9RKRoPetEggzCoD89veDA9jPR2Kw6RYkS
XzYm2fEv16/HRNYt7hJzneFqRIjHW5qAgSs/bcaRwPAU/QQzzJPVKCQNr4y0weyg
B8HCtGjfod0p1A==
=gdMc
-----END PGP PUBLIC KEY BLOCK-----
```

For reference, the following are the details of the public key.

```
Key ID:           A6310ACC4672
Type:            RSA
Size:           4096/4096
Created:        2019-09-18
Expires:       2024-07-26
User ID:       AWS CLI Team <aws-cli@amazon.com>
Key fingerprint: FB5D B77F D5C1 18B8 0511 ADA8 A631 0ACC 4672 475C
```

- c. Import the AWS CLI public key with the following command, substituting *public-key-file-name* with the file name of the public key you created.

```
$ gpg --import public-key-file-name
gpg: /home/username/.gnupg/trustdb.gpg: trustdb created
gpg: key A6310ACC4672475C: public key "AWS CLI Team <aws-cli@amazon.com>"
imported
gpg: Total number processed: 1
gpg:             imported: 1
```

- d. Download the AWS CLI signature file for the package you downloaded. It has the same path and name as the .zip file it corresponds to, but has the extension .sig. In the following examples, we save it to the current directory as a file named awscliv2.sig.

Linux x86 (64-bit)

For the latest version of the AWS CLI, use the following command block:

```
$ curl -o awscliv2.sig https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip.sig
```

For a specific version of the AWS CLI, append a hyphen and the version number to the filename. For this example the filename for version *2.0.30* would be awscli-exe-linux-x86_64-2.0.30.zip.sig resulting in the following command:

```
$ curl -o awscliv2.sig https://awscli.amazonaws.com/awscli-exe-linux-x86_64-2.0.30.zip.sig
```

For a list of versions, see the [AWS CLI version 2 Changelog](#) on *GitHub*.

Linux ARM

For the latest version of the AWS CLI, use the following command block:

```
$ curl -o awscliv2.sig https://awscli.amazonaws.com/awscli-exe-linux-aarch64.zip.sig
```

For a specific version of the AWS CLI, append a hyphen and the version number to the filename. For this example the filename for version **2.0.30** would be `awscli-exe-linux-aarch64-2.0.30.zip.sig` resulting in the following command:

```
$ curl -o awscliv2.sig https://awscli.amazonaws.com/awscli-exe-linux-aarch64-2.0.30.zip.sig
```

For a list of versions, see the [AWS CLI version 2 Changelog](#) on *GitHub*.

- e. Verify the signature, passing both the downloaded `.sig` and `.zip` file names as parameters to the `gpg` command.

```
$ gpg --verify awscliv2.sig awscliv2.zip
```

The output should look similar to the following.

```
gpg: Signature made Mon Nov  4 19:00:01 2019 PST
gpg:                using RSA key FB5D B77F D5C1 18B8 0511 ADA8 A631 0ACC 4672
475C
gpg: Good signature from "AWS CLI Team <aws-cli@amazon.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:                There is no indication that the signature belongs to the owner.
Primary key fingerprint: FB5D B77F D5C1 18B8 0511 ADA8 A631 0ACC 4672 475C
```

Important

The warning in the output is expected and doesn't indicate a problem. It occurs because there isn't a chain of trust between your personal PGP key (if you have one) and the AWS CLI PGP key. For more information, see [Web of trust](#).

3. Unzip the installer. If your Linux distribution doesn't have a built-in unzip command, use an equivalent to unzip it. The following example command unzips the package and creates a directory named `aws` under the current directory.

```
$ unzip awscliv2.zip
```

4. Run the install program. The installation command uses a file named `install` in the newly unzipped `aws` directory. By default, the files are all installed to `/usr/local/aws-cli`, and a symbolic link is created in `/usr/local/bin`. The command includes `sudo` to grant write permissions to those directories.

```
$ sudo ./aws/install
```

You can install without `sudo` if you specify directories that you already have write permissions to. Use the following instructions for the `install` command to specify the installation location:

- Ensure that the paths you provide to the `-i` and `-b` parameters contain no volume name or directory names that contain any space characters or other white space characters. If there is a space, the installation fails.
- `--install-dir` or `-i` – This option specifies the directory to copy all of the files to.

The default value is `/usr/local/aws-cli`.

- `--bin-dir` or `-b` – This option specifies that the main `aws` program in the `install` directory is symbolically linked to the file `aws` in the specified path. You must have write permissions to the specified directory. Creating a symlink to a directory that is already in your path eliminates the need to add the `install` directory to the user's `$PATH` variable.

The default value is `/usr/local/bin`.

```
$ ./aws/install -i /usr/local/aws-cli -b /usr/local/bin
```

Note

To update your current installation of the AWS CLI version 2 to a newer version, add your existing symlink and installer information to construct the `install` command with the `--update` parameter.

```
$ sudo ./aws/install --bin-dir /usr/local/bin --install-dir /usr/local/aws-  
cli --update
```

To locate the existing symlink and installation directory, use the following steps:

1. Use the `which` command to find your symlink. This gives you the path to use with the `--bin-dir` parameter.

```
$ which aws  
/usr/local/bin/aws
```

2. Use the `ls` command to find the directory that your symlink points to. This gives you the path to use with the `--install-dir` parameter.

```
$ ls -l /usr/local/bin/aws  
lrwxrwxrwx 1 ec2-user ec2-user 49 Oct 22 09:49 /usr/local/bin/aws -> /usr/  
local/aws-cli/v2/current/bin/aws
```

5. Confirm the installation with the following command.

```
$ aws --version  
aws-cli/2.15.30 Python/3.11.6 Linux/5.10.205-195.807.amzn2.x86_64 botocore/2.4.5
```

If the `aws` command cannot be found, you might need to restart your terminal or follow the troubleshooting in [Troubleshoot errors](#).

(Optional) Verifying the integrity of your downloaded zip file

If you chose to manually download the AWS CLI version 2 installer package `.zip` in the above steps, you can use the following steps to verify the signatures by using the GnuPG tool.

The AWS CLI version 2 installer package `.zip` files are cryptographically signed using PGP signatures. If there is any damage or alteration of the files, this verification fails and you should not proceed with installation.

1. Download and install the `gpg` command using your package manager. For more information about GnuPG, see the [GnuPG website](#).
2. To create the public key file, create a text file and paste in the following text.

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
```

```
mQINBF2Cr7UBEADJZHcgus0Jl7ENSyumXh85z0TRV0xJorM2B/JL0kH0yigQ1uUG
ZMLhENaG0bYatdrKP+3H911vK050pXwn0/R7fB/FSTouki4ciIx50uLlnJZIxSzx
PqG10mkxImLnbGwoi6Lto0LYxqHN2iQtzlwTVmq9733zd3XfcXrZ3+Lb1HAgEt5G
TfnxEKJ8soPLyWmwDH6HWcnjZ/aIQRBTIQ05uVeEoYxSh6w0ai7ss/KveoSNBbYz
gbdzoqI2Y8cgh2bnfpgp3DSasaLZEdCSsIsK1u05CinE7k2qZ7KgKAUIcT/cR/grk
C6VwsnDU00UCideXcQ8WeHutqvgZH1JgKDbznoIzeQHJD238GEu+eKhRHcz8/jeG
94zkcqJ0z3KbZGYMiTh277Fvj9zzvZsbMBCedV1BTg3TqgvDX4bdkhf5cH+7NtW0
lrFj6UwAsGukBTA0xC01/dnSmZhJ7Z1KmEWilro/g0rjt0xqRQutlIqG22TaqoPG
fYVN+en3ZwbT97kcgZDwqbuykNt64oZwC4XKCa3mprEGC3IbJTBFqglXmZ719ywG
EEUJY01b2XrSuPwm139beWdKM8kzr10jnl0m6+lpTRCBfo0wa9F8YZRhHPAkWkKX
XDe0GpWrj4oh0x0d2GWkyV5xyN14p2tQ0Cd00Dmz80yUTgRpPVQuT0EHXQARAQAB
tCFBV1MgQ0xJIFRlYW0gPGF3cy1jbG1AYW1hem9uLmNvbT6JAlQEEwEiAD4CGwMF
CwkIBwIGFQoJCAcCBYCAwECHgECF4AWIQT7Xbd/1cEYuAURraimMQrMRnJHXAUc
ZMKcEgUJCSEf3QAKCRCmMQrMRnJHXciLD/4vior9J5tB+icri5WbDudS3ak/ve4q
XS6ZLm5S8l+CBxy5aLQUlyFhuaaEHDC11fg780duxatzeHENASYVo3mmKNwrCBza
NJaeaWKLgQT0MKwBSP5aa3dva8P/4oUP9GsQn0uWoXwNDWfrMbNI8gn+jC/3MigW
vD3fu6zC0WWLITNv2SJoQ1wILmb/uGfha68o4iTB0vcftVRua06DyqF+CrHX/0j0
kLEDQFMY9M4tsYT7X8NwfI8Vmc89nzpVL9fwda44WwpKIw1FBZP8S0sgDx2xDsxv
L8km2Gt0iH0cHqF0+V7xtTKZylo1iDbJKhu80Kc+YC/TmozD8oeGU2rEFXfLegwS
zT9N+jB38+dqaP9pRdsi45iGqyA8yavVBabpL0IQ9jU6eIV+kmcjIjcun/Uo8SjJ
0xQAsm41rxPaKV6vJUn10wVNUhSkKk8mzN01SZwu7Hua6rdcCaGeB8uJ44AP3QzW
BNnrjtoN6A1N0D2wFmfE/YL/rHPxU1XwPntubYB/t3rXFL7ENQ00QH0KVXgRCley
sHMglg46c+nQLRzVTshjDjmtzvH9rcV9RKR0PetEggzCoD89veDA9jPR2Kw6RYkS
XzYm2fEv16/HRNYt7hJzneFqRIjHW5qAgSs/bcaRwPAU/QQzzJPVKCQNr4y0weyg
B8HCtGjfod0p1A==
=gdMc
```

```
-----END PGP PUBLIC KEY BLOCK-----
```

For reference, the following are the details of the public key.

```
Key ID:          A6310ACC4672
Type:            RSA
Size:            4096/4096
Created:         2019-09-18
Expires:         2024-07-26
User ID:         AWS CLI Team <aws-cli@amazon.com>
Key fingerprint: FB5D B77F D5C1 18B8 0511 ADA8 A631 0ACC 4672 475C
```

3. Import the AWS CLI public key with the following command, substituting *public-key-file-name* with the file name of the public key you created.

```
$ gpg --import public-key-file-name
gpg: /home/username/.gnupg/trustdb.gpg: trustdb created
gpg: key A6310ACC4672475C: public key "AWS CLI Team <aws-cli@amazon.com>" imported
gpg: Total number processed: 1
gpg:             imported: 1
```

4. Download the AWS CLI signature file for the package you downloaded. It has the same path and name as the .zip file it corresponds to, but has the extension .sig. In the following examples, we save it to the current directory as a file named awscliv2.sig.

Linux x86 (64-bit)

For the latest version of the AWS CLI, use the following command block:

```
$ curl -o awscliv2.sig https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip.sig
```

For a specific version of the AWS CLI, append a hyphen and the version number to the filename. For this example the filename for version *2.0.30* would be awscli-exe-linux-x86_64-2.0.30.zip.sig resulting in the following command:

```
$ curl -o awscliv2.sig https://awscli.amazonaws.com/awscli-exe-linux-x86_64-2.0.30.zip.sig
```

For a list of versions, see the [AWS CLI version 2 Changelog](#) on *GitHub*.

Linux ARM

For the latest version of the AWS CLI, use the following command block:

```
$ curl -o awscliv2.sig https://awscli.amazonaws.com/awscli-exe-linux-aarch64.zip.sig
```

For a specific version of the AWS CLI, append a hyphen and the version number to the filename. For this example the filename for version *2.0.30* would be awscli-exe-linux-aarch64-2.0.30.zip.sig resulting in the following command:

```
$ curl -o awscliv2.sig https://awscli.amazonaws.com/awscli-exe-linux-aarch64-2.0.30.zip.sig
```

For a list of versions, see the [AWS CLI version 2 Changelog](#) on *GitHub*.

5. Verify the signature, passing both the downloaded `.sig` and `.zip` file names as parameters to the `gpg` command.

```
$ gpg --verify awscliv2.sig awscliv2.zip
```

The output should look similar to the following.

```
gpg: Signature made Mon Nov  4 19:00:01 2019 PST
gpg:                using RSA key FB5D B77F D5C1 18B8 0511 ADA8 A631 0ACC 4672 475C
gpg: Good signature from "AWS CLI Team <aws-cli@amazon.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:                There is no indication that the signature belongs to the owner.
Primary key fingerprint: FB5D B77F D5C1 18B8 0511 ADA8 A631 0ACC 4672 475C
```

Important

The warning in the output is expected and doesn't indicate a problem. It occurs because there isn't a chain of trust between your personal PGP key (if you have one) and the AWS CLI PGP key. For more information, see [Web of trust](#).

macOS

Installation requirements

- You know which release of the AWS CLI version 2 you'd like to install. For a list of versions, see the [AWS CLI version 2 Changelog](#) on *GitHub*.
- We support the AWS CLI version 2 on Apple-supported versions of 64-bit macOS.
- Because AWS doesn't maintain third-party repositories, we can't guarantee that they contain the latest version of the AWS CLI.

Installation instructions

You can install the AWS CLI version 2 on macOS in the following ways.

GUI installer

The following steps show how to install or update to the latest version of the AWS CLI version 2 by using the standard macOS user interface and your browser. If you are updating to the latest version, use the same installation method that you used for your current version.

1. In your browser, download your specific version of the AWS CLI by appending a hyphen and the version number to the filename.

```
https://awscli.amazonaws.com/AWSCLIV2-version.number.pkg
```

For this example, the filename for version *2.0.30* would be `AWSCLIV2-2.0.30.pkg` resulting in the following link: <https://awscli.amazonaws.com/AWSCLIV2-2.0.30.pkg>.

2. Run your downloaded file and follow the on-screen instructions. You can choose to install the AWS CLI version 2 in the following ways:

- **For all users on the computer (requires sudo)**

- You can install to any folder, or choose the recommended default folder of `/usr/local/aws-cli`.
- The installer automatically creates a symlink at `/usr/local/bin/aws` that links to the main program in the installation folder you chose.

- **For only the current user (doesn't require sudo)**

- You can install to any folder to which you have write permission.
- Due to standard user permissions, after the installer finishes, you must manually create a symlink file in your `$PATH` that points to the `aws` and `aws_completer` programs by using the following commands at the command prompt. If your `$PATH` includes a folder you can write to, you can run the following command without `sudo` if you specify that folder as the target's path. If you don't have a writable folder in your `$PATH`, you must use `sudo` in the commands to get permissions to write to the specified target folder. The default location for a symlink is `/usr/local/bin/`.

```
$ sudo ln -s /folder/installed/aws-cli/aws /usr/local/bin/aws
$ sudo ln -s /folder/installed/aws-cli/aws_completer /usr/local/bin/
aws_completer
```

Note

You can view debug logs for the installation by pressing **Cmd+L** anywhere in the installer. This opens a log pane that enables you to filter and save the log. The log file is also automatically saved to `/var/log/install.log`.

3. To verify that the shell can find and run the `aws` command in your `$PATH`, use the following commands.

```
$ which aws
/usr/local/bin/aws
$ aws --version
aws-cli/2.15.30 Python/3.11.6 Darwin/23.3.0 botocore/2.4.5
```

If the `aws` command cannot be found, you might need to restart your terminal or follow the troubleshooting in [Troubleshoot errors](#).

Command line installer - All users

If you have `sudo` permissions, you can install the AWS CLI version 2 for all users on the computer. We provide the steps in one easy to copy and paste group. See the descriptions of each line in the following steps.

For a specific version of the AWS CLI, append a hyphen and the version number to the filename. For this example the filename for version `2.0.30` would be `AWSCLI2-2.0.30.pkg` resulting in the following command:

```
$ curl "https://awscli.amazonaws.com/AWSCLI2-2.0.30.pkg" -o "AWSCLI2.pkg"
$ sudo installer -pkg AWSCLI2.pkg -target /
```

1. Download the file using the `curl` command. The `-o` option specifies the file name that the downloaded package is written to. In this example, the file is written to `AWSCLI2.pkg` in the current folder.

For a specific version of the AWS CLI, append a hyphen and the version number to the filename. For this example the filename for version `2.0.30` would be `AWSCLI2-2.0.30.pkg` resulting in the following command:

```
$ curl "https://awscli.amazonaws.com/AWSCLIV2-2.0.30.pkg" -o "AWSCLIV2.pkg"
```

For a list of versions, see the [AWS CLI version 2 Changelog](#) on *GitHub*.

2. Run the standard macOS installer program, specifying the downloaded .pkg file as the source. Use the `-pkg` parameter to specify the name of the package to install, and the `-target /` parameter for which drive to install the package to. The files are installed to `/usr/local/aws-cli`, and a symlink is automatically created in `/usr/local/bin`. You must include `sudo` on the command to grant write permissions to those folders.

```
$ sudo installer -pkg ./AWSCLIV2.pkg -target /
```

After installation is complete, debug logs are written to `/var/log/install.log`.

3. To verify that the shell can find and run the `aws` command in your `$PATH`, use the following commands.

```
$ which aws
/usr/local/bin/aws
$ aws --version
aws-cli/2.15.30 Python/3.11.6 Darwin/23.3.0 botocore/2.4.5
```

If the `aws` command cannot be found, you might need to restart your terminal or follow the troubleshooting in [Troubleshoot errors](#).

Command line - Current user

1. To specify which folder the AWS CLI is installed to, you must create an XML file. This file is an XML-formatted file that looks like the following example. Leave all values as shown, except you must replace the path `/Users/myusername` in line 9 with the path to the folder you want the AWS CLI version 2 installed to. *The folder must already exist, or the command fails.* This XML example specifies that the installer installs the AWS CLI in the folder `/Users/myusername`, where it creates a folder named `aws-cli`.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
```

```
<array>
  <dict>
    <key>choiceAttribute</key>
    <string>customLocation</string>
    <key>attributeSetting</key>
    <string>/Users/myusername</string>
    <key>choiceIdentifier</key>
    <string>default</string>
  </dict>
</array>
</plist>
```

2. Download the pkg installer using the `curl` command. The `-o` option specifies the file name that the downloaded package is written to. In this example, the file is written to `AWSCLI2.pkg` in the current folder.

For the specific version of the AWS CLI, append a hyphen and the version number to the filename. For this example the filename for version `2.0.30` would be `AWSCLI2-2.0.30.pkg` resulting in the following command:

```
$ curl "https://awscli.amazonaws.com/AWSCLI2-2.0.30.pkg" -o "AWSCLI2.pkg"
```

For a list of versions, see the [AWS CLI version 2 Changelog](#) on *GitHub*.

3. Run the standard macOS installer program with the following options:
 - Specify the name of the package to install by using the `-pkg` parameter.
 - Specify installing to a *current user only* by setting the `-target` parameter to `CurrentUserHomeDirectory`.
 - Specify the path (relative to the current folder) and name of the XML file that you created in the `-applyChoiceChangesXML` parameter.

The following example installs the AWS CLI in the folder `/Users/myusername/aws-cli`.

```
$ installer -pkg AWSCLI2.pkg \
            -target CurrentUserHomeDirectory \
            -applyChoiceChangesXML choices.xml
```

4. Because standard user permissions typically don't allow writing to folders in your `$PATH`, the installer in this mode doesn't try to add the symlinks to the `aws` and `aws_completer`

programs. For the AWS CLI to run correctly, you must manually create the symlinks after the installer finishes. If your `$PATH` includes a folder you can write to and you specify the folder as the target's path, you can run the following command without `sudo`. If you don't have a writable folder in your `$PATH`, you must use `sudo` for permissions to write to the specified target folder. The default location for a symlink is `/usr/local/bin/`.

```
$ sudo ln -s /folder/installed/aws-cli/aws /usr/local/bin/aws
$ sudo ln -s /folder/installed/aws-cli/aws_completer /usr/local/bin/
aws_completer
```

After installation is complete, debug logs are written to `/var/log/install.log`.

5. To verify that the shell can find and run the `aws` command in your `$PATH`, use the following commands.

```
$ which aws
/usr/local/bin/aws
$ aws --version
aws-cli/2.15.30 Python/3.11.6 Darwin/23.3.0 botocore/2.4.5
```

If the `aws` command cannot be found, you might need to restart your terminal or follow the troubleshooting in [Troubleshoot errors](#).

Windows

Installation requirements

- You know which release of the AWS CLI version 2 you'd like to install. For a list of versions, see the [AWS CLI version 2 Changelog](#) on *GitHub*.
- We support the AWS CLI on Microsoft-supported versions of 64-bit Windows.
- Admin rights to install software

Installation instructions

To update your current installation of AWS CLI version 2 on Windows, download a new installer each time you update to overwrite previous versions. AWS CLI is updated regularly. To see when the latest version was released, see the [AWS CLI version 2 Changelog](#) on *GitHub*.

1. Download and run the AWS CLI MSI installer for Windows (64-bit) in one of the following ways:

- **Downloading and running the MSI installer:** To create your download link for a specific version of the AWS CLI, append a hyphen and the version number to the filename.

```
https://awscli.amazonaws.com/AWSCLIV2-version.number.msi
```

For this example the filename for version `2.0.30` would be `AWSCLIV2-2.0.30.msi` resulting in the following link: <https://awscli.amazonaws.com/AWSCLIV2-2.0.30.msi>.

- **Using the `msiexec` command:** Alternatively, you can use the MSI installer by adding the link to the `msiexec` command. For a specific version of the AWS CLI, append a hyphen and the version number to the filename.

```
C:\> msiexec.exe /i https://awscli.amazonaws.com/AWSCLIV2-version.number.msi
```

For this example the filename for version `2.0.30` would be `AWSCLIV2-2.0.30.msi` resulting in the following link <https://awscli.amazonaws.com/AWSCLIV2-2.0.30.msi>.

```
C:\> msiexec.exe /i https://awscli.amazonaws.com/AWSCLIV2-2.0.30.msi
```

For various parameters that can be used with `msiexec`, see [msiexec](#) on the *Microsoft Docs* website.

For a list of versions, see the [AWS CLI version 2 Changelog](#) on *GitHub*.

2. To confirm the installation, open the **Start** menu, search for `cmd` to open a command prompt window, and at the command prompt use the `aws --version` command.

```
C:\> aws --version  
aws-cli/2.15.30 Python/3.11.6 Windows/10 exe/AMD64 prompt/off
```

If Windows is unable to find the program, you might need to close and reopen the command prompt window to refresh the path, or follow the troubleshooting in [Troubleshoot errors](#).

Troubleshooting AWS CLI install and uninstall errors

If you come across issues after installing or uninstalling the AWS CLI, see [Troubleshoot errors](#) for troubleshooting steps. For the most relevant troubleshooting steps, see [the section called "Command not found errors"](#), [the section called "The "aws --version" command returns a different version than you installed"](#), and [the section called "The "aws --version" command returns a version after uninstalling the AWS CLI"](#).

Next steps

After completing the steps in [the section called "Prerequisites"](#) and installing the AWS CLI, you should perform a [the section called "Setup"](#).

Build and install the AWS CLI from source

This topic describes how to install or update from source to the latest release of the AWS Command Line Interface (AWS CLI) on supported operating systems.

For information on the latest releases of AWS CLI, see the [AWS CLI version 2 Changelog](#) on GitHub.

Important

AWS CLI versions 1 and 2 use the same aws command name. If you previously installed AWS CLI version 1, see [Migrate from AWS CLI version 1 to version 2](#).

Topics

- [Why build from source?](#)
- [Quicksteps](#)
- [Step 1: Setup all requirements](#)
- [Step 2: Configuring the AWS CLI source installation](#)
- [Step 3: Building the AWS CLI](#)
- [Step 4: Installing the AWS CLI](#)
- [Step 5: Verifying the AWS CLI installation](#)
- [Workflow examples](#)

- [Troubleshooting AWS CLI install and uninstall errors](#)
- [Next steps](#)

Why build from source?

The AWS CLI is [available as pre-built installers](#) for most platforms and environments as well as a Docker image.

Generally, these installers provide coverage for most use-cases. The instructions for installing from source are to help with the use-cases our installers do not cover. Some of these use-cases include the following:

- The pre-built installers do not support your environment. For example, ARM 32-bit is not supported by the pre-built installers.
- The pre-built installers have dependencies your environment lacks. For example, Alpine Linux uses [musl](#), but the current installers require `glibc` causing the pre-built installers to not immediately work.
- The pre-built installers require resources your environment restricts access to. For example, security hardened systems might not give permissions to shared memory. This is needed for the frozen aws installer.
- The pre-built installers are often blockers for maintainers in package managers, as full control over the building process for code and packages is preferred. Building from source enables distribution maintainers a more streamlined process to keep the AWS CLI updated. Enabling maintainers provides customers more up-to-date versions of the AWS CLI when installing from a 3rd party package manager such as `asbrew`, `yum`, and `apt`.
- Customers that patch AWS CLI functionality require building and installing the AWS CLI from source. This is especially important for community members that want to test changes they've made to the source prior to contributing the change to the AWS CLI GitHub repository.

Quicksteps

Note

All code examples are assumed to run from the root of the source directory.

To build and install the AWS CLI from source, follow the steps in this section. The AWS CLI leverages [GNU Autotools](#) to install from source. In the simplest case, the AWS CLI can be installed from source by running the default example commands from the root of the AWS CLI GitHub repository.

1. [Setup all requirements for your environment](#). This includes being able to run [GNU Autotools](#) generated files and Python 3.8 or later is installed.
2. In your terminal, navigate to the top level of the AWS CLI source folder and run the `./configure` command. This command checks the system for all required dependencies and generates a `Makefile` for building and installing the AWS CLI based on detected and specified configurations.

Linux and macOS

The following `./configure` command example sets the build configuration for the AWS CLI using default settings.

```
$ ./configure
```

Windows PowerShell

Before running any commands calling MSYS2, you must preserve your current working directory:

```
PS C:\> $env:HERE_INVOKING = 'yes'
```

Then use the following `./configure` command example to set the build configuration for the AWS CLI using your local path to your Python executable, installing to `C:\Program Files\AWSCLI`, and downloading all dependencies.

```
PS C:\> C:\msys64\usr\bin\bash -lc " PYTHON='C:\path\to\python.exe' ./configure --prefix='C:\Program Files\AWSCLI' --with-download-deps "
```

For details, available configuration options, and default setting information, see the [the section called “Step 2: Configuring the AWS CLI source installation”](#) section.

3. Run the `make` command. This command builds the AWS CLI according to your configuration settings.

The following `make` command example builds with default options using your existing `./configure` settings.

Linux and macOS

```
$ make
```

Windows PowerShell

```
PS C:\> C:\msys64\usr\bin\bash -lc "make"
```

For details and available build options, see the [the section called “Step 3: Building the AWS CLI”](#) section.

4. Run the `make install` command. This command installs your built AWS CLI to the configured location on your system.

The following `make install` command example installs your built AWS CLI and creates symlinks in your configured locations using default command settings.

Linux and macOS

```
$ make install
```

Windows PowerShell

```
PS C:\> C:\msys64\usr\bin\bash -lc "make install"
```

After installing, add the path to the AWS CLI using the following:

```
PS C:\> $Env: PATH += ";C:\Program Files\AWSCLI\bin\"
```

For details and available install options, see the [the section called “Step 4: Installing the AWS CLI”](#) section.

5. Confirm the AWS CLI successfully installed using the following command:

```
$ aws --version
aws-cli/2.15.30 Python/3.11.6 Windows/10 exe/AMD64 prompt/off
```

For troubleshooting steps for install errors see the [the section called “Troubleshooting AWS CLI install and uninstall errors”](#) section.

Step 1: Setup all requirements

To build the AWS CLI from source you need the following completed beforehand:

Note

All code examples are assumed to run from the root of the source directory.

1. Download the AWS CLI source by either forking the AWS CLI GitHub repository or downloading the source tarball. The instructions is one of the following:
 - Fork and clone the [AWS CLI repository](#) from *GitHub*. For more information, see [Fork a repo](#) in the *GitHub Docs*.
 - Download the latest source tarball at <https://awscli.amazonaws.com/awscli.tar.gz> extract the contents using the following commands:

```
$ curl -o awscli.tar.gz https://awscli.amazonaws.com/awscli.tar.gz
$ tar -xzf awscli.tar.gz
```

Note

To download a specific version, use the following link format: <https://awscli.amazonaws.com/awscli-versionnumber.tar.gz>

For example, for version 2.10.0 the link is the following: <https://awscli.amazonaws.com/awscli-2.10.0.tar.gz>

Source versions are available starting with version **2.10.0** of the AWS CLI.

(Optional) Verifying the integrity of your downloaded zip file by completing the following steps:

1. You can use the following steps to verify the signatures by using the GnuPG tool.

The AWS CLI installer package .zip files are cryptographically signed using PGP signatures. If there is any damage or alteration of the files, this verification fails and you should not proceed with installation.

2. Download and install the gpg command using your package manager. For more information about GnuPG, see the [GnuPG website](#).
3. To create the public key file, create a text file and paste in the following text.

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
```

```
mQINBF2Cr7UBEADJZHcgus0J17ENSyumXh85z0TRV0xJorM2B/JL0kH0yigQ1uUG
ZMLhENAG0bYatdrKP+3H911vK050pXwn0/R7fB/FSTouki4ciIx50uLlnJZIXSzx
PqG10mkxImLNBGwoi6Lto0LYxqHN2iQtz1wTVmq9733zd3XfcXrZ3+Lb1HAgEt5G
TfNxEKJ8soPLYWmwDH6HWCnjZ/aIQRBTIQ05uVeEoYxSh6w0ai7ss/KveoSNBbYz
gbdzoqI2Y8cgH2nbfpg3DSasaLZEdCSsIsK1u05CinE7k2qZ7KgKAUIcT/cR/grk
C6VwsnDU00UCideXcQ8WeHutqvgZH1JgKDbznoIzeQHJD238GEu+eKhRHcz8/jeG
94zkcgJ0z3KbZGYMiTh277Fvj9zzvZsbMBCedV1BTg3TqgvdX4bdkhf5cH+7NtW0
1rFj6UwAsGukBTA0xC0l/dnSmZhJ7Z1KmEWi1ro/g0rjt0xqRQut1IqG22TaqoPG
fYVN+en3Zwbt97kcgZDwqbuykNt64oZwc4XKCa3mprEGC3IbJTBfqq1XmZ719ywG
EEUJY01b2XrSuPwml39beWdKM8kzr10jn10m6+1pTRCBfo0wa9F8YZRhHPAkWkKX
XDe0GpWRj4oh0x0d2GWkyV5xyN14p2tQ0Cd00Dmz80yUTgRpPVQUt0EhXQARAQAB
tCFBV1MgQ0xJIFR1YW0gPGF3cy1jbG1AYW1hem9uLmNvbT6JAlQEEwEIAD4WIQT7
Xbd/1cEYuAURraimMQrMRnJHXAUCXYKvtQIbAwUJB4TOAAULCQgHAgYVCgkICwIE
FgIDAQIeAQIXgAAKCRcmMQrMRnJHXJIXEACHLUIkg80uPUkGjE3jejvQSA1aWuAM
yzy6fdpd1RUz6M6nmsUh0ExjVIvibEJpzK5mhuSZ41b0vJ2ZUPgCv4zs2nBd7BGJ
MxKiWgBREgVtdqZ0SzyYH4PYCJSE732x/Fw9hfnh1dMTXNcrQXzw0mmFNNegG00x
au+VnpcR5Kz3smiTrIwZbRudo1ijhCYPQ7t5CMp9kjC6b0bvy1hSIg2xNbMAN/Do
ikebAl36uA6Y/Uczjj3GxZW4ZWeFirmidKbtqvUz2y0UFszobjiBSqZZHCreC34B
hw9bFNpuWC/0SrXgohdsc6vK50pDgDv5kM2qo9tMQ/izsAwTh/d/GzZv8H41V9e0
tEis+EpR497PaxKKh9tJf0N6Q1YLRHof5xePZt0I1S3gfvS5hXA3HJ9yIxb8T0H
QYmVr3aIUes20i6meI3fuV36VFupwfrTKaL7VXnsrK2fq5cRvyJLNzXucg0WAjPF
RrAGLzY7nPlxeg1a0aeP+pdsqjq1PJom80CWc1+6DWbg0jsC74WoesAqgBI0DMB
rsa1ly/q+bPzpsnWjzHV8+1/EtZmSc8ZUGSJOPkfc7h0bnfk118h+1QtKTjZme4d
H17gsBJr+opwJw/Zio2LMjQB0qlm3K1A4zFTh7wBC7He6KPQea1p2XAMgtvAttNe
YLZATHZKTJyiqA==
=vY0k
```

```
-----END PGP PUBLIC KEY BLOCK-----
```

For reference, the following are the details of the public key.

```
Key ID:           A6310ACC4672
Type:             RSA
Size:            4096/4096
Created:         2019-09-18
Expires:        2023-09-17
User ID:        AWS CLI Team <aws-cli@amazon.com>
Key fingerprint: FB5D B77F D5C1 18B8 0511 ADA8 A631 0ACC 4672 475C
```

4. Import the AWS CLI public key with the following command, substituting *public-key-file-name* with the file name of the public key you created.

```
$ gpg --import public-key-file-name
gpg: /home/username/.gnupg/trustdb.gpg: trustdb created
gpg: key A6310ACC4672475C: public key "AWS CLI Team <aws-cli@amazon.com>"
imported
gpg: Total number processed: 1
gpg:             imported: 1
```

5. Download the AWS CLI signature file for the package you downloaded at <https://awscli.amazonaws.com/awscli.tar.gz.sig>. It has the same path and name as the tarball file it corresponds to, but has the extension `.sig`. Save it in the same path as the tarball file. Or use the following command block:

```
$ curl -o awscliv2.sig https://awscli.amazonaws.com/awscli.tar.gz.sig
```

6. Verify the signature, passing both the downloaded `.sig` and `.zip` file names as parameters to the `gpg` command.

```
$ gpg --verify awscliv2.sig awscli.tar.gz
```

The output should look similar to the following.

```
gpg: Signature made Mon Nov  4 19:00:01 2019 PST
gpg:             using RSA key FB5D B77F D5C1 18B8 0511 ADA8 A631 0ACC 4672
475C
gpg: Good signature from "AWS CLI Team <aws-cli@amazon.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
```

```
gpg:          There is no indication that the signature belongs to the owner.  
Primary key fingerprint: FB5D B77F D5C1 18B8 0511  ADA8 A631 0ACC 4672 475C
```

Important

The warning in the output is expected and doesn't indicate a problem. It occurs because there isn't a chain of trust between your personal PGP key (if you have one) and the AWS CLI PGP key. For more information, see [Web of trust](#).

2. You have an environment that can run [GNU Autotools](#) generated files such as `configure` and `Makefile`. These files are widely portable across POSIX platforms.

Linux and macOS

If Autotools is not already installed in your environment or you need to update them, then follow the installation instructions found in [How do I install the Autotools \(as user\)?](#) or [Basic Installation](#) in the *GNU documentation*.

Windows PowerShell

Warning

We suggest if you are in a Windows environment, you use the pre-built installers. For install instructions on the pre-built installers, see [the section called "Install/Update"](#)

Since Windows does not come with a POSIX-compliant shell, you need to install additional software to install the AWS CLI from source. [MSYS2](#) provides a collection of tools and libraries to help build and install Windows software, especially for the POSIX-based scripting that Autotools uses.

1. Install MSYS2. For information on installing and using MSYS2, see the [install and usage instructions](#) in the *MSYS2 Documentation*.
2. Open the MSYS2 terminal and install autotools using the following command.

```
$ pacman -S autotools
```

Note

When using the `configure`, `build`, and `install` code examples in this guide for Windows, the default MSYS2 install path of `C:\msys64\usr\bin\bash` is assumed. When calling MSYS2 inside of PowerShell you'll be using the following format, with the `bash` command in quotes:

```
PS C:\> C:\msys64\usr\bin\bash -lc "command example"
```

The following command example calls the `./configure` command.

```
PS C:\> C:\msys64\usr\bin\bash -lc "./configure"
```

3. A Python 3.8 or later interpreter is installed. The minimum Python version required follows the same timelines as the official [Python support policy for AWS SDKs and Tools](#). An interpreter is only supported 6 months after its end-of-support date.
4. **(Optional)** Install all build and runtime Python library dependencies of the AWS CLI. The `./configure` command informs you if you are missing any dependencies and how to install them.

You can automatically install and use these dependencies through configuration, see [the section called "Downloading dependencies"](#) for more information.

Step 2: Configuring the AWS CLI source installation

Configuration for building and installing the AWS CLI is specified using the `configure` script. For the documentation of all configuration options, run the `configure` script with the `--help` option:

Linux and macOS

```
$ ./configure --help
```

Windows PowerShell

```
PS C:\> C:\msys64\usr\bin\bash -lc "./configure --help"
```

The most important options are the following:

- [Install location](#)
- [Python interpreter](#)
- [Downloading dependencies](#)
- [Install type](#)

Install location

The source installation of the AWS CLI uses two configurable directories to install the AWS CLI:

- `libdir` - Parent directory where the AWS CLI will be installed. The path to the AWS CLI installation is `<libdir-value>/aws-cli`. The default `libdir` value for Linux and macOS is `/usr/local/lib` making the default installation directory `/usr/local/lib/aws-cli`
- `bindir` - Directory where the AWS CLI executables are installed. The default location is `/usr/local/bin`.

The following configure options control the directories used:

- `--prefix` - Sets the directory prefix to use for the installation. The default value for Linux and macOS is `/usr/local`.
- `--libdir` - Sets the `libdir` to use for installing the AWS CLI. The default value is `<prefix-value>/lib`. If both `--libdir` and `--prefix` are not specified, the default for Linux and macOS is `/usr/local/lib/`.
- `--bindir` - Sets the `bindir` to use for installing the AWS CLI `aws` and `aws_completer` executables. The default value is `<prefix-value>/bin`. If both `bindir` and `--prefix` are not specified, the default for Linux and macOS is `/usr/local/bin/`.

Linux and macOS

The following command example uses the `--prefix` option to do a local user install of the AWS CLI. This command installs the AWS CLI in `$HOME/.local/lib/aws-cli` and the executables in `$HOME/.local/bin`:

```
$ ./configure --prefix=$HOME/.local
```


The following command example uses the `--libdir` option to install the AWS CLI as an add-on application in the `/opt` directory. This command installs the AWS CLI at `/opt/aws-cli` and the executables at their default location of `/usr/local/bin`.

```
$ ./configure --libdir=/opt
```

Windows PowerShell

The following command example uses the `--prefix` option to do a local user install of the AWS CLI. This command installs the AWS CLI in `$HOME/.local/lib/aws-cli` and the executables in `$HOME/.local/bin`:

```
$ C:\msys64\usr\bin\bash -lc "./configure --prefix='C:\Program Files\AWSCLI'"
```

The following command example uses the `--libdir` option to install the AWS CLI as an add-on application in the `/opt` directory. This command installs the AWS CLI at `C:\Program Files\AWSCLI\opt\aws-cli`.

Python interpreter

Note

It is highly recommended to specify the Python interpreter when installing for Windows.

The `./configure` script automatically selects an installed Python 3.8 or later interpreter to use in building and running the AWS CLI using the [AM_PATH_PYTHON](#) Autoconf macro.

The Python interpreter to use can be explicitly set using the `PYTHON` environment variable when running the `configure` script:

Linux and macOS

```
$ PYTHON=/path/to/python ./configure
```

Windows PowerShell

```
PS C:\> C:\msys64\usr\bin\bash -lc "PYTHON='C:\path\to\python' ./configure"
```

Downloading dependencies

By default, it is required that all build and runtime dependencies of the AWS CLI are already installed on the system. This includes any Python library dependencies. All dependencies are checked when the configure script is run, and if the system is missing any Python dependencies, the configure script errors out.

The following code example errors out when your system is missing dependencies:

Linux and macOS

```
$ ./configure
checking for a Python interpreter with version >= 3.8... python
checking for python... /Users/username/.envs/env3.11/bin/python
checking for python version... 3.11
checking for python platform... darwin
checking for GNU default python prefix... ${prefix}
checking for GNU default python exec_prefix... ${exec_prefix}
checking for python script directory (pythondir)... ${PYTHON_PREFIX}/lib/python3.11/
site-packages
checking for python extension module directory (pyexecdir)... ${PYTHON_EXEC_PREFIX}/
lib/python3.11/site-packages
checking for --with-install-type... system-sandbox
checking for --with-download-deps... Traceback (most recent call last):
  File "<frozen runpy>", line 198, in _run_module_as_main
  File "<frozen runpy>", line 88, in _run_code
  File "/Users/username/aws-code/aws-cli/./backends/build_system/__main__.py", line
125, in <module>
    main()
  File "/Users/username/aws-code/aws-cli/./backends/build_system/__main__.py", line
121, in main
    parsed_args.func(parsed_args)
  File "/Users/username/aws-code/aws-cli/./backends/build_system/__main__.py", line
49, in validate
    validate_env(parsed_args.artifact)
  File "/Users/username/aws-code/aws-cli/./backends/build_system/validate_env.py",
line 68, in validate_env
    raise UnmetDependenciesException(unmet_deps, in_venv)
validate_env.UnmetDependenciesException: Environment requires following Python
dependencies:

awsCRT (required: ('>=0.12.4', '<0.17.0')) (version installed: None)
```

We recommend using `--with-download-deps` flag to automatically create a `virtualenv` and download the dependencies.

If you want to manage the dependencies yourself instead, run the following `pip` command:

```
/Users/username/.envs/env3.11/bin/python -m pip install --prefer-binary
'awscli>=0.12.4,<0.17.0'
```

```
configure: error: "Python dependencies not met."
```

Windows PowerShell

```
PS C:\> C:\msys64\usr\bin\bash -lc "./configure"
checking for a Python interpreter with version >= 3.8... python
checking for python... /Users/username/.envs/env3.11/bin/python
checking for python version... 3.11
checking for python platform... darwin
checking for GNU default python prefix... ${prefix}
checking for GNU default python exec_prefix... ${exec_prefix}
checking for python script directory (pythondir)... ${PYTHON_PREFIX}/lib/python3.11/
site-packages
checking for python extension module directory (pyexecdir)... ${PYTHON_EXEC_PREFIX}/
lib/python3.11/site-packages
checking for --with-install-type... system-sandbox
checking for --with-download-deps... Traceback (most recent call last):
  File "<frozen runpy>", line 198, in _run_module_as_main
  File "<frozen runpy>", line 88, in _run_code
  File "/Users/username/aws-code/aws-cli/./backends/build_system/__main__.py", line
125, in <module>
    main()
  File "/Users/username/aws-code/aws-cli/./backends/build_system/__main__.py", line
121, in main
    parsed_args.func(parsed_args)
  File "/Users/username/aws-code/aws-cli/./backends/build_system/__main__.py", line
49, in validate
    validate_env(parsed_args.artifact)
  File "/Users/username/aws-code/aws-cli/./backends/build_system/validate_env.py",
line 68, in validate_env
    raise UnmetDependenciesException(unmet_deps, in_venv)
validate_env.UnmetDependenciesException: Environment requires following Python
dependencies:

awscli (required: ('>=0.12.4', '<0.17.0')) (version installed: None)
```

```
We recommend using --with-download-deps flag to automatically create a virtualenv
and download the dependencies.
```

```
If you want to manage the dependencies yourself instead, run the following pip
command:
```

```
/Users/username/.envs/env3.11/bin/python -m pip install --prefer-binary
'awscli>=0.12.4,<0.17.0'
```

```
configure: error: "Python dependencies not met."
```

To automatically install the required Python dependencies, use the `--with-download-deps` option. When using this flag, the build process does the following:

- Skips the Python library dependencies check.
- Configures the settings to download all required Python dependencies and use **only** the downloaded dependencies to build the AWS CLI during the make build.

The following configure command example uses the `--with-download-deps` option to download and use the Python dependencies:

Linux and macOS

```
$ ./configure --with-download-deps
```

Windows PowerShell

```
PS C:\> C:\msys64\usr\bin\bash -lc "./configure --with-download-deps"
```

Install type

The source install process supports the following installation types:

- `system-sandbox` - **(Default)** Creates an isolated Python virtual environment, installs the AWS CLI into the virtual environment, and symlinks to the `aws` and `aws_completer` executable in the virtual environment. This install of the AWS CLI depends directly on the selected Python interpreter for its runtime.

This is a lightweight install mechanism to get the AWS CLI installed on a system and follows best Python practices by sandboxing the installation in a virtual environment. This installation is intended for customers that want to install the AWS CLI from source in the most frictionless way possible with the installation coupled to your installation of Python.

- `portable-exe` - Freezes the AWS CLI into a standalone executable that can be distributed to environments of similar architectures. This is the same process used to generate the official pre-built executables of the AWS CLI. The `portable-exe` freezes in a copy of the Python interpreter chosen in the `configure` step to use for the runtime of the AWS CLI. This allows it to be moved to other machines that may not have a Python interpreter.

This type of builds is useful because you can ensure your AWS CLI installation isn't coupled to the environment's installed Python version and you can distribute a build to other system that may not already have Python installed. This enables you to control the dependencies and security on the AWS CLI executables you use.

To configure the installation type, use the `--with-install-type` option and specify a value of `portable-exe` or `system-sandbox`.

The following `./configure` command example specifies a value of `portable-exe`:

Linux and macOS

```
$ ./configure --with-install-type=portable-exe
```

Windows PowerShell

```
PS C:\> C:\msys64\usr\bin\bash -lc "./configure --with-install-type=portable-exe"
```

Step 3: Building the AWS CLI

Use the `make` command to build the AWS CLI using your configuration settings:

Linux and macOS

```
$ make
```

Windows PowerShell

```
PS C:\> C:\msys64\usr\bin\bash -lc "make"
```

Note

When using the make command, the following steps are completed behind the scenes:

1. A virtual environment is created in the build directory using the Python [venv](#) module. The virtual environment is bootstrapped with a [version of pip that is vendored in the Python standard library](#).
2. Copies Python library dependencies. Depending on if the `--with-download-deps` flag is specified in the `configure` command, this step does one of the following:
 - The `--with-download-deps` **is** specified. Python dependencies are pip installed. This includes `wheel`, `setuptools`, and all AWS CLI runtime dependencies. If you are building the `portable-exe`, `pyinstaller` is installed. These requirements are all specified in lock files generated from [pip-compile](#).
 - The `--with-download-deps` **is not** specified. Python libraries from the Python interpreter's site package plus any scripts (e.g. `pyinstaller`) are copied into the virtual environment being used for the build.
3. Runs `pip install` directly on the AWS CLI codebase to do an offline, in-tree build and install of the AWS CLI into the build virtual environment. This install uses the pip flags [--no-build-isolation](#), [--use-feature=in-tree-build](#), [--no-cache-dir](#), and [--no-index](#).
4. **(Optional)** If the `--install-type` is set to `portable-exe` in the `configure` command, builds a standalone executable using [pyinstaller](#).

Step 4: Installing the AWS CLI

The `make install` command installs your built AWS CLI to the configured location on the system.

Linux and macOS

The following command example installs the AWS CLI using your configuration and build settings:

```
$ make install
```

Windows PowerShell

The following command example installs the AWS CLI using your configuration and build settings, then adds an environment variable with the path for the AWS CLI:

```
PS C:\> C:\msys64\usr\bin\bash -lc " make install "  
PS C:\> $Env: PATH +=";C:\Program Files\AWSCLI\bin\"
```

The `make install` rule supports the [DESTDIR](#) variable. When specified, this variable prefixes the specified path to the already configured installation path when installing the AWS CLI. By default, no value is set for this variable.

Linux and macOS

The following code example uses a `--prefix=/usr/local` flag for configuring an install location, and then alters that destination using `DESTDIR=/tmp/stage` for the `make install` command. These commands result in the AWS CLI being installed at `/tmp/stage/usr/local/lib/aws-cli` and its executables located in `/tmp/stage/usr/local/bin`.

```
$ ./configure --prefix=/usr/local  
$ make  
$ make DESTDIR=/tmp/stage install
```

Windows PowerShell

The following code example uses a `--prefix=\awscli` flag for configuring an install location, and then alters that destination using `DESTDIR=C:\Program Files` for the `make install` command. These commands result in the AWS CLI being installed at `C:\Program Files\awscli`.

```
$ ./configure --prefix=\awscli  
$ make
```

```
$ make DESTDIR='C:\Program Files' install
```

Note

When running `make install`, the following steps are completed behind the scenes

1. Moves one of the following to the configured install directory:
 - If the install type is `system-sandbox`, moves your built virtual environment.
 - If the install type is `portable-exe`, moves your built standalone executable.
2. Creates symlinks for both the `aws` and `aws_completer` executables in your configured `bin` directory.

Step 5: Verifying the AWS CLI installation

Confirm the AWS CLI successfully installed by using the following command:

```
$ aws --version  
aws-cli/2.15.30 Python/3.11.6 Windows/10 exe/AMD64 prompt/off
```

If the `aws` command is not recognized, you may need to restart your terminal for new symlinks to update. If you come across additional issues after installing or uninstalling the AWS CLI, see [Troubleshoot errors](#) for common troubleshooting steps

Workflow examples

This section provides some basic workflow examples for installing from source.

Basic Linux and macOS install

The following example is a basic installation workflow where the AWS CLI is installed in the default location of `/usr/local/lib/aws-cli`.

```
$ cd path/to/cli/respository/  
$ ./configure  
$ make  
$ make install
```


Automated Windows install

Note

You must run PowerShell as an Administrator to use this workflow.

MSYS2 can be used in an automated fashion in a CI setting, see [Using MSYS2 in CI](#) in the *MSYS2 Documentation*.

Downloaded Tarball

Download the `awscli.tar.gz` file, extract, and install the AWS CLI. When using the following commands, replace the following paths:

- `C:\msys64\usr\bin\bash` with the location of your MSYS2 path.
- `.\awscli-2.x.x\` with the extracted `awscli.tar.gz` folder name.
- `PYTHON='C:\path\to\python.exe'` with your local Python path.

The following code example automates building and installing the AWS CLI from PowerShell using MSYS2, and specifies which local install of Python to use:

```
PS C:\> curl -o awscli.tar.gz https://awscli.amazonaws.com/awscli.tar.gz #
  Download the awscli.tar.gz file in the current working directory
PS C:\> tar -xvzf .\awscli.tar.gz # Extract awscli.tar.gz file
PS C:\> cd .\awscli-2.x.x\ # Navigate to the root of the extracted files
PS C:\> $env:CHERE_INVOKING = 'yes' # Preserve the current working directory
PS C:\> C:\msys64\usr\bin\bash -lc " PYTHON='C:\path\to\python.exe' ./configure --
prefix='C:\Program Files\AWSCLI' --with-download-deps "
PS C:\> C:\msys64\usr\bin\bash -lc "make"
PS C:\> C:\msys64\usr\bin\bash -lc "make install"
PS C:\> $Env:PATH +=";C:\Program Files\AWSCLI\bin\"
PS C:\> aws --version
aws-cli/2.15.30 Python/3.11.6 Windows/10 source-sandbox/AMD64 prompt/off
```

GitHub Repository

Download the `awscli.tar.gz` file, extract, and install the AWS CLI. When using the following commands, replace the following paths:

- C:\msys64\usr\bin\bash with the location of your MSYS2 path.
- C:\path\to\cli\repository\ with the path to your cloned [AWS CLI repository](#) from *GitHub*. For more information, see [Fork a repo](#) in the *GitHub Docs*
- PYTHON='C:\path\to\python.exe' with your local Python path.

The following code example automates building and installing the AWS CLI from PowerShell using MSYS2, and specifies which local install of Python to use:

```
PS C:\> cd C:\path\to\cli\repository\
PS C:\> $env:CHERE_INVOKING = 'yes' # Preserve the current working directory
PS C:\> C:\msys64\usr\bin\bash -lc " PYTHON='C:\path\to\python.exe' ./configure --
prefix='C:\Program Files\AWSCLI' --with-download-deps "
PS C:\> C:\msys64\usr\bin\bash -lc "make"
PS C:\> C:\msys64\usr\bin\bash -lc "make install"
PS C:\> $Env:PATH +=";C:\Program Files\AWSCLI\bin\"
PS C:\> aws --version
```

Alpine Linux container

Below is an example Dockerfile that can be used to get a working installation of the AWS CLI in an Alpine Linux container as an [alternative to pre-built binaries for Alpine](#). When using this example, replace *AWSCLI_VERSION* with you desired AWS CLI version number:

```
FROM python:3.8-alpine AS builder

ENV AWSCLI_VERSION=2.10.1

RUN apk add --no-cache \
    curl \
    make \
    cmake \
    gcc \
    g++ \
    libc-dev \
    libffi-dev \
    openssl-dev \
    && curl https://awscli.amazonaws.com/awscli-${AWSCLI_VERSION}.tar.gz | tar -xz \
    && cd awscli-${AWSCLI_VERSION} \
    && ./configure --prefix=/opt/aws-cli/ --with-download-deps \
```

```
&& make \  
&& make install  
  
FROM python:3.8-alpine  
  
RUN apk --no-cache add groff  
  
COPY --from=builder /opt/aws-cli/ /opt/aws-cli/  
  
ENTRYPOINT ["/opt/aws-cli/bin/aws"]
```

This image is built and the AWS CLI invoked from a container similar to the one that is built on Amazon Linux 2:

```
$ docker build --tag awscli-alpine .  
$ docker run --rm -it awscli-alpine --version  
aws-cli/2.2.1 Python/3.8.11 Linux/5.10.25-linuxkit source-sandbox/x86_64.alpine.3  
prompt/off
```

The final size of this image is smaller than the size of the official AWS CLI Docker image. For information on the official Docker image, see [the section called “Amazon ECR Public/Docker”](#).

Troubleshooting AWS CLI install and uninstall errors

For troubleshooting steps for install errors, see [Troubleshoot errors](#) for common troubleshooting steps. For the most relevant troubleshooting steps, see [the section called “Command not found errors”](#), [the section called “The “aws --version” command returns a different version than you installed”](#), and [the section called “The “aws --version” command returns a version after uninstalling the AWS CLI”](#).

For any issues not covered in the troubleshooting guides, search the issues with the `source-distribution` label in the [AWS CLI Repository](#) on *GitHub*. If no existing issues cover your errors, [create a new issue](#) to receive help from the AWS CLI maintainers.

Next steps

After installing the AWS CLI, you should perform a [the section called “Setup”](#).

Run the AWS CLI from the official Amazon ECR Public or Docker images

This topic describes how to run, version control, and configure the AWS CLI version 2 on Docker using either the official Amazon Elastic Container Registry Public (Amazon ECR Public) or Docker Hub image. For more information on how to use Docker, see [Docker's documentation](#).

Official images provide isolation, portability, and security that AWS directly supports and maintains. This enables you to use the AWS CLI version 2 in a container-based environment without having to manage the installation yourself.

Topics

- [Prerequisites](#)
- [Deciding between Amazon ECR Public and Docker Hub](#)
- [Run the official AWS CLI version 2 images](#)
- [Notes on interfaces and backwards compatibility of the official images](#)
- [Use specific versions and tags](#)
- [Update to the latest official image](#)
- [Share host files, credentials, environment variables, and configuration](#)
- [Shorten the docker run command](#)

Prerequisites

You must have Docker installed. For installation instructions, see the [Docker website](#).

To verify your installation of Docker, run the following command and confirm there is an output.

```
$ docker --version
Docker version 19.03.1
```

Deciding between Amazon ECR Public and Docker Hub

We recommend using Amazon ECR Public over Docker Hub for AWS CLI images. Docker Hub has stricter rate limiting for public consumers which can cause throttling issues. In addition, Amazon ECR Public replicates images in more than one region to provide strong availability and handle region outage issues.

For more information on Docker Hub rate limiting see [Understanding Docker Hub Rate Limiting](#) on the *Docker* website.

Run the official AWS CLI version 2 images

The first time you use the `docker run` command, the latest image is downloaded to your computer. Each subsequent use of the `docker run` command runs from your local copy.

To run the AWS CLI version 2 Docker images, use the `docker run` command.

Amazon ECR Public

The official AWS CLI version 2 Amazon ECR Public image is hosted on Amazon ECR Public in the [aws-cli/aws-cli repository](#).

```
$ docker run --rm -it public.ecr.aws/aws-cli/aws-cli command
```

Docker Hub

The official AWS CLI version 2 Docker image is hosted on Docker Hub in the `amazon/aws-cli` repository.

```
$ docker run --rm -it amazon/aws-cli command
```

This is how the command functions:

- `docker run --rm -it repository/name` – The equivalent of the `aws` executable. Each time you run this command, Docker spins up a container of your downloaded image, and executes your `aws` command. By default, the image uses the latest version of the AWS CLI version 2.

For example, to call the `aws --version` command in Docker, you run the following.

Amazon ECR Public

```
$ docker run --rm -it public.ecr.aws/aws-cli/aws-cli --version  
aws-cli/2.15.30 Python/3.7.3 Linux/4.9.184-linuxkit botocore/2.4.5dev10
```

Docker Hub

```
$ docker run --rm -it amazon/aws-cli --version
```

```
aws-cli/2.15.30 Python/3.7.3 Linux/4.9.184-linuxkit botocore/2.4.5dev10
```

- `--rm` – Specifies to clean up the container after the command exits.
- `-it` – Specifies to open a pseudo-TTY with `stdin`. This enables you to provide input to the AWS CLI version 2 while it's running in a container, for example, by using the `aws configure` and `aws help` commands. When choosing whether to omit `-it`, consider the following:
 - If you are running scripts, `-it` is not needed.
 - If you are experiencing errors with your scripts, omitting `-it` from your Docker call might fix the issue.
 - If you are trying to pipe output, `-it` might cause errors and omitting `-it` from your Docker call might resolve this issue. If you'd like to keep the `-it` flag, but still would like to pipe output, disabling the [client-side pager](#) the AWS CLI uses by default should resolve the issue.

For more information about the `docker run` command, see the [Docker reference guide](#).

Notes on interfaces and backwards compatibility of the official images

- The only tool supported on the image is the AWS CLI. Only the `aws` executable should ever be directly run. For example, even though `less` and `groff` are explicitly installed on the image, they should not be executed directly outside of an AWS CLI command.
- The `/aws` working directory is user controlled. The image will not write to this directory, unless instructed by the user in running an AWS CLI command.
- There are no backwards compatibility guarantees in relying on the latest tag. To guarantee backwards compatibility, you must pin to a specific `<major.minor.patch>` tag as those tags are immutable; they will only ever be pushed to once.

Use specific versions and tags

The official AWS CLI version 2 image has multiple versions you can use, starting with version `2.0.6`. To run a specific version of the AWS CLI version 2, append the appropriate tag to your `docker run` command. The first time you use the `docker run` command with a tag, the latest image for that tag is downloaded to your computer. Each subsequent use of the `docker run` command with that tag runs from your local copy.

You can use two types of tags:

- `latest` – Defines the latest version of the AWS CLI version 2 for the image. We recommend you use the `latest` tag when you want the latest version of the AWS CLI version 2. However, there are no backward-compatibility guarantees when relying on this tag. The `latest` tag is used by default in the `docker run` command. To explicitly use the `latest` tag, append the tag to the container image name.

Amazon ECR Public

```
$ docker run --rm -it public.ecr.aws/aws-cli/aws-cli:latest command
```

Docker Hub

```
$ docker run --rm -it amazon/aws-cli:latest command
```

- `<major.minor.patch>` – Defines a specific version of the AWS CLI version 2 for the image. If you plan to use an official image in production, we recommend you use a specific version of the AWS CLI version 2 to ensure backward compatibility. For example, to run version `2.0.6`, append the version to the container image name.

Amazon ECR Public

```
$ docker run --rm -it public.ecr.aws/aws-cli/aws-cli:2.0.6 command
```

Docker Hub

```
$ docker run --rm -it amazon/aws-cli:2.0.6 command
```

Update to the latest official image

Because the latest image is downloaded to your computer only the first time you use the `docker run` command, you need to manually pull an updated image. To manually update to the latest version, we recommend you pull the `latest` tagged image. Pulling the image downloads the latest version to your computer.

Amazon ECR Public

```
$ docker pull public.ecr.aws/aws-cli/aws-cli:latest
```

Docker Hub

```
$ docker pull amazon/aws-cli:latest
```

Share host files, credentials, environment variables, and configuration

Because the AWS CLI version 2 is run in a container, by default the CLI can't access the host file system, which includes configuration and credentials. To share the host file system, credentials, and configuration to the container, mount the host system's `~/ .aws` directory to the container at `/root/ .aws` with the `-v` flag to the `docker run` command. This allows the AWS CLI version 2 running in the container to locate host file information.

Amazon ECR Public

Linux and macOS

```
$ docker run --rm -it -v ~/.aws:/root/.aws public.ecr.aws/aws-cli/aws-cli command
```

Windows Command Prompt

```
$ docker run --rm -it -v %userprofile%\ .aws:/root/.aws public.ecr.aws/aws-cli/aws-cli command
```

Windows PowerShell

```
C:\> docker run --rm -it -v $env:userprofile\ .aws:/root/.aws public.ecr.aws/aws-cli/aws-cli command
```

Docker Hub

Linux and macOS

```
$ docker run --rm -it -v ~/.aws:/root/.aws amazon/aws-cli command
```

Windows Command Prompt

```
$ docker run --rm -it -v %userprofile%\ .aws:/root/.aws amazon/aws-cli command
```

Windows PowerShell


```
C:\> docker run --rm -it -v $env:userprofile\.aws:/root/.aws amazon/aws-cli command
```

For more information about the `-v` flag and mounting, see the [Docker reference guide](#).

Note

For information on config and credentials files, see [the section called “Configuration and credential file settings”](#).

Example 1: Providing credentials and configuration

In this example, we're providing host credentials and configuration when running the `s3 ls` command to list your buckets in Amazon Simple Storage Service (Amazon S3). The below examples use the default location for AWS CLI credentials and configuration files, to use a different location, change the file path.

Amazon ECR Public

Linux and macOS

```
$ docker run --rm -it -v ~/.aws:/root/.aws public.ecr.aws/aws-cli/aws-cli s3 ls
2020-03-25 00:30:48 aws-cli-docker-demo
```

Windows Command Prompt

```
$ docker run --rm -it -v %userprofile%.aws:/root/.aws public.ecr.aws/aws-cli/aws-
cli s3 ls
2020-03-25 00:30:48 aws-cli-docker-demo
```

Windows PowerShell

```
C:\> docker run --rm -it -v $env:userprofile\.aws:/root/.aws public.ecr.aws/aws-cli/
aws-cli s3 ls
```

Docker Hub

Linux and macOS

```
$ docker run --rm -it -v ~/.aws:/root/.aws amazon/aws-cli s3 ls
2020-03-25 00:30:48 aws-cli-docker-demo
```

Windows Command Prompt

```
$ docker run --rm -it -v %userprofile%\aws:/root/.aws amazon/aws-cli s3 ls
2020-03-25 00:30:48 aws-cli-docker-demo
```

Windows PowerShell

```
C:\> docker run --rm -it -v $env:userprofile\aws:/root/.aws amazon/aws-cli s3 ls
```

You can call specific system's environment variables using the `-e` flag. To use an environment variable, call it by name.

Amazon ECR Public

Linux and macOS

```
$ docker run --rm -it -v ~/.aws:/root/.aws -e ENVVAR_NAME public.ecr.aws/aws-cli/
aws-cli s3 ls
2020-03-25 00:30:48 aws-cli-docker-demo
```

Windows Command Prompt

```
$ docker run --rm -it -v %userprofile%\aws:/root/.aws -e ENVVAR_NAME
public.ecr.aws/aws-cli/aws-cli s3 ls
2020-03-25 00:30:48 aws-cli-docker-demo
```

Windows PowerShell

```
C:\> docker run --rm -it -v $env:userprofile\aws:/root/.aws -e ENVVAR_NAME
public.ecr.aws/aws-cli/aws-cli s3 ls
```

Docker Hub

Linux and macOS

```
$ docker run --rm -it -v ~/.aws:/root/.aws -e ENVVAR_NAME amazon/aws-cli s3 ls
```

```
2020-03-25 00:30:48 aws-cli-docker-demo
```

Windows Command Prompt

```
$ docker run --rm -it -v %userprofile%\aws:/root/.aws -e ENVVAR_NAME amazon/aws-cli  
s3 ls  
2020-03-25 00:30:48 aws-cli-docker-demo
```

Windows PowerShell

```
C:\> docker run --rm -it -v $env:userprofile\aws:/root/.aws -e ENVVAR_NAME amazon/  
aws-cli s3 ls
```

Example 2: Downloading an Amazon S3 file to your host system

For some AWS CLI version 2 commands, you can read files from the host system in the container or write files from the container to the host system.

In this example, we download the S3 object `s3://aws-cli-docker-demo/hello` to your local file system by mounting the current working directory to the container's `/aws` directory. By downloading the `hello` object to the container's `/aws` directory, the file is saved to the host system's current working directory also.

Amazon ECR Public

Linux and macOS

```
$ docker run --rm -it -v ~/.aws:/root/.aws -v $(pwd):/aws public.ecr.aws/aws-cli/  
aws-cli s3 cp s3://aws-cli-docker-demo/hello .  
download: s3://aws-cli-docker-demo/hello to ./hello
```

Windows Command Prompt

```
$ docker run --rm -it -v %userprofile%\aws:/root/.aws -v %cd%:/aws public.ecr.aws/  
aws-cli/aws-cli s3 cp s3://aws-cli-docker-demo/hello .  
download: s3://aws-cli-docker-demo/hello to ./hello
```

Windows PowerShell

```
C:\> docker run --rm -it -v $env:userprofile\.aws:/root/.aws -v $pwd\aws:/aws
public.ecr.aws/aws-cli/aws-cli s3 cp s3://aws-cli-docker-demo/hello .
```

Docker Hub

Linux and macOS

```
$ docker run --rm -it -v ~/.aws:/root/.aws -v $(pwd):/aws amazon/aws-cli s3 cp s3://
aws-cli-docker-demo/hello .
download: s3://aws-cli-docker-demo/hello to ./hello
```

Windows Command Prompt

```
$ docker run --rm -it -v %userprofile%.aws:/root/.aws -v %cd%:/aws amazon/aws-cli
s3 cp s3://aws-cli-docker-demo/hello .
download: s3://aws-cli-docker-demo/hello to ./hello
```

Windows PowerShell

```
C:\> docker run --rm -it -v $env:userprofile\.aws:/root/.aws -v $pwd\aws:/aws
amazon/aws-cli s3 cp s3://aws-cli-docker-demo/hello .
```

To confirm the downloaded file exists in the local file system, run the following.

Linux and macOS

```
$ cat hello
Hello from Docker!
```

Windows PowerShell

```
$ type hello
Hello from Docker!
```

Example 3: Using your `AWS_PROFILE` environment variable

You can call specific system's environment variables using the `-e` flag. Call each environment variable you'd like to use. In this example, we're providing host credentials, configuration, and the

AWS_PROFILE environment variable when running the `s3 ls` command to list your buckets in Amazon Simple Storage Service (Amazon S3).

Amazon ECR Public

Linux and macOS

```
$ docker run --rm -it -v ~/.aws:/root/.aws -e AWS_PROFILE public.ecr.aws/aws-cli/  
aws-cli s3 ls  
2020-03-25 00:30:48 aws-cli-docker-demo
```

Windows Command Prompt

```
$ docker run --rm -it -v %userprofile%\aws:/root/.aws -e AWS_PROFILE  
public.ecr.aws/aws-cli/aws-cli s3 ls  
2020-03-25 00:30:48 aws-cli-docker-demo
```

Windows PowerShell

```
C:\> docker run --rm -it -v $env:userprofile\aws:/root/.aws -e AWS_PROFILE  
public.ecr.aws/aws-cli/aws-cli s3 ls
```

Docker Hub

Linux and macOS

```
$ docker run --rm -it -v ~/.aws:/root/.aws -e AWS_PROFILE amazon/aws-cli s3 ls  
2020-03-25 00:30:48 aws-cli-docker-demo
```

Windows Command Prompt

```
$ docker run --rm -it -v %userprofile%\aws:/root/.aws -e AWS_PROFILE amazon/aws-cli  
s3 ls  
2020-03-25 00:30:48 aws-cli-docker-demo
```

Windows PowerShell

```
C:\> docker run --rm -it -v $env:userprofile\aws:/root/.aws -e AWS_PROFILE amazon/  
aws-cli s3 ls
```

Shorten the docker run command

To shorten the `docker run` command, we suggest you use your operating system's ability to create a [symbolic link](#) (symlink) or [alias](#) in Linux and macOS, or [doskey](#) in Windows. To set the `aws` alias, you can run one of the following commands.

- For basic access to `aws` commands, run the following.

Amazon ECR Public

Linux and macOS

```
$ alias aws='docker run --rm -it public.ecr.aws/aws-cli/aws-cli'
```

Windows Command Prompt

```
C:\> doskey aws=docker run --rm -it public.ecr.aws/aws-cli/aws-cli $*
```

Windows PowerShell

```
C:\> Function AWSCLI {docker run --rm -it public.ecr.aws/aws-cli/aws-cli $args}  
Set-Alias -Name aws -Value AWSCLI
```

Docker Hub

Linux and macOS

```
$ alias aws='docker run --rm -it amazon/aws-cli'
```

Windows Command Prompt

```
C:\> doskey aws=docker run --rm -it amazon/aws-cli $*
```

Windows PowerShell

```
C:\> Function AWSCLI {docker run --rm -it amazon/aws-cli $args}  
Set-Alias -Name aws -Value AWSCLI
```

- For access to the host file system and configuration settings when using aws commands, run the following.

Amazon ECR Public

Linux and macOS

```
$ alias aws='docker run --rm -it -v ~/.aws:/root/.aws -v $(pwd):/aws public.ecr.aws/aws-cli/aws-cli'
```

Windows Command Prompt

```
C:\> doskey aws=docker run --rm -it -v %userprofile%.aws:/root/.aws -v %cd%:/aws public.ecr.aws/aws-cli/aws-cli $*
```

Windows PowerShell

```
C:\> Function AWSCLI {docker run --rm -it -v $env:userprofile\.aws:/root/.aws -v $pwd\aws:/aws public.ecr.aws/aws-cli/aws-cli $args}
Set-Alias -Name aws -Value AWSCLI
```

Docker Hub

Linux and macOS

```
$ alias aws='docker run --rm -it -v ~/.aws:/root/.aws -v $(pwd):/aws amazon/aws-cli'
```

Windows Command Prompt

```
C:\> doskey aws=docker run --rm -it -v %userprofile%.aws:/root/.aws -v %cd%:/aws amazon/aws-cli $*
```

Windows PowerShell

```
C:\> Function AWSCLI {docker run --rm -it -v $env:userprofile\.aws:/root/.aws -v $pwd\aws:/aws amazon/aws-cli $args}
Set-Alias -Name aws -Value AWSCLI
```

- To assign a specific version to use in your aws alias, append your version tag.

Amazon ECR Public

Linux and macOS

```
$ alias aws='docker run --rm -it -v ~/.aws:/root/.aws -v $(pwd):/aws  
public.ecr.aws/aws-cli/aws-cli:2.0.6'
```

Windows Command Prompt

```
C:\> doskey aws=docker run --rm -it -v %userprofile%\aws:/root/.aws -v %cd%:/aws  
public.ecr.aws/aws-cli/aws-cli:2.0.6 $*
```

Windows PowerShell

```
C:\> Function AWSCLI {docker run --rm -it -v $env:userprofile\aws:/root/.aws -v  
$pwd\aws:/aws public.ecr.aws/aws-cli/aws-cli:2.0.6 $args}  
Set-Alias -Name aws -Value AWSCLI
```

Docker Hub

Linux and macOS

```
$ alias aws='docker run --rm -it -v ~/.aws:/root/.aws -v $(pwd):/aws amazon/aws-  
cli:2.0.6'
```

Windows Command Prompt

```
C:\> doskey aws=docker run --rm -it -v %userprofile%\aws:/root/.aws -v %cd%:/aws  
amazon/aws-cli:2.0.6 $*
```

Windows PowerShell

```
C:\> Function AWSCLI {docker run --rm -it -v $env:userprofile\aws:/root/.aws -v  
$pwd\aws:/aws amazon/aws-cli:2.0.6 $args}  
Set-Alias -Name aws -Value AWSCLI
```

After setting your alias, you can run the AWS CLI version 2 from within a container as if it's installed on your host system.


```
$ aws --version
aws-cli/2.15.30 Python/3.7.3 Linux/4.9.184-linuxkit botocore/2.4.5dev10
```

Set up the AWS CLI

This topic explains how to quickly configure basic settings that the AWS Command Line Interface (AWS CLI) uses to interact with AWS. These include your security credentials, the default output format, and the default AWS Region.

Topics

- [Gather your credential information for programmatic access](#)
- [Setting up new configuration and credentials](#)
- [Using existing configuration and credentials files](#)

Gather your credential information for programmatic access

You'll need programmatic access if you want to interact with AWS outside of the AWS Management Console. For authentication and credential instructions, choose one of the following options:

Authentication type	Purpose	Instructions
IAM Identity Center workforce users short-term credentials	<p>(Recommended) Use short-term credentials for an IAM Identity Center workforce user.</p> <p>Security best practice is to use AWS Organizations with IAM Identity Center. It combines short-term credentials with a user directory, such as the built-in IAM Identity Center directory or Active Directory.</p>	the section called "IAM Identity Center authentication"
IAM user short-term credentials	Use IAM user short-term credentials, which are	the section called "Short-term credentials"

Authentication type	Purpose	Instructions
	more secure than long-term credentials. If your credentials are compromised, there is a limited time they can be used before they expire.	
IAM or IAM Identity Center users on an Amazon EC2 instance.	Use Amazon EC2 instance metadata to query for temporary credentials using the role assigned to the Amazon EC2 instance.	the section called "Use credentials for Amazon EC2 instance metadata"
Assume roles for permissions	Pair another credential method and assume a role for temporary access to AWS services your user might not have access to.	the section called "IAM roles"
IAM user long-term credentials	(Not recommended) Use long-term credentials, which have no expiration.	the section called "IAM users"
External storage of IAM or IAM Identity Center workforce users	(Not recommended) Pair another credential method but store credential values in a location outside of the AWS CLI. This method is only as secure as the external location the credentials are stored.	the section called "External credentials"

Setting up new configuration and credentials

The AWS CLI stores your configuration and credential information in a *profile* (a collection of settings) in the `credentials` and `config` files.

There are primarily two methods to quickly get setup:

- [Configuring using AWS CLI commands](#)
- [Manually editing the credentials and config files](#)

The following examples use sample values for each of the authentication methods. Replace sample values with your own.

Configuring using AWS CLI commands

For general use, the `aws configure` or `aws configure sso` commands in your preferred terminal are the fastest way to set up your AWS CLI installation. Based on the credential method you prefer, the AWS CLI prompts you for the relevant information. By default, the information in this profile is used when you run an AWS CLI command that doesn't explicitly specify a profile to use.

For more information on the credentials and config files, see [Configuration and credential file settings](#).

IAM Identity Center (SSO)

This example is for AWS IAM Identity Center using the `aws configure sso` wizard. For more information, see [the section called "Configure automatic token refresh"](#).

```
$ aws configure sso
SSO session name (Recommended): my-sso
SSO start URL [None]: https://my-sso-portal.awsapps.com/start
SSO region [None]:us-east-1

Attempting to automatically open the SSO authorization page in your default browser.

There are 2 AWS accounts available to you.
> DeveloperAccount, developer-account-admin@example.com (111122223333)
  ProductionAccount, production-account-admin@example.com (444455556666)

Using the account ID 111122223333

There are 2 roles available to you.
> ReadOnly
  FullAccess
```

```
Using the role name "ReadOnly"  
  
CLI default client Region [None]: us-west-2  
CLI default output format [None]: json  
CLI profile name [123456789011_ReadOnly]: user1
```

IAM Identity Center (Legacy SSO)

This example is for the legacy method of AWS IAM Identity Center using the `aws configure sso` wizard. To use the legacy SSO, leave the session name blank. For more information, see [the section called "Configure legacy non-refreshable"](#).

```
$ aws configure sso  
SSO session name (Recommended):  
SSO start URL [None]: https://my-sso-portal.awsapps.com/start  
SSO region [None]:us-east-1  
  
SSO authorization page has automatically been opened in your default browser.  
Follow the instructions in the browser to complete this authorization request.  
  
There are 2 AWS accounts available to you.  
> DeveloperAccount, developer-account-admin@example.com (111122223333)  
   ProductionAccount, production-account-admin@example.com (444455556666)  
  
Using the account ID 111122223333  
  
There are 2 roles available to you.  
> ReadOnly  
   FullAccess  
  
Using the role name "ReadOnly"  
  
CLI default client Region [None]: us-west-2  
CLI default output format [None]: json  
CLI profile name [123456789011_ReadOnly]: user1
```

Short-term credentials

This example is for the short-term credentials from AWS Identity and Access Management. The `aws configure` wizard is used to set initial values and then the `aws configure set` command assigns the last value needed. For more information, see [the section called "Short-term credentials"](#).

Long-term credentials

Warning

To avoid security risks, don't use IAM users for authentication when developing purpose-built software or working with real data. Instead, use federation with an identity provider such as [AWS IAM Identity Center](#).

This example is for the long-term credentials from AWS Identity and Access Management. For more information, see [the section called "IAM users"](#).

```
$ aws configure
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
Default region name [None]: us-west-2
Default output format [None]: json
```

For more detailed information on authentication and credential methods see [Authentication and access credentials](#).

Manually editing the credentials and config files

When copy and pasting information, we suggest manually editing the config and credentials file. Based on the credential method you prefer, the files are setup in a different way.

The files are stored in your home directory under the `.aws` folder. Where you find your home directory location varies based on the operating system, but is referred to using the environment variables `%UserProfile%` in Windows and `$HOME` or `~` (tilde) in Unix-based systems. For more information on where these settings are stored, see [the section called "Where are configuration settings stored?"](#).

The following examples show a default profile and a profile named `user1` and use sample values. Replace sample values with your own. For more information on the credentials and config files, see [Configuration and credential file settings](#).

IAM Identity Center (SSO)

This example is for AWS IAM Identity Center. For more information, see [the section called "Configure automatic token refresh"](#).

Credentials file

The `credentials` file is not used for this authentication method.

Config file

```
[default]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = readOnly
region = us-west-2
output = text

[profile user1]
sso_session = my-sso
sso_account_id = 444455556666
sso_role_name = readOnly
region = us-east-1
output = json

[sso-session my-sso]
sso_region = us-east-1
sso_start_url = https://my-sso-portal.awsapps.com/start
sso_registration_scopes = sso:account:access
```

IAM Identity Center (Legacy SSO)

This example is for the legacy method of AWS IAM Identity Center. For more information, see [the section called “Configure legacy non-refreshable”](#).

Credentials file

The `credentials` file is not used for this authentication method.

Config file

```
[default]
sso_start_url = https://my-sso-portal.awsapps.com/start
sso_region = us-east-1
sso_account_id = 111122223333
sso_role_name = readOnly
region = us-west-2
output = text
```



```
credential_source=Ec2InstanceMetadata
region=us-west-2
output=json

[profile user1]
role_arn=arn:aws:iam::777788889999:role/user1role
credential_source=Ec2InstanceMetadata
region=us-east-1
output=text
```

Long-term credentials

Warning

To avoid security risks, don't use IAM users for authentication when developing purpose-built software or working with real data. Instead, use federation with an identity provider such as [AWS IAM Identity Center](#).

This example is for the long-term credentials from AWS Identity and Access Management. For more information, see [the section called "IAM users"](#).

Credentials file

```
[default]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY

[user1]
aws_access_key_id=AKIAI44QH8DHBEXAMPLE
aws_secret_access_key=je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY
```

Config file

```
[default]
region=us-west-2
output=json

[profile user1]
region=us-east-1
output=text
```

For more detailed information on authentication and credential methods see [Authentication and access credentials](#).

Using existing configuration and credentials files

If you have existing configuration and credentials files, these can be used for the AWS CLI.

To use the `config` and `credentials` files, move them to the folder named `.aws` in your home directory. Where you find your home directory location varies based on the operating system, but is referred to using the environment variables `%UserProfile%` in Windows and `$HOME` or `~` (tilde) in Unix-based systems.

You can specify a non-default location for the `config` and `credentials` files by setting the `AWS_CONFIG_FILE` and `AWS_SHARED_CREDENTIALS_FILE` environment variables to another local path. See [Environment variables to configure the AWS CLI](#) for details.

For more detailed information on configuration and credentials files, see [the section called "Configuration and credential file settings"](#).

Configure the AWS CLI

This section explains how to configure the settings that the AWS Command Line Interface (AWS CLI) uses to interact with AWS. These include the following:

- **Credentials** identify who is calling the API. Access credentials are used to encrypt the request to the AWS servers to confirm your identity and retrieve associated permissions policies. These permissions determine the actions you can perform. For information on setting up your credentials, see [Authentication and access credentials](#).
- **Other configuration details** to tell the AWS CLI how to process requests, such as the default output format and the default AWS Region.

Note

AWS requires that all incoming requests are cryptographically signed. The AWS CLI does this for you. The "signature" includes a date/time stamp. Therefore, you must ensure that your computer's date and time are set correctly. If you don't, and the date/time in the signature is too far off of the date/time recognized by the AWS service, AWS rejects the request.

Configuration and credentials precedence

Credentials and configuration settings are located in multiple places, such as the system or user environment variables, local AWS configuration files, or explicitly declared on the command line as a parameter. Certain locations take precedence over others. The AWS CLI credentials and configuration settings take precedence in the following order:

1. **Command line options** – Overrides settings in any other location, such as the `--region`, `--output`, and `--profile` parameters.
2. **Environment variables** – You can store values in your system's environment variables.
3. **Assume role** – Assume the permissions of an IAM role through configuration or the `aws sts assume-role` command.
4. **Assume role with web identity** – Assume the permissions of an IAM role using web identity through configuration or the `aws sts assume-role` command.

5. [AWS IAM Identity Center](#) – The IAM Identity Center configuration settings stored in the config file are updated when you run the `aws configure sso` command. Credentials are then authenticated when you run the `aws sso login` command. The config file is located at `~/.aws/config` on Linux or macOS, or at `C:\Users\USERNAME\.aws\config` on Windows.
6. [Credentials file](#) – The credentials and config file are updated when you run the command `aws configure`. The credentials file is located at `~/.aws/credentials` on Linux or macOS, or at `C:\Users\USERNAME\.aws\credentials` on Windows.
7. [Custom process](#) – Get your credentials from an external source.
8. [Configuration file](#) – The credentials and config file are updated when you run the command `aws configure`. The config file is located at `~/.aws/config` on Linux or macOS, or at `C:\Users\USERNAME\.aws\config` on Windows.
9. [Container credentials](#) – You can associate an IAM role with each of your Amazon Elastic Container Service (Amazon ECS) task definitions. Temporary credentials for that role are then available to that task's containers. For more information, see [IAM Roles for Tasks](#) in the *Amazon Elastic Container Service Developer Guide*.
10. [Amazon EC2 instance profile credentials](#) – You can associate an IAM role with each of your Amazon Elastic Compute Cloud (Amazon EC2) instances. Temporary credentials for that role are then available to code running in the instance. The credentials are delivered through the Amazon EC2 metadata service. For more information, see [IAM Roles for Amazon EC2](#) in the *Amazon EC2 User Guide* and [Using Instance Profiles](#) in the *IAM User Guide*.

Additional topics in this section

- [the section called “Configuration and credential file settings”](#)
- [the section called “Environment Variables”](#)
- [the section called “Command line options”](#)
- [the section called “Command completion”](#)
- [the section called “Retries”](#)
- [the section called “Use an HTTP proxy”](#)

Configuration and credential file settings

You can save your frequently used configuration settings and credentials in files that are maintained by the AWS CLI.

The files are divided into `profiles`. By default, the AWS CLI uses the settings found in the profile named `default`. To use alternate settings, you can create and reference additional profiles.

You can override an individual setting by either setting one of the supported environment variables, or by using a command line parameter. For more information on configuration setting precedence, see [Configure the AWS CLI](#).

Note

For information on setting up your credentials, see [Authentication and access credentials](#).

Topics

- [Format of the configuration and credential files](#)
- [Where are configuration settings stored?](#)
- [Using named profiles](#)
- [Set and view configuration settings using commands](#)
- [Setting new configuration and credentials command examples](#)
- [Supported config file settings](#)

Format of the configuration and credential files

The `config` and `credentials` files are organized into sections. Sections include *profiles*, *ssosessions*, and *services*. A section is a named collection of settings, and continues until another section definition line is encountered. Multiple profiles and sections can be stored in the `config` and `credentials` files.

These files are plaintext files that use the following format:

- Section names are enclosed in brackets `[]` such as `[default]`, `[profile user1]`, and `[sso-session]`.
- All entries in a section take the general form of `setting_name=value`.

- Lines can be commented out by starting the line with a hash character (#).

The config and credentials files contain the following section types:

- [Section type: profile](#)
- [Section type: sso-session](#)
- [Section type: services](#)

Section type: profile

The AWS CLI stores

Depending on the file, profile section names use the following format:

- **Config file:** [default] [profile *user1*]
- **Credentials file:** [default] [*user1*]

Do **not** use the word profile when creating an entry in the credentials file.

Each profile can specify different credentials and can also specify different AWS Regions and output formats. When naming the profile in a config file, include the prefix word "profile", but do not include it in the credentials file.

The following examples show a credentials and config file with two profiles, region, and output specified. The first [*default*] is used when you run a AWS CLI command with no profile specified. The second is used when you run a AWS CLI command with the `--profile user1` parameter.

IAM Identity Center (SSO)

This example is for AWS IAM Identity Center. For more information, see [the section called "Configure automatic token refresh"](#).

Credentials file

The credentials file is not used for this authentication method.

Config file

```
[default]
```

```
sso_session = my-ss0
sso_account_id = 111122223333
sso_role_name = readOnly
region = us-west-2
output = text

[profile user1]
sso_session = my-ss0
sso_account_id = 444455556666
sso_role_name = readOnly
region = us-east-1
output = json

[sso-session my-ss0]
sso_region = us-east-1
sso_start_url = https://my-ss0-portal.awsapps.com/start
sso_registration_scopes = sso:account:access
```

IAM Identity Center (Legacy SSO)

This example is for the legacy method of AWS IAM Identity Center. For more information, see [the section called “Configure legacy non-refreshable”](#).

Credentials file

The credentials file is not used for this authentication method.

Config file

```
[default]
sso_start_url = https://my-ss0-portal.awsapps.com/start
sso_region = us-east-1
sso_account_id = 111122223333
sso_role_name = readOnly
region = us-west-2
output = text

[profile user1]
sso_start_url = https://my-ss0-portal.awsapps.com/start
sso_region = us-east-1
sso_account_id = 444455556666
sso_role_name = readOnly
region = us-east-1
```


Long-term credentials

Warning

To avoid security risks, don't use IAM users for authentication when developing purpose-built software or working with real data. Instead, use federation with an identity provider such as [AWS IAM Identity Center](#).

This example is for the long-term credentials from AWS Identity and Access Management. For more information, see [the section called "IAM users"](#).

Credentials file

```
[default]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY

[user1]
aws_access_key_id=AKIAI44QH8DHBEXAMPLE
aws_secret_access_key=je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY
```

Config file

```
[default]
region=us-west-2
output=json

[profile user1]
region=us-east-1
output=text
```

For more information and additional authorization and credential methods see, see [the section called "IAM users"](#).

Section type: sso-session

The `sso-session` section of the `config` file is used to group configuration variables for acquiring SSO access tokens, which can then be used to acquire AWS credentials. The following settings are used:

- **(Required)** [sso_start_url](#)
- **(Required)** [sso_region](#)
- [sso_account_id](#)
- [sso_role_name](#)
- [sso_registration_scopes](#)

You define an `sso-session` section and associate it to a profile. `sso_region` and `sso_start_url` must be set within the `sso-session` section. Typically, `sso_account_id` and `sso_role_name` must be set in the profile section so that the SDK can request SSO credentials.

The following example configures the SDK to request SSO credentials and supports automated token refresh:

```
[profile dev]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole

[sso-session my-sso]
sso_region = us-east-1
sso_start_url = https://my-sso-portal.awsapps.com/start
```

This also allows `sso-session` configurations to be reused across multiple profiles:

```
[profile dev]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole

[profile prod]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole2

[sso-session my-sso]
sso_region = us-east-1
sso_start_url = https://my-sso-portal.awsapps.com/start
```

However, `sso_account_id` and `sso_role_name` aren't required for all scenarios of SSO token configuration. If your application only uses AWS services that support bearer authentication, then traditional AWS credentials are not needed. Bearer authentication is an HTTP authentication scheme that uses security tokens called bearer tokens. In this scenario, `sso_account_id` and `sso_role_name` aren't required. See the individual guide for your AWS service to determine if it supports bearer token authorization.

Additionally, registration scopes can be configured as part of a `sso-session`. Scope is a mechanism in OAuth 2.0 to limit an application's access to a user's account. An application can request one or more scopes, and the access token issued to the application will be limited to the scopes granted. These scopes define the permissions requested to be authorized for the registered OIDC client and access tokens retrieved by the client. The following example sets `sso_registration_scopes` to provide access for listing accounts/roles:

```
[sso-session my-sso]  
sso_region = us-east-1  
sso_start_url = https://my-sso-portal.awsapps.com/start  
sso_registration_scopes = sso:account:access
```

The authentication token is cached to disk under the `~/ .aws/sso/cache` directory with a filename based on the session name.

For more information on this configuration type, see [the section called "Configure automatic token refresh"](#).

Section type: services

The `services` section is a group of settings that configures custom endpoints for AWS service requests. A profile then is linked to a `services` section.

```
[profile dev]  
services = my-services
```

The `services` section is separated into subsections by `<SERVICE> =` lines, where `<SERVICE>` is the AWS service identifier key. The AWS service identifier is based on the API model's `serviceId` by replacing all spaces with underscores and lowercasing all letters. For a list of all service identifier keys to use in the `services` section, see [Use endpoints in the AWS CLI](#). The service identifier key is followed by nested settings with each on its own line and indented by two spaces.

The following example configures the endpoint to use for requests made to the Amazon DynamoDB service in the *my-services* section that is used in the *dev* profile. Any immediately following lines that are indented are included in that subsection and apply to that service.

```
[profile dev]
services = my-services

[services my-services]
dynamodb =
  endpoint_url = http://localhost:8000
```

For more information on service-specific endpoints, see [Use endpoints in the AWS CLI](#).

If your profile has role-based credentials configured through a `source_profile` parameter for IAM assume role functionality, the SDK only uses service configurations for the specified profile. It does not use profiles that are role chained to it. For example, using the following shared config file:

```
[profile A]
credential_source = Ec2InstanceMetadata
endpoint_url = https://profile-a-endpoint.aws/

[profile B]
source_profile = A
role_arn = arn:aws:iam::123456789012:role/roleB
services = profileB

[services profileB]
ec2 =
  endpoint_url = https://profile-b-ec2-endpoint.aws
```

If you use profile B and make a call in your code to Amazon EC2, the endpoint resolves as `https://profile-b-ec2-endpoint.aws`. If your code makes a request to any other service, the endpoint resolution will not follow any custom logic. The endpoint does not resolve to the global endpoint defined in profile A. For a global endpoint to take effect for profile B, you would need to set `endpoint_url` directly within profile B.

Where are configuration settings stored?

The AWS CLI stores sensitive credential information that you specify with `aws configure` in a local file named `credentials`, in a folder named `.aws` in your home directory. The less sensitive

configuration options that you specify with `aws configure` are stored in a local file named `config`, also stored in the `.aws` folder in your home directory.

Storing credentials in the config file

You can keep all of your profile settings in a single file as the AWS CLI can read credentials from the `config` file. If there are credentials in both files for a profile sharing the same name, the keys in the credentials file take precedence. We suggest keeping credentials in the `credentials` files. These files are also used by the various language software development kits (SDKs). If you use one of the SDKs in addition to the AWS CLI, confirm if the credentials should be stored in their own file.

Where you find your home directory location varies based on the operating system, but is referred to using the environment variables `%UserProfile%` in Windows and `$HOME` or `~` (tilde) in Unix-based systems. You can specify a non-default location for the files by setting the `AWS_CONFIG_FILE` and `AWS_SHARED_CREDENTIALS_FILE` environment variables to another local path. See [Environment variables to configure the AWS CLI](#) for details.

When you use a shared profile that specifies an AWS Identity and Access Management (IAM) role, the AWS CLI calls the AWS STS `AssumeRole` operation to retrieve temporary credentials. These credentials are then stored (in `~/.aws/cli/cache`). Subsequent AWS CLI commands use the cached temporary credentials until they expire, and at that point the AWS CLI automatically refreshes the credentials.

Using named profiles

If no profile is explicitly defined, the default profile is used.

To use a named profile, add the `--profile profile-name` option to your command. The following example lists all of your Amazon EC2 instances using the credentials and settings defined in the `user1` profile.

```
$ aws ec2 describe-instances --profile user1
```

To use a named profile for multiple commands, you can avoid specifying the profile in every command by setting the `AWS_PROFILE` environment variable as the default profile. You can override this setting by using the `--profile` parameter.

Linux or macOS

```
$ export AWS_PROFILE=user1
```

Windows

```
C:\> setx AWS_PROFILE user1
```

Using [set](#) to set an environment variable changes the value used until the end of the current command prompt session, or until you set the variable to a different value.

Using [setx](#) to set an environment variable changes the value in all command shells that you create after running the command. It does **not** affect any command shell that is already running at the time you run the command. Close and restart the command shell to see the effects of the change.

Setting the environment variable changes the default profile until the end of your shell session, or until you set the variable to a different value. You can make environment variables persistent across future sessions by putting them in your shell's startup script. For more information, see [Environment variables to configure the AWS CLI](#).

Set and view configuration settings using commands

There are several ways to view and set your configuration settings using commands.

[aws configure](#)

Run this command to quickly set and view your credentials, Region, and output format. The following example shows sample values.

```
$ aws configure
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJa1rXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
Default region name [None]: us-west-2
Default output format [None]: json
```

[aws configure set](#)

You can set any credentials or configuration settings using `aws configure set`. Specify the profile that you want to view or modify with the `--profile` setting.

For example, the following command sets the `region` in the profile named `integ`.

```
$ aws configure set region us-west-2 --profile integ
```

To remove a setting, use an empty string as the value, or manually delete the setting in your `config` and `credentials` files in a text editor.

```
$ aws configure set cli_pager "" --profile integ
```

aws configure get

You can retrieve any credentials or configuration settings you've set using `aws configure get`. Specify the profile that you want to view or modify with the `--profile` setting.

For example, the following command retrieves the `region` setting in the profile named `integ`.

```
$ aws configure get region --profile integ  
us-west-2
```

If the output is empty, the setting is not explicitly set and uses the default value.

aws configure import

Import CSV credentials generated from the IAM web console. This is not for credentials generated from IAM Identity Center; customers who use IAM Identity Center should use `aws configure sso`. A CSV file is imported with the profile name matching the username. The CSV file must contain the following headers.

- User Name
- Access key ID
- Secret access key

Note

During initial key pair creation, once you close the **Download .csv file** dialog box, you cannot access your secret access key after you close the dialog box. If you need a `.csv` file, you'll need to create one yourself with the required headers and your stored key pair information. If you do not have access to your key pair information, you need to create a new key pair.

```
$ aws configure import --csv file://credentials.csv
```

aws configure list

To list configuration data, use the `aws configure list` command. This command lists the profile, access key, secret key, and region configuration information used for the specified profile. For each configuration item, it shows the value, where the configuration value was retrieved, and the configuration variable name.

For example, if you provide the AWS Region in an environment variable, this command shows you the name of the region you've configured, that this value came from an environment variable, and the name of the environment variable.

For temporary credential methods such as roles and IAM Identity Center, this command displays the temporarily cached access key and secret access key is displayed.

```
$ aws configure list
  Name                Value                Type    Location
  ----                -
  profile              <not set>           None    None
  access_key           *****ABCD         shared-credentials-file
  secret_key           *****ABCD         shared-credentials-file
  region               us-west-2           env     AWS_DEFAULT_REGION
```

aws configure list-profiles

To list all your profile names, use the `aws configure list-profiles` command.

```
$ aws configure list-profiles
default
test
```

aws configure sso

Run this command to quickly set and view your AWS IAM Identity Center credentials, Region, and output format. The following example shows sample values.

```
$ aws configure sso
SSO session name (Recommended): my-sso
SSO start URL [None]: https://my-sso-portal.awsapps.com/start
SSO region [None]: us-east-1
```

```
SSO registration scopes [None]: ss0:account:access
```

aws configure sso-session

Run this command to quickly set and view your AWS IAM Identity Center credentials, Region, and output format in the sso-session section of the credentials and config files. The following example shows sample values.

```
$ aws configure sso-session  
SSO session name: my-sso  
SSO start URL [None]: https://my-sso-portal.awsapps.com/start  
SSO region [None]: us-east-1  
SSO registration scopes [None]: ss0:account:access
```

Setting new configuration and credentials command examples

The following examples show configuring a default profile with credentials, region, and output specified for different authentication methods.

IAM Identity Center (SSO)

This example is for AWS IAM Identity Center using the `aws configure sso` wizard. For more information, see [the section called "Configure automatic token refresh"](#).

```
$ aws configure sso  
SSO session name (Recommended): my-sso  
SSO start URL [None]: https://my-sso-portal.awsapps.com/start  
SSO region [None]: us-east-1  
  
Attempting to automatically open the SSO authorization page in your default browser.  
  
There are 2 AWS accounts available to you.  
> DeveloperAccount, developer-account-admin@example.com (111122223333)  
   ProductionAccount, production-account-admin@example.com (444455556666)  
  
Using the account ID 111122223333  
  
There are 2 roles available to you.  
> ReadOnly  
   FullAccess
```

```
Using the role name "ReadOnly"  
  
CLI default client Region [None]: us-west-2  
CLI default output format [None]: json  
CLI profile name [123456789011_ReadOnly]: user1
```

IAM Identity Center (Legacy SSO)

This example is for the legacy method of AWS IAM Identity Center using the `aws configure sso` wizard. To use the legacy SSO, leave the session name blank. For more information, see [the section called "Configure legacy non-refreshable"](#).

```
$ aws configure sso  
SSO session name (Recommended):  
SSO start URL [None]: https://my-sso-portal.awsapps.com/start  
SSO region [None]:us-east-1  
  
SSO authorization page has automatically been opened in your default browser.  
Follow the instructions in the browser to complete this authorization request.  
  
There are 2 AWS accounts available to you.  
> DeveloperAccount, developer-account-admin@example.com (111122223333)  
   ProductionAccount, production-account-admin@example.com (444455556666)  
  
Using the account ID 111122223333  
  
There are 2 roles available to you.  
> ReadOnly  
   FullAccess  
  
Using the role name "ReadOnly"  
  
CLI default client Region [None]: us-west-2  
CLI default output format [None]: json  
CLI profile name [123456789011_ReadOnly]: user1
```

Short-term credentials

This example is for the short-term credentials from AWS Identity and Access Management. The `aws configure` wizard is used to set initial values and then the `aws configure set` command assigns the last value needed. For more information, see [the section called "Short-term credentials"](#).

Long-term credentials

Warning

To avoid security risks, don't use IAM users for authentication when developing purpose-built software or working with real data. Instead, use federation with an identity provider such as [AWS IAM Identity Center](#).

This example is for the long-term credentials from AWS Identity and Access Management. For more information, see [the section called "IAM users"](#).

```
$ aws configure
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
Default region name [None]: us-west-2
Default output format [None]: json
```

Supported config file settings

Topics

- [Global settings](#)
- [S3 Custom command settings](#)

The following settings are supported in the `config` file. The values listed in the specified (or default) profile are used unless they are overridden by the presence of an environment variable with the same name, or a command line option with the same name. For more information on what order settings take precedence, see [Configure the AWS CLI](#)

Global settings

`aws_access_key_id`

Specifies the AWS access key used as part of the credentials to authenticate the command request. Although this can be stored in the `config` file, we recommend that you store this in the `credentials` file.

Can be overridden by the `AWS_ACCESS_KEY_ID` environment variable. You can't specify the access key ID as a command line option.

```
aws_access_key_id = AKIAIOSFODNN7EXAMPLE
```

`aws_secret_access_key`

Specifies the AWS secret key used as part of the credentials to authenticate the command request. Although this can be stored in the config file, we recommend that you store this in the `credentials` file.

Can be overridden by the `AWS_SECRET_ACCESS_KEY` environment variable. You can't specify the secret access key as a command line option.

```
aws_secret_access_key = wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

`aws_session_token`

Specifies an AWS session token. A session token is required only if you manually specify temporary security credentials. Although this can be stored in the config file, we recommend that you store this in the `credentials` file.

Can be overridden by the `AWS_SESSION_TOKEN` environment variable. You can't specify the session token as a command line option.

```
aws_session_token = AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT  
+FvqwqKwRc0IfrrRh3c/LTo6UDdyJw00vEVPvLXCrrrUtdnniCEXAMPLE/  
IvU1dYUg2RVAJBanLiHb4IgrmpRV3zrkuWJ0gQs8IZZaIv2BXIa2R40lgk
```

`ca_bundle`

Specifies a CA certificate bundle (a file with the `.pem` extension) that is used to verify SSL certificates.

Can be overridden by the [AWS_CA_BUNDLE](#) environment variable or the [--ca-bundle](#) command line option.

```
ca_bundle = dev/apps/ca-certs/cabundle-2019mar05.pem
```

cli_auto_prompt

Enables the auto-prompt for the AWS CLI version 2. There are two settings that can be used:

- **on** uses the full auto-prompt mode each time you attempt to run an aws command. This includes pressing **ENTER** after both a complete command or incomplete command.

```
cli_auto_prompt = on
```

- **on-partial** uses partial auto-prompt mode. If a command is incomplete or cannot be run due to client-side validation errors, auto-prompt is used. This mode is particular useful if you have pre-existing scripts, runbooks, or you only want to be auto-prompted for commands you are unfamiliar with rather than prompted on every command.

```
cli_auto_prompt = on-partial
```

You can override this setting by using the [aws_cli_auto_prompt](#) environment variable or the [--cli-auto-prompt](#) and [--no-cli-auto-prompt](#) command line parameters.

For information on the AWS CLI version 2 auto-prompt feature, see [Have the AWS CLI prompt you for commands](#).

cli_binary_format

Specifies how the AWS CLI version 2 interprets binary input parameters. It can be one of the following values:

- **base64** – This is the default value. An input parameter that is typed as a binary large object (BLOB) accepts a base64-encoded string. To pass true binary content, put the content in a file and provide the file's path and name with the `fileb://` prefix as the parameter's value. To pass base64-encoded text contained in a file, provide the file's path and name with the `file://` prefix as the parameter's value.
- **raw-in-base64-out** – Default for the AWS CLI version 1. If the setting's value is `raw-in-base64-out`, files referenced using the `file://` prefix is read as text and then the AWS CLI attempts to encode it to binary.

This entry does not have an equivalent environment variable. You can specify the value on a single command by using the `--cli-binary-format raw-in-base64-out` parameter.

```
cli_binary_format = raw-in-base64-out
```


If you reference a binary value in a file using the `fileb://` prefix notation, the AWS CLI *always* expects the file to contain raw binary content and does not attempt to convert the value.

If you reference a binary value in a file using the `file://` prefix notation, the AWS CLI handles the file according to the current `cli_binary_format` setting. If that setting's value is `base64` (the default when not explicitly set), the AWS CLI expects the file to contain base64-encoded text. If that setting's value is `raw-in-base64-out`, the AWS CLI expects the file to contain raw binary content.

cli_history

Disabled by default. This setting enables command history for the AWS CLI. After enabling this setting, the AWS CLI records the history of `aws` commands.

```
cli_history = enabled
```

You can list your history using the `aws history list` command, and use the resulting `command_ids` in the `aws history show` command for details. For more information see [aws history](#) in the *AWS CLI reference guide*.

cli_pager

Specifies the pager program used for output. By default, AWS CLI version 2 returns all output through your operating system's default pager program.

Can be overridden by the `AWS_PAGER` environment variable.

```
cli_pager=less
```

To disable all use of an external paging program, set the variable to an empty string as shown in the following example.

```
cli_pager=
```

cli_timestamp_format

Specifies the format of timestamp values included in the output. You can specify either of the following values:

- **iso8601** – The default value for the AWS CLI version 2. If specified, the AWS CLI reformats all timestamps according to [ISO 8601](#).

ISO 8601 formatted timestamps look like the following examples. The first example shows the time in [Coordinated Universal Time \(UTC\)](#) by including a Z after the time. The date and the time are separated by a T.

```
2019-10-31T22:21:41Z
```

To specify a different time zone, instead of the Z, specify a + or - and the number of hours the desired time zone is ahead of or behind UTC, as a two-digit value. The following example shows the same time as the previous example but adjusted to Pacific Standard time, which is eight hours behind UTC.

```
2019-10-31T14:21:41-08
```

- **wire** – The default value for the AWS CLI version 1. If specified, the AWS CLI displays all timestamp values exactly as received in the HTTP query response.

This entry does not have an equivalent environment variable or command line option.

```
cli_timestamp_format = iso8601
```

[credential_process](#)

Specifies an external command that the AWS CLI runs to generate or retrieve authentication credentials to use for this command. The command must return the credentials in a specific format. For more information about how to use this setting, see [Source credentials with an external process](#).

This entry does not have an equivalent environment variable or command line option.

```
credential_process = /opt/bin/awscreds-retriever --username susan
```

[credential_source](#)

Used within Amazon EC2 instances or containers to specify where the AWS CLI can find credentials to use to assume the role you specified with the `role_arn` parameter. You cannot specify both `source_profile` and `credential_source` in the same profile.

This parameter can have one of three values:

- **Environment** – Specifies that the AWS CLI is to retrieve source credentials from environment variables.
- **Ec2InstanceMetadata** – Specifies that the AWS CLI is to use the IAM role attached to the [EC2 instance profile](#) to get source credentials.
- **EcsContainer** – Specifies that the AWS CLI is to use the IAM role attached to the ECS container as source credentials.

```
credential_source = Ec2InstanceMetadata
```

duration_seconds

Specifies the maximum duration of the role session, in seconds. The value can range from 900 seconds (15 minutes) up to the maximum session duration setting for the role (which can be a maximum of 43200). This is an optional parameter and by default, the value is set to 3600 seconds.

endpoint_url

Specifies the endpoint that is used for all service requests. If this setting is used in the [services](#) section of the config file, then the endpoint is used only for the specified service.

The following example uses the global endpoint `http://localhost:1234` and a service-specific endpoint of `http://localhost:4567` for Amazon S3.

```
[profile dev]
endpoint_url = http://localhost:1234
services = s3-specific

[services s3-specific]
s3 =
    endpoint_url = http://localhost:4567
```

Endpoint configuration settings are located in multiple places, such as the system or user environment variables, local AWS configuration files, or explicitly declared on the command line as a parameter. The AWS CLI endpoint configuration settings take precedence in the following order:

1. The [--endpoint-url](#) command line option.

2. If enabled, the [AWS_IGNORE_CONFIGURED_ENDPOINT_URLS](#) global endpoint environment variable or profile setting [ignore_configure_endpoint_urls](#) to ignore custom endpoints.
3. The value provided by a service-specific environment variable [AWS_ENDPOINT_URL_<SERVICE>](#), such as `AWS_ENDPOINT_URL_DYNAMODB`.
4. The values provided by the [AWS_USE_DUALSTACK_ENDPOINT](#), [AWS_USE_FIPS_ENDPOINT](#), and [AWS_ENDPOINT_URL](#) environment variables.
5. The service-specific endpoint value provided by the [endpoint_url](#) setting within a services section of the shared config file.
6. The value provided by the [endpoint_url](#) setting within a profile of the shared config file.
7. [use_dualstack_endpoint](#), [use_fips_endpoint](#), and [endpoint_url](#) settings.
8. Any default endpoint URL for the respective AWS service is used last. For a list of the standard service endpoints available in each Region, see [AWS Regions and Endpoints](#) in the *Amazon Web Services General Reference*.

ignore_configure_endpoint_urls

If enabled, the AWS CLI ignores all custom endpoint configurations specified in the config file. Valid values are **true** and **false**.

```
ignore_configure_endpoint_urls = true
```

Endpoint configuration settings are located in multiple places, such as the system or user environment variables, local AWS configuration files, or explicitly declared on the command line as a parameter. The AWS CLI endpoint configuration settings take precedence in the following order:

1. The [--endpoint-url](#) command line option.
2. If enabled, the [AWS_IGNORE_CONFIGURED_ENDPOINT_URLS](#) global endpoint environment variable or profile setting [ignore_configure_endpoint_urls](#) to ignore custom endpoints.
3. The value provided by a service-specific environment variable [AWS_ENDPOINT_URL_<SERVICE>](#), such as `AWS_ENDPOINT_URL_DYNAMODB`.
4. The values provided by the [AWS_USE_DUALSTACK_ENDPOINT](#), [AWS_USE_FIPS_ENDPOINT](#), and [AWS_ENDPOINT_URL](#) environment variables.

5. The service-specific endpoint value provided by the [endpoint_url](#) setting within a services section of the shared config file.
6. The value provided by the [endpoint_url](#) setting within a profile of the shared config file.
7. [use_dualstack_endpoint](#), [use_fips_endpoint](#), and [endpoint_url](#) settings.
8. Any default endpoint URL for the respective AWS service is used last. For a list of the standard service endpoints available in each Region, see [AWS Regions and Endpoints](#) in the *Amazon Web Services General Reference*.

[external_id](#)

Specifies a unique identifier that is used by third parties to assume a role in their customers' accounts. This maps to the `ExternalId` parameter in the `AssumeRole` operation. This parameter is needed only if the trust policy for the role specifies a value for `ExternalId`. For more information, see [How to use an external ID when granting access to your AWS resources to a third party](#) in the *IAM User Guide*.

[max_attempts](#)

Specifies a value of maximum retry attempts the AWS CLI retry handler uses, where the initial call counts toward the `max_attempts` value that you provide.

You can override this value by using the `AWS_MAX_ATTEMPTS` environment variable.

```
max_attempts = 3
```

[mfa_serial](#)

The identification number of an MFA device to use when assuming a role. This is mandatory only if the trust policy of the role being assumed includes a condition that requires MFA authentication. The value can be either a serial number for a hardware device (such as GAHT12345678) or an Amazon Resource Name (ARN) for a virtual MFA device (such as `arn:aws:iam::123456789012:mfa/user`).

output

Specifies the default output format for commands requested using this profile. You can specify any of the following values:

- [json](#) – The output is formatted as a [JSON](#) string.
- [yaml](#) – The output is formatted as a [YAML](#) string.

- **`yaml-stream`** – The output is streamed and formatted as a [YAML](#) string. Streaming allows for faster handling of large data types.
- **`text`** – The output is formatted as multiple lines of tab-separated string values. This can be useful to pass the output to a text processor, like `grep`, `sed`, or `awk`.
- **`table`** – The output is formatted as a table using the characters `+|-` to form the cell borders. It typically presents the information in a "human-friendly" format that is much easier to read than the others, but not as programmatically useful.

Can be overridden by the `AWS_DEFAULT_OUTPUT` environment variable or the `--output` command line option.

```
output = table
```

parameter_validation

Specifies whether the AWS CLI client attempts to validate parameters before sending them to the AWS service endpoint.

- **`true`** – This is the default value. If specified, the AWS CLI performs local validation of command line parameters.
- **`false`** – If specified, the AWS CLI does not validate command line parameters before sending them to the AWS service endpoint.

This entry does not have an equivalent environment variable or command line option.

```
parameter_validation = false
```

region

Specifies the AWS Region to send requests to for commands requested using this profile.

- You can specify any of the Region codes available for the chosen service as listed in [AWS Regions and Endpoints](#) in the *Amazon Web Services General Reference*.
- `aws_global` enables you to specify the global endpoint for services that support a global endpoint in addition to Regional endpoints, such as AWS Security Token Service (AWS STS) and Amazon Simple Storage Service (Amazon S3).

You can override this value by using the `AWS_REGION` environment variable, `AWS_DEFAULT_REGION` environment variable, or the `--region` command line option.

```
region = us-west-2
```

retry_mode

Specifies which retry mode AWS CLI uses. There are three retry modes available: legacy (default), standard, and adaptive. For more information on retries, see [AWS CLI retries](#).

You can override this value by using the `AWS_RETRY_MODE` environment variable.

```
retry_mode = standard
```

role_arn

Specifies the Amazon Resource Name (ARN) of an IAM role that you want to use to run the AWS CLI commands. You must also specify one of the following parameters to identify the credentials that have permission to assume this role:

- `source_profile`
- `credential_source`

```
role_arn = arn:aws:iam::123456789012:role/role-name
```

The environment variable [AWS_ROLE_ARN](#) overrides this setting.

For more information on using web identities, see [the section called "Assume role with web identity"](#).

role_session_name

Specifies the name to attach to the role session. This value is provided to the `RoleSessionName` parameter when the AWS CLI calls the `AssumeRole` operation, and becomes part of the assumed role user ARN: `arn:aws:sts::123456789012:assumed-role/role_name/role_session_name`. This is an optional parameter. If you do not provide this value, a session name is generated automatically. This name appears in AWS CloudTrail logs for entries associated with this session.

```
role_session_name = maria_garcia_role
```

The environment variable [AWS_ROLE_SESSION_NAME](#) overrides this setting.

For more information on using web identities, see [the section called “Assume role with web identity”](#).

services

Specifies the service configuration to use for your profile.

```
[profile dev-s3-specific-and-global]
endpoint_url = http://localhost:1234
services = s3-specific

[services s3-specific]
s3 =
  endpoint_url = http://localhost:4567
```

For more information on the services section, see [the section called “services”](#).

The environment variable [AWS_ROLE_SESSION_NAME](#) overrides this setting.

For more information on using web identities, see [the section called “Assume role with web identity”](#).

source_profile

Specifies a named profile with long-term credentials that the AWS CLI can use to assume a role that you specified with the `role_arn` parameter. You cannot specify both `source_profile` and `credential_source` in the same profile.

```
source_profile = production-profile
```

sso_account_id

Specifies the AWS account ID that contains the IAM role with the permission that you want to grant to the associated IAM Identity Center user.

This setting does not have an environment variable or command line option.

```
sso_account_id = 123456789012
```

sso_region

Specifies the AWS Region that contains the AWS access portal host. This is separate from, and can be a different Region than the default CLI `region` parameter.

This setting does not have an environment variable or command line option.

```
sso_region = us_west-2
```

sso_registration_scopes

A comma-delimited list of scopes to be authorized for the sso-session. Scopes authorize access to IAM Identity Center bearer token authorized endpoints. A valid scope is a string, such as sso:account:access. This setting isn't applicable to the legacy non-refreshable configuration.

```
sso_registration_scopes = sso:account:access
```

sso_role_name

Specifies the friendly name of the IAM role that defines the user's permissions when using this profile.

This setting does not have an environment variable or command line option.

```
sso_role_name = ReadAccess
```

sso_start_url

Specifies the URL that points to the organization's AWS access portal. The AWS CLI uses this URL to establish a session with the IAM Identity Center service to authenticate its users. To find your AWS access portal URL, use one of the following:

- Open your invitation email, the AWS access portal URL is listed.
- Open the AWS IAM Identity Center console at <https://console.aws.amazon.com/singlesignon/>. The AWS access portal URL is listed in your settings.

This setting does not have an environment variable or command line option.

```
sso_start_url = https://my-sso-portal.awsapps.com/start
```

use_dualstack_endpoint

Enables the use of dual-stack endpoints to send AWS requests. To learn more about dual-stack endpoints, which support both IPv4 and IPv6 traffic, see [Using Amazon S3 dual-stack endpoints](#) in the *Amazon Simple Storage Service User Guide*. Dual-stack endpoints are available for some services in some regions. If a dual-stack endpoint does not exist for the service or AWS Region, the request fails. This is disabled by default.

This is mutually exclusive with the `use_accelerate_endpoint` setting.

Endpoint configuration settings are located in multiple places, such as the system or user environment variables, local AWS configuration files, or explicitly declared on the command line as a parameter. The AWS CLI endpoint configuration settings take precedence in the following order:

1. The `--endpoint-url` command line option.
2. If enabled, the `AWS_IGNORE_CONFIGURED_ENDPOINT_URLS` global endpoint environment variable or profile setting `ignore_configure_endpoint_urls` to ignore custom endpoints.
3. The value provided by a service-specific environment variable `AWS_ENDPOINT_URL_<SERVICE>`, such as `AWS_ENDPOINT_URL_DYNAMODB`.
4. The values provided by the `AWS_USE_DUALSTACK_ENDPOINT`, `AWS_USE_FIPS_ENDPOINT`, and `AWS_ENDPOINT_URL` environment variables.
5. The service-specific endpoint value provided by the `endpoint_url` setting within a services section of the shared config file.
6. The value provided by the `endpoint_url` setting within a profile of the shared config file.
7. `use_dualstack_endpoint`, `use_fips_endpoint`, and `endpoint_url` settings.
8. Any default endpoint URL for the respective AWS service is used last. For a list of the standard service endpoints available in each Region, see [AWS Regions and Endpoints](#) in the *Amazon Web Services General Reference*.

use_fips_endpoint

Some AWS services offer endpoints that support [Federal Information Processing Standard \(FIPS\) 140-2](#) in some AWS Regions. When the AWS service supports FIPS, this setting specifies what FIPS endpoint the AWS CLI should use. Unlike standard AWS endpoints, FIPS endpoints

use a TLS software library that complies with FIPS 140-2. These endpoints might be required by enterprises that interact with the United States government.

If this setting is enabled, but a FIPS endpoint does not exist for the service in your AWS Region, the AWS command may fail. In this case, manually specify the endpoint to use in the command using the [--endpoint-url](#) option or use [service-specific endpoints](#).

For more information on specifying FIPS endpoints by AWS Region, see [FIPS Endpoints by Service](#).

Endpoint configuration settings are located in multiple places, such as the system or user environment variables, local AWS configuration files, or explicitly declared on the command line as a parameter. The AWS CLI endpoint configuration settings take precedence in the following order:

1. The [--endpoint-url](#) command line option.
2. If enabled, the [AWS_IGNORE_CONFIGURED_ENDPOINT_URLS](#) global endpoint environment variable or profile setting [ignore_configure_endpoint_urls](#) to ignore custom endpoints.
3. The value provided by a service-specific environment variable [AWS_ENDPOINT_URL_<SERVICE>](#), such as [AWS_ENDPOINT_URL_DYNAMODB](#).
4. The values provided by the [AWS_USE_DUALSTACK_ENDPOINT](#), [AWS_USE_FIPS_ENDPOINT](#), and [AWS_ENDPOINT_URL](#) environment variables.
5. The service-specific endpoint value provided by the [endpoint_url](#) setting within a services section of the shared config file.
6. The value provided by the [endpoint_url](#) setting within a profile of the shared config file.
7. [use_dualstack_endpoint](#), [use_fips_endpoint](#), and [endpoint_url](#) settings.
8. Any default endpoint URL for the respective AWS service is used last. For a list of the standard service endpoints available in each Region, see [AWS Regions and Endpoints](#) in the *Amazon Web Services General Reference*.

[web_identity_token_file](#)

Specifies the path to a file that contains an OAuth 2.0 access token or OpenID Connect ID token that is provided by an identity provider. The AWS CLI loads the contents of this file and passes it as the `WebIdentityToken` argument to the `AssumeRoleWithWebIdentity` operation.

The environment variable [AWS_WEB_IDENTITY_TOKEN_FILE](#) overrides this setting.

For more information on using web identities, see [the section called "Assume role with web identity"](#).

tcp_keepalive

Specifies whether the AWS CLI client uses TCP keep-alive packets.

This entry does not have an equivalent environment variable or command line option.

```
tcp_keepalive = false
```

S3 Custom command settings

Amazon S3 supports several settings that configure how the AWS CLI performs Amazon S3 operations. Some apply to all S3 commands in both the `s3api` and `s3` namespaces. Others are specifically for the S3 "custom" commands that abstract common operations and do more than a one-to-one mapping to an API operation. The `aws s3 transfer` commands `cp`, `sync`, `mv`, and `rm` have additional settings you can use to control S3 transfers.

All of these options can be configured by specifying the `s3` nested setting in your `config` file. Each setting is then indented on its own line.

Note

These settings are entirely optional. You should be able to successfully use the `aws s3 transfer` commands without configuring any of these settings. These settings are provided to enable you to tune for performance or to account for the specific environment where you are running these `aws s3` commands.

These settings are all set under a top-level `s3` key in the `config` file, as shown in the following example for the development profile.

```
[profile development]
s3 =
  max_concurrent_requests = 20
  max_queue_size = 10000
  multipart_threshold = 64MB
  multipart_chunksize = 16MB
```

```
max_bandwidth = 50MB/s
use_accelerate_endpoint = true
addressing_style = path
```

The following settings apply to any S3 command in the `s3` or `s3api` namespaces.

addressing_style

Specifies which addressing style to use. This controls whether the bucket name is in the hostname or is part of the URL. Valid values are: `path`, `virtual`, and `auto`. The default value is `auto`.

There are two styles of constructing an Amazon S3 endpoint. The first is called `virtual` and includes the bucket name as part of the hostname. For example: `https://bucketname.s3.amazonaws.com`. Alternatively, with the `path` style, you treat the bucket name as if it is a path in the URI; for example, `https://s3.amazonaws.com/bucketname`. The default value in the CLI is to use `auto`, which attempts to use the `virtual` style where it can, but will fall back to `path` style when required. For example, if your bucket name is not DNS compatible, the bucket name cannot be part of the hostname and must be in the path. With `auto`, the CLI will detect this condition and automatically switch to `path` style for you. If you set the addressing style to `path`, you must then ensure that the AWS Region you configured in the AWS CLI matches the Region of your bucket.

payload_signing_enabled

Specifies whether to SHA256 sign sigv4 payloads. By default, this is disabled for streaming uploads (`UploadPart` and `PutObject`) when using HTTPS. By default, this is set to `false` for streaming uploads (`UploadPart` and `PutObject`), but only if a `ContentMD5` is present (it is generated by default) and the endpoint uses HTTPS.

If set to `true`, S3 requests receive additional content validation in the form of a SHA256 checksum which is calculated for you and included in the request signature. If set to `false`, the checksum isn't calculated. Disabling this can be useful to reduce the performance overhead created by the checksum calculation.

use_accelerate_endpoint

Use the Amazon S3 Accelerate endpoint for all `s3` and `s3api` commands. The default value is `false`. This is mutually exclusive with the `use_dualstack_endpoint` setting.

If set to true, the AWS CLI directs all Amazon S3 requests to the S3 Accelerate endpoint at `s3-accelerate.amazonaws.com`. To use this endpoint, you must enable your bucket to use S3 Accelerate. All requests are sent using the virtual style of bucket addressing: `my-bucket.s3-accelerate.amazonaws.com`. Any `ListBuckets`, `CreateBucket`, and `DeleteBucket` requests aren't sent to the S3 Accelerate endpoint as that endpoint doesn't support those operations. This behavior can also be set if the `--endpoint-url` parameter is set to `https://s3-accelerate.amazonaws.com` or `http://s3-accelerate.amazonaws.com` for any `s3` or `s3api` command.

The following settings apply only to commands in the `s3` namespace command set.

max_bandwidth

Specifies the maximum bandwidth that can be consumed for uploading and downloading data to and from Amazon S3. The default is no limit.

This limits the maximum bandwidth that the S3 commands can use to transfer data to and from Amazon S3. This value applies to only uploads and downloads; it doesn't apply to copies or deletes. The value is expressed as bytes per second. The value can be specified as:

- An integer. For example, `1048576` sets the maximum bandwidth usage to 1 megabyte per second.
- An integer followed by a rate suffix. You can specify rate suffixes using: `KB/s`, `MB/s`, or `GB/s`. For example, `300KB/s`, `10MB/s`.

In general, we recommend that you first try to lower bandwidth consumption by lowering `max_concurrent_requests`. If that doesn't adequately limit bandwidth consumption to the desired rate, you can use the `max_bandwidth` setting to further limit bandwidth consumption. This is because `max_concurrent_requests` controls how many threads are currently running. If you instead first lower `max_bandwidth` but leave a high `max_concurrent_requests` setting, it can result in threads having to wait unnecessarily. This can lead to excess resource consumption and connection timeouts.

max_concurrent_requests

Specifies the maximum number of concurrent requests. The default value is 10.

The `aws s3` transfer commands are multithreaded. At any given time, multiple Amazon S3 requests can be running. For example, when you use the command `aws s3 cp localdir`

`s3://bucket/ --recursive` to upload files to an S3 bucket, the AWS CLI can upload the files `localdir/file1`, `localdir/file2`, and `localdir/file3` in parallel. The setting `max_concurrent_requests` specifies the maximum number of transfer operations that can run at the same time.

You might need to change this value for a few reasons:

- Decreasing this value – On some environments, the default of 10 concurrent requests can overwhelm a system. This can cause connection timeouts or slow the responsiveness of the system. Lowering this value makes the S3 transfer commands less resource intensive. The tradeoff is that S3 transfers can take longer to complete. Lowering this value might be necessary if you use a tool to limit bandwidth.
- Increasing this value – In some scenarios, you might want the Amazon S3 transfers to complete as quickly as possible, using as much network bandwidth as necessary. In this scenario, the default number of concurrent requests might not be sufficient to use all of the available network bandwidth. Increasing this value can improve the time it takes to complete an Amazon S3 transfer.

max_queue_size

Specifies the maximum number of tasks in the task queue. The default value is 1000.

The AWS CLI internally uses a model where it queues up Amazon S3 tasks that are then executed by consumers whose numbers are limited by `max_concurrent_requests`. A task generally maps to a single Amazon S3 operation. For example, a task could be a `PutObjectTask`, or a `GetObjectTask`, or an `UploadPartTask`. The rate at which tasks are added to the queue can be much faster than the rate at which consumers finish the tasks. To avoid unbounded growth, the task queue size is capped to a specific size. This setting changes the value of that maximum number.

You generally don't need to change this setting. This setting also corresponds to the number of tasks that the AWS CLI is aware of that need to be run. This means that by default the AWS CLI can only see 1000 tasks ahead. Increasing this value means that the AWS CLI can more quickly know the total number of tasks needed, assuming that the queuing rate is quicker than the rate of task completion. The tradeoff is that a larger `max_queue_size` requires more memory.

multipart_chunksize

Specifies the chunk size that the AWS CLI uses for multipart transfers of individual files. The default value is 8 MB, with a minimum of 5 MB.

When a file transfer exceeds the `multipart_threshold`, the AWS CLI divides the file into chunks of this size. This value can be specified using the same syntax as `multipart_threshold`, either as the number of bytes as an integer, or by using a size and a suffix.

`multipart_threshold`

Specifies the size threshold the AWS CLI uses for multipart transfers of individual files. The default value is 8 MB.

When uploading, downloading, or copying a file, the Amazon S3 commands switch to multipart operations if the file exceeds this size. You can specify this value in one of two ways:

- The file size in bytes. For example, 1048576.
- The file size with a size suffix. You can use KB, MB, GB, or TB. For example: 10MB, 1GB.

Note

S3 can impose constraints on valid values that can be used for multipart operations. For more information, see the [S3 Multipart Upload documentation](#) in the *Amazon Simple Storage Service User Guide*.

Environment variables to configure the AWS CLI

Environment variables provide another way to specify configuration options and credentials, and can be useful for scripting or temporarily setting a named profile as the default.

Precedence of options

- If you specify an option by using one of the environment variables described in this topic, it overrides any value loaded from a profile in the configuration file.
- If you specify an option by using a parameter on the AWS CLI command line, it overrides any value from either the corresponding environment variable or a profile in the configuration file.

For more information about precedence and how the AWS CLI determines which credentials to use, see [Configure the AWS CLI](#).

Topics

- [How to set environment variables](#)
- [AWS CLI supported environment variables](#)

How to set environment variables

The following examples show how you can configure environment variables for the default user.

Linux or macOS

```
$ export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
$ export AWS_DEFAULT_REGION=us-west-2
```

Setting the environment variable changes the value used until the end of your shell session, or until you set the variable to a different value. You can make the variables persistent across future sessions by setting them in your shell's startup script.

Windows Command Prompt

To set for all sessions

```
C:\> setx AWS_ACCESS_KEY_ID AKIAIOSFODNN7EXAMPLE
C:\> setx AWS_SECRET_ACCESS_KEY wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
C:\> setx AWS_DEFAULT_REGION us-west-2
```

Using [setx](#) to set an environment variable changes the value used in both the current command prompt session and all command prompt sessions that you create after running the command. It does **not** affect other command shells that are already running at the time you run the command. You may need to restart your terminal for settings to load.

To set for current session only

Using [set](#) to set an environment variable changes the value used until the end of the current command prompt session, or until you set the variable to a different value.

```
C:\> set AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
C:\> set AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
C:\> set AWS_DEFAULT_REGION=us-west-2
```

PowerShell

```
PS C:\> $Env:AWS_ACCESS_KEY_ID="AKIAIOSFODNN7EXAMPLE"
PS C:\> $Env:AWS_SECRET_ACCESS_KEY="wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY"
PS C:\> $Env:AWS_DEFAULT_REGION="us-west-2"
```

If you set an environment variable at the PowerShell prompt as shown in the previous examples, it saves the value for only the duration of the current session. To make the environment variable setting persistent across all PowerShell and Command Prompt sessions, store it by using the **System** application in **Control Panel**. Alternatively, you can set the variable for all future PowerShell sessions by adding it to your PowerShell profile. See the [PowerShell documentation](#) for more information about storing environment variables or persisting them across sessions.

AWS CLI supported environment variables

The AWS CLI supports the following environment variables.

AWS_ACCESS_KEY_ID

Specifies an AWS access key associated with an IAM account.

If defined, this environment variable overrides the value for the profile setting `aws_access_key_id`. You can't specify the access key ID by using a command line option.

AWS_CA_BUNDLE

Specifies the path to a certificate bundle to use for HTTPS certificate validation.

If defined, this environment variable overrides the value for the profile setting `ca_bundle`. You can override this environment variable by using the `--ca-bundle` command line parameter.

AWS_CLI_AUTO_PROMPT

Enables the auto-prompt for the AWS CLI version 2. There are two settings that can be used:

- **on** uses the full auto-prompt mode each time you attempt to run an aws command. This includes pressing **ENTER** after both a complete command or incomplete command.
- **on-partial** uses partial auto-prompt mode. If a command is incomplete or cannot be run due to client-side validation errors, auto-prompt is used. This mode is useful if you have pre-

existing scripts, runbooks, or you only want to be auto-prompted for commands you are unfamiliar with rather than prompted on every command.

If defined, this environment variable overrides the value for the [cli_auto_prompt](#) profile setting. You can override this environment variable by using the [--cli-auto-prompt](#) and [--no-cli-auto-prompt](#) command line parameters.

For information on the AWS CLI version 2 auto-prompt feature, see [Have the AWS CLI prompt you for commands](#).

AWS_CLI_FILE_ENCODING

Specifies the encoding used for text files. By default encoding matches your locale. To set encoding different from the locale, use the `aws_cli_file_encoding` environment variable. For example, if you use Windows with default encoding CP1252, setting `aws_cli_file_encoding=UTF-8` sets the CLI to open text files using UTF-8.

AWS_CLI_S3_MV_VALIDATE_SAME_S3_PATHS

If the source and destination buckets are the same when using custom the `s3 mv` command, the source file or object can be moved onto itself, which can result in accidental deletion of your source file or object. The `AWS_CLI_S3_MV_VALIDATE_SAME_S3_PATHS` environment variable and `--validate-same-s3-paths` option specifies whether to validate your access point ARNs or access point aliases in your Amazon S3 source or destination URIs.

Note

Path validation for `s3 mv` requires additional API calls.

AWS_CONFIG_FILE

Specifies the location of the file that the AWS CLI uses to store configuration profiles. The default path is `~/.aws/config`.

You can't specify this value in a named profile setting or by using a command line parameter.

AWS_DATA_PATH

A list of additional directories to check outside of the built-in search path of `~/.aws/models` when loading AWS CLI data. Setting this environment variable indicates additional directories to

check first before falling back to the built-in search path. Multiple entries should be separated with the `os.pathsep` character, which is `:` on Linux or macOS and `;` on Windows.

AWS_DEFAULT_OUTPUT

Specifies the [output format](#) to use.

If defined, this environment variable overrides the value for the profile setting `output`. You can override this environment variable by using the `--output` command line parameter.

AWS_DEFAULT_REGION

The `Default region name` identifies the AWS Region whose servers you want to send your requests to by default. This is typically the Region closest to you, but it can be any Region. For example, you can type `us-west-2` to use US West (Oregon). This is the Region that all later requests are sent to, unless you specify otherwise in an individual command.

Note

You must specify an AWS Region when using the AWS CLI, either explicitly or by setting a default Region. For a list of the available Regions, see [Regions and Endpoints](#). The Region designators used by the AWS CLI are the same names that you see in AWS Management Console URLs and service endpoints.

If defined, this environment variable overrides the value for the profile setting `region`. You can override this environment variable by using the `--region` command line parameter and the AWS SDK compatible `AWS_REGION` environment variable.

AWS_EC2_METADATA_DISABLED

Disables the use of the Amazon EC2 instance metadata service (IMDS).

If set to `true`, user credentials or configuration (like the Region) are not requested from IMDS.

AWS_ENDPOINT_URL

Specifies the endpoint that is used for all service requests.

Endpoint configuration settings are located in multiple places, such as the system or user environment variables, local AWS configuration files, or explicitly declared on the command line

as a parameter. The AWS CLI endpoint configuration settings take precedence in the following order:

1. The [--endpoint-url](#) command line option.
2. If enabled, the [AWS_IGNORE_CONFIGURED_ENDPOINT_URLS](#) global endpoint environment variable or profile setting [ignore_configure_endpoint_urls](#) to ignore custom endpoints.
3. The value provided by a service-specific environment variable [AWS_ENDPOINT_URL_<SERVICE>](#), such as `AWS_ENDPOINT_URL_DYNAMODB`.
4. The values provided by the [AWS_USE_DUALSTACK_ENDPOINT](#), [AWS_USE_FIPS_ENDPOINT](#), and [AWS_ENDPOINT_URL](#) environment variables.
5. The service-specific endpoint value provided by the [endpoint_url](#) setting within a services section of the shared config file.
6. The value provided by the [endpoint_url](#) setting within a profile of the shared config file.
7. [use_dualstack_endpoint](#), [use_fips_endpoint](#), and [endpoint_url](#) settings.
8. Any default endpoint URL for the respective AWS service is used last. For a list of the standard service endpoints available in each Region, see [AWS Regions and Endpoints](#) in the *Amazon Web Services General Reference*.

AWS_ENDPOINT_URL_<SERVICE>

Specifies a custom endpoint that is used for a specific service, where `<SERVICE>` is replaced with the AWS service identifier. For example, Amazon DynamoDB has a `serviceId` of [DynamoDB](#). For this service, the endpoint URL environment variable is `AWS_ENDPOINT_URL_DYNAMODB`.

For a list of all service-specific environment variables, see [List of service-specific identifiers](#).

Endpoint configuration settings are located in multiple places, such as the system or user environment variables, local AWS configuration files, or explicitly declared on the command line as a parameter. The AWS CLI endpoint configuration settings take precedence in the following order:

1. The [--endpoint-url](#) command line option.
2. If enabled, the [AWS_IGNORE_CONFIGURED_ENDPOINT_URLS](#) global endpoint environment variable or profile setting [ignore_configure_endpoint_urls](#) to ignore custom endpoints.

3. The value provided by a service-specific environment variable [AWS_ENDPOINT_URL_<SERVICE>](#), such as `AWS_ENDPOINT_URL_DYNAMODB`.
4. The values provided by the [AWS_USE_DUALSTACK_ENDPOINT](#), [AWS_USE_FIPS_ENDPOINT](#), and [AWS_ENDPOINT_URL](#) environment variables.
5. The service-specific endpoint value provided by the [endpoint_url](#) setting within a services section of the shared config file.
6. The value provided by the [endpoint_url](#) setting within a profile of the shared config file.
7. [use_dualstack_endpoint](#), [use_fips_endpoint](#), and [endpoint_url](#) settings.
8. Any default endpoint URL for the respective AWS service is used last. For a list of the standard service endpoints available in each Region, see [AWS Regions and Endpoints](#) in the *Amazon Web Services General Reference*.

AWS_IGNORE_CONFIGURED_ENDPOINT_URLS

If enabled, the AWS CLI ignores all custom endpoint configurations. Valid values are **true** and **false**.

Endpoint configuration settings are located in multiple places, such as the system or user environment variables, local AWS configuration files, or explicitly declared on the command line as a parameter. The AWS CLI endpoint configuration settings take precedence in the following order:

1. The [--endpoint-url](#) command line option.
2. If enabled, the [AWS_IGNORE_CONFIGURED_ENDPOINT_URLS](#) global endpoint environment variable or profile setting [ignore_configure_endpoint_urls](#) to ignore custom endpoints.
3. The value provided by a service-specific environment variable [AWS_ENDPOINT_URL_<SERVICE>](#), such as `AWS_ENDPOINT_URL_DYNAMODB`.
4. The values provided by the [AWS_USE_DUALSTACK_ENDPOINT](#), [AWS_USE_FIPS_ENDPOINT](#), and [AWS_ENDPOINT_URL](#) environment variables.
5. The service-specific endpoint value provided by the [endpoint_url](#) setting within a services section of the shared config file.
6. The value provided by the [endpoint_url](#) setting within a profile of the shared config file.
7. [use_dualstack_endpoint](#), [use_fips_endpoint](#), and [endpoint_url](#) settings.

- Any default endpoint URL for the respective AWS service is used last. For a list of the standard service endpoints available in each Region, see [AWS Regions and Endpoints](#) in the *Amazon Web Services General Reference*.

AWS_MAX_ATTEMPTS

Specifies a value of maximum retry attempts the AWS CLI retry handler uses, where the initial call counts toward the value that you provide. For more information on retries, see [AWS CLI retries](#).

If defined, this environment variable overrides the value for the profiles setting `max_attempts`.

AWS_METADATA_SERVICE_NUM_ATTEMPTS

When attempting to retrieve credentials on an Amazon EC2 instance that has been configured with an IAM role, the AWS CLI attempts to retrieve credentials once from the instance metadata service before stopping. If you know your commands will run on an Amazon EC2 instance, you can increase this value to make AWS CLI retry multiple times before giving up.

AWS_METADATA_SERVICE_TIMEOUT

The number of seconds before a connection to the instance metadata service should time out. When attempting to retrieve credentials on an Amazon EC2 instance that is configured with an IAM role, a connection to the instance metadata service times out after 1 second by default. If you know you're running on an Amazon EC2 instance with an IAM role configured, you can increase this value if needed.

AWS_PAGER

Specifies the pager program used for output. By default, AWS CLI version 2 returns all output through your operating system's default pager program.

To disable all use of an external paging program, set the variable to an empty string.

If defined, this environment variable overrides the value for the profile setting `cli_pager`.

AWS_PROFILE

Specifies the name of the AWS CLI profile with the credentials and options to use. This can be the name of a profile stored in a `credentials` or `config` file, or the value `default` to use the default profile.

If defined, this environment variable overrides the behavior of using the profile named `[default]` in the configuration file. You can override this environment variable by using the `--profile` command line parameter.

AWS_REGION

The AWS SDK compatible environment variable that specifies the AWS Region to send the request to.

If defined, this environment variable overrides the values in the environment variable `AWS_DEFAULT_REGION` and the profile setting `region`. You can override this environment variable by using the `--region` command line parameter.

AWS_RETRY_MODE

Specifies which retry mode AWS CLI uses. There are three retry modes available: legacy (default), standard, and adaptive. For more information on retries, see [AWS CLI retries](#).

If defined, this environment variable overrides the value for the profiles setting `retry_mode`.

AWS_ROLE_ARN

Specifies the Amazon Resource Name (ARN) of an IAM role with a web identity provider that you want to use to run the AWS CLI commands.

Used with the `AWS_WEB_IDENTITY_TOKEN_FILE` and `AWS_ROLE_SESSION_NAME` environment variables.

If defined, this environment variable overrides the value for the profile setting [role_arn](#). You can't specify a role session name as a command line parameter.

Note

This environment variable only applies to an assumed role with web identity provider it does not apply to the general assume role provider configuration.

For more information on using web identities, see [the section called “Assume role with web identity”](#).

AWS_ROLE_SESSION_NAME


Specifies the name to attach to the role session. This value is provided to the `RoleSessionName` parameter when the AWS CLI calls the `AssumeRole` operation, and becomes part of the assumed role user ARN: `arn:aws:sts::123456789012:assumed-role/role_name/role_session_name`. This is an optional parameter. If you do not provide

this value, a session name is generated automatically. This name appears in AWS CloudTrail logs for entries associated with this session.

If defined, this environment variable overrides the value for the profile setting [role_session_name](#).

Used with the `AWS_ROLE_ARN` and `AWS_WEB_IDENTITY_TOKEN_FILE` environment variables.

For more information on using web identities, see [the section called "Assume role with web identity"](#).

 **Note**

This environment variable only applies to an assumed role with web identity provider it does not apply to the general assume role provider configuration.

AWS_SECRET_ACCESS_KEY

Specifies the secret key associated with the access key. This is essentially the "password" for the access key.

If defined, this environment variable overrides the value for the profile setting `aws_secret_access_key`. You can't specify the secret access key ID as a command line option.

AWS_SESSION_TOKEN

Specifies the session token value that is required if you are using temporary security credentials that you retrieved directly from AWS STS operations. For more information, see the [Output section of the assume-role command](#) in the *AWS CLI Command Reference*.

If defined, this environment variable overrides the value for the profile setting `aws_session_token`.

AWS_SHARED_CREDENTIALS_FILE

Specifies the location of the file that the AWS CLI uses to store access keys. The default path is `~/.aws/credentials`.

You can't specify this value in a named profile setting or by using a command line parameter.

AWS_USE_DUALSTACK_ENDPOINT

Enables the use of dual-stack endpoints to send AWS requests. To learn more about dual-stack endpoints, which support both IPv4 and IPv6 traffic, see [Using Amazon S3 dual-stack endpoints](#) in the *Amazon Simple Storage Service User Guide*. Dual-stack endpoints are available for some services in some regions. If a dual-stack endpoint does not exist for the service or AWS Region, the request fails. This is disabled by default.

Endpoint configuration settings are located in multiple places, such as the system or user environment variables, local AWS configuration files, or explicitly declared on the command line as a parameter. The AWS CLI endpoint configuration settings take precedence in the following order:

1. The [--endpoint-url](#) command line option.
2. If enabled, the [AWS_IGNORE_CONFIGURED_ENDPOINT_URLS](#) global endpoint environment variable or profile setting [ignore_configure_endpoint_urls](#) to ignore custom endpoints.
3. The value provided by a service-specific environment variable [AWS_ENDPOINT_URL_<SERVICE>](#), such as [AWS_ENDPOINT_URL_DYNAMODB](#).
4. The values provided by the [AWS_USE_DUALSTACK_ENDPOINT](#), [AWS_USE_FIPS_ENDPOINT](#), and [AWS_ENDPOINT_URL](#) environment variables.
5. The service-specific endpoint value provided by the [endpoint_url](#) setting within a services section of the shared config file.
6. The value provided by the [endpoint_url](#) setting within a profile of the shared config file.
7. [use_dualstack_endpoint](#), [use_fips_endpoint](#), and [endpoint_url](#) settings.
8. Any default endpoint URL for the respective AWS service is used last. For a list of the standard service endpoints available in each Region, see [AWS Regions and Endpoints](#) in the *Amazon Web Services General Reference*.

AWS_USE_FIPS_ENDPOINT

Some AWS services offer endpoints that support [Federal Information Processing Standard \(FIPS\) 140-2](#) in some AWS Regions. When the AWS service supports FIPS, this setting specifies what FIPS endpoint the AWS CLI should use. Unlike standard AWS endpoints, FIPS endpoints use a TLS software library that complies with FIPS 140-2. These endpoints might be required by enterprises that interact with the United States government.

If this setting is enabled, but a FIPS endpoint does not exist for the service in your AWS Region, the AWS command may fail. In this case, manually specify the endpoint to use in the command using the `--endpoint-url` option or use [service-specific endpoints](#).

For more information on specifying FIPS endpoints by AWS Region, see [FIPS Endpoints by Service](#).

Endpoint configuration settings are located in multiple places, such as the system or user environment variables, local AWS configuration files, or explicitly declared on the command line as a parameter. The AWS CLI endpoint configuration settings take precedence in the following order:

1. The `--endpoint-url` command line option.
2. If enabled, the `AWS_IGNORE_CONFIGURED_ENDPOINT_URLS` global endpoint environment variable or profile setting `ignore_configure_endpoint_urls` to ignore custom endpoints.
3. The value provided by a service-specific environment variable `AWS_ENDPOINT_URL_<SERVICE>`, such as `AWS_ENDPOINT_URL_DYNAMODB`.
4. The values provided by the `AWS_USE_DUALSTACK_ENDPOINT`, `AWS_USE_FIPS_ENDPOINT`, and `AWS_ENDPOINT_URL` environment variables.
5. The service-specific endpoint value provided by the `endpoint_url` setting within a services section of the shared config file.
6. The value provided by the `endpoint_url` setting within a profile of the shared config file.
7. `use_dualstack_endpoint`, `use_fips_endpoint`, and `endpoint_url` settings.
8. Any default endpoint URL for the respective AWS service is used last. For a list of the standard service endpoints available in each Region, see [AWS Regions and Endpoints](#) in the *Amazon Web Services General Reference*.


AWS_WEB_IDENTITY_TOKEN_FILE

Specifies the path to a file that contains an OAuth 2.0 access token or OpenID Connect ID token that is provided by an identity provider. The AWS CLI loads the contents of this file and passes it as the `WebIdentityToken` argument to the `AssumeRoleWithWebIdentity` operation.

Used with the `AWS_ROLE_ARN` and `AWS_ROLE_SESSION_NAME` environment variables.

If defined, this environment variable overrides the value for the profile setting `web_identity_token_file`.

For more information on using web identities, see [the section called “Assume role with web identity”](#).

 **Note**

This environment variable only applies to an assumed role with web identity provider it does not apply to the general assume role provider configuration.

Command line options

In the AWS CLI, command line options are global parameters you can use to override the default configuration settings, any corresponding profile setting, or environment variable setting for that single command. You can't use command line options to directly specify credentials, although you can specify which profile to use.

Topics

- [How to use command line options](#)
- [AWS CLI supported global command line options](#)
- [Common uses of command line options](#)

How to use command line options

Most command line options are simple strings, such as the profile name `profile1` in the following example:

```
$ aws s3 ls --profile profile1
example-bucket-1
example-bucket-2
...
```

Each option that takes an argument requires a space or equals sign (=) separating the argument from the option name. If the argument value is a string that contains a space, you must use quotation marks around the argument. For details on argument types and formatting for parameters, see [Specify parameter values for the AWS CLI](#).

AWS CLI supported global command line options

In the AWS CLI you can use the following command line options to override the default configuration settings, any corresponding profile setting, or environment variable setting for that single command.

--ca-bundle *<string>*

Specifies the certificate authority (CA) certificate bundle to use when verifying SSL certificates.

If defined, this option overrides the value for the profile setting [ca_bundle](#) and the [AWS_CA_BUNDLE](#) environment variable.

--cli-auto-prompt

Enables auto-prompt mode for a single command. As the following examples show, you can specify it at any point.

```
$ aws --cli-auto-prompt
$ aws dynamodb --cli-auto-prompt
$ aws dynamodb describe-table --cli-auto-prompt
```

This option overrides the [aws_cli_auto_prompt](#) environment variable and the [cli_auto_prompt](#) profile setting.

For information on the AWS CLI version 2 auto-prompt feature, see [Have the AWS CLI prompt you for commands](#).

--cli-binary-format

Specifies how the AWS CLI version 2 interprets binary input parameters. It can be one of the following values:

- **base64** – This is the default value. An input parameter that is typed as a binary large object (BLOB) accepts a base64-encoded string. To pass true binary content, put the content in a file and provide the file's path and name with the `fileb://` prefix as the parameter's value. To pass base64-encoded text contained in a file, provide the file's path and name with the `file://` prefix as the parameter's value.
- **raw-in-base64-out** – Default for the AWS CLI version 1. If the setting's value is `raw-in-base64-out`, files referenced using the `file://` prefix is read as text and then the AWS CLI attempts to encode it to binary.

This overrides the [cli_binary_format](#) file configuration setting.

```
$ aws lambda invoke \  
  --cli-binary-format raw-in-base64-out \  
  --function-name my-function \  
  --invocation-type Event \  
  --payload '{ "name": "Bob" }' \  
  response.json
```

If you reference a binary value in a file using the `fileb://` prefix notation, the AWS CLI *always* expects the file to contain raw binary content and does not attempt to convert the value.

If you reference a binary value in a file using the `file://` prefix notation, the AWS CLI handles the file according to the current `cli_binary_format` setting. If that setting's value is `base64` (the default when not explicitly set), the AWS CLI expects the file to contain base64-encoded text. If that setting's value is `raw-in-base64-out`, the AWS CLI expects the file to contain raw binary content.

--cli-connect-timeout *<integer>*

Specifies the maximum socket connect time in seconds. If the value is set to zero (0), the socket connect waits indefinitely (is blocking) and doesn't timeout.

--cli-read-timeout *<integer>*

Specifies the maximum socket read time in seconds. If the value is set to zero (0) the socket read waits indefinitely (is blocking) and doesn't timeout.

--color *<string>*

Specifies support for color output. Valid values are `on`, `off`, and `auto`. The default value is `auto`.

--debug

A Boolean switch that enables debug logging. The AWS CLI by default provides cleaned up information regarding any successes or failures regarding command outcomes in the command output. The `--debug` option provides the full Python logs. This includes additional `stderr` diagnostic information about the operation of the command that can be useful when troubleshooting why a command provides unexpected results. To easily view debug logs, we suggest sending the logs to a file to more easily search the information. You can do this by using one of the following.

To send **only** the `stderr` diagnostic information, append `2> debug.txt` where `debug.txt` is the name you want to use for your debug file:

```
$ aws servicename commandname options --debug 2> debug.txt
```

To send **both** the output and `stderr` diagnostic information, append `&> debug.txt` where `debug.txt` is the name you want to use for your debug file:

```
$ aws servicename commandname options --debug &> debug.txt
```

--endpoint-url *<string>*

Specifies the URL to send the request to. For most commands, the AWS CLI automatically determines the URL based on the selected service and the specified AWS Region. However, some commands require that you specify an account-specific URL. You can also configure some AWS services to [host an endpoint directly within your private VPC](#), which might then need to be specified.

The following command example uses a custom Amazon S3 endpoint URL.

```
$ aws s3 ls --endpoint-url http://localhost:4567
```

Endpoint configuration settings are located in multiple places, such as the system or user environment variables, local AWS configuration files, or explicitly declared on the command line as a parameter. The AWS CLI endpoint configuration settings take precedence in the following order:

1. The `--endpoint-url` command line option.
2. If enabled, the `AWS_IGNORE_CONFIGURED_ENDPOINT_URLS` global endpoint environment variable or profile setting `ignore_configure_endpoint_urls` to ignore custom endpoints.
3. The value provided by a service-specific environment variable `AWS_ENDPOINT_URL_<SERVICE>`, such as `AWS_ENDPOINT_URL_DYNAMODB`.
4. The values provided by the `AWS_USE_DUALSTACK_ENDPOINT`, `AWS_USE_FIPS_ENDPOINT`, and `AWS_ENDPOINT_URL` environment variables.
5. The service-specific endpoint value provided by the `endpoint_url` setting within a services section of the shared config file.

6. The value provided by the [endpoint_url](#) setting within a profile of the shared config file.
7. [use_dualstack_endpoint](#), [use_fips_endpoint](#), and [endpoint_url](#) settings.
8. Any default endpoint URL for the respective AWS service is used last. For a list of the standard service endpoints available in each Region, see [AWS Regions and Endpoints](#) in the *Amazon Web Services General Reference*.

--no-cli-auto-prompt

Disables auto-prompt mode for a single command.

```
$ aws dynamodb describe-table --table-name Table1 --no-cli-auto-prompt
```

This option overrides the [aws_cli_auto_prompt](#) environment variable and the [cli_auto_prompt](#) profile setting.

For information on the AWS CLI version 2 auto-prompt feature, see [Have the AWS CLI prompt you for commands](#).

--no-cli-pager

A Boolean switch that disables using a pager for the output of the command.

--no-paginate

A Boolean switch that disables the multiple calls the automatically AWS CLI makes to receive all command results that creates pagination of the output. This means only the first page of your output is displayed.

--no-sign-request

A Boolean switch that disables signing the HTTP requests to the AWS service endpoint. This prevents credentials from being loaded.

--no-verify-ssl

By default, the AWS CLI uses SSL when communicating with AWS services. For each SSL connection and call, the AWS CLI verifies the SSL certificates. Using this option overrides the default behavior of verifying SSL certificates.

Warning

This option is **not** best practice. If you use `--no-verify-ssl`, your traffic between your client and AWS services is no longer secured. This means your traffic is a security

risk and vulnerable to man-in-the-middle exploits. If you're having issues with certificates, it's best to resolve those issues instead. For certificate troubleshooting steps, see [the section called "SSL certificate errors"](#).

--output *<string>*

Specifies the output format to use for this command. You can specify any of the following values:

- **json** – The output is formatted as a [JSON](#) string.
- **yaml** – The output is formatted as a [YAML](#) string.
- **yaml-stream** – The output is streamed and formatted as a [YAML](#) string. Streaming allows for faster handling of large data types.
- **text** – The output is formatted as multiple lines of tab-separated string values. This can be useful to pass the output to a text processor, like `grep`, `sed`, or `awk`.
- **table** – The output is formatted as a table using the characters `+|-` to form the cell borders. It typically presents the information in a "human-friendly" format that is much easier to read than the others, but not as programmatically useful.

--profile *<string>*

Specifies the [named profile](#) to use for this command. To set up additional named profiles, you can use the `aws configure` command with the `--profile` option.

```
$ aws configure --profile <profilename>
```

--query *<string>*

Specifies a [JMESPath query](#) to use in filtering the response data. For more information, see [Filter AWS CLI output](#).

--region *<string>*

Specifies which AWS Region to send this command's AWS request to. For a list of all of the Regions that you can specify, see [AWS Regions and Endpoints](#) in the *Amazon Web Services General Reference*.

--version

A Boolean switch that displays the current version of the AWS CLI program that is running.

Common uses of command line options

Common uses for command line options include checking your resources in multiple AWS Regions, and changing the output format for legibility or ease of use when scripting. In the following examples, we run the **describe-instances** command against each Region until we find which Region our instance is in.

```
$ aws ec2 describe-instances --output table --region us-west-1
-----
|DescribeInstances|
+-----+
$ aws ec2 describe-instances --output table --region us-west-2
-----
|
|          DescribeInstances          |
+-----+
||
||          Reservations              ||
|+-----+|
||  OwnerId          | 012345678901    ||
||  ReservationId   | r-abcdefgh    ||
|+-----+|
||| | | | |
|||          Instances                |||
||+-----+||
|||  AmiLaunchIndex | 0                |||
|||  Architecture   | x86_64          |||
...

```

Command completion

The AWS Command Line Interface (AWS CLI) includes a bash-compatible command-completion feature that enables you to use the **Tab** key to complete a partially entered command. On most systems you need to configure this manually.

For information on the AWS CLI version 2 auto-prompt feature instead, see [Have the AWS CLI prompt you for commands](#).

Topics

- [How it works](#)
- [Configuring command completion on Linux or macOS](#)
- [Configuring command completion on Windows](#)

How it works

When you partially enter a command, parameter, or option, the command-completion feature either automatically completes your command or displays a suggested list of commands. To prompt command completion, you partially enter in a command and press the completion key, which is typically *Tab* in most shells.

The following examples show different ways that you can use command completion:

- Partially enter a command and press *Tab* to display a suggested list of commands.

```
$ aws dynamodb dTAB
delete-backup                describe-global-table
delete-item                  describe-global-table-settings
delete-table                 describe-limits
describe-backup              describe-table
describe-continuous-backups describe-table-replica-auto-scaling
describe-contributor-insights describe-time-to-live
describe-endpoints
```

- Partially enter a parameter and press *Tab* to display a suggested list of parameters.

```
$ aws dynamodb delete-table --TAB
--ca-bundle                --endpoint-url            --profile
--cli-connect-timeout      --generate-cli-skeleton  --query
--cli-input-json           --no-paginate             --region
--cli-read-timeout         --no-sign-request        --table-name
--color                    --no-verify-ssl          --version
--debug                    --output
```

- Enter a parameter and press *Tab* to display a suggested list of resource values. This feature is available only in the AWS CLI version 2.

```
$ aws dynamodb db delete-table --table-name TAB
Table 1                Table 2                Table 3
```

Configuring command completion on Linux or macOS

To configure command completion on Linux or macOS, you must know the name of the shell you're using and the location of the `aws_completer` script.

Note

Command completion is automatically configured and enabled by default on Amazon EC2 instances that run Amazon Linux.

Topics

- [Confirm the completer's folder is in your path](#)
- [Enable command completion](#)
- [Verify command completion](#)

Confirm the completer's folder is in your path

For the AWS completer to work successfully, the `aws_completer` needs to be in your shell's path. The `which` command can check if the completer is in your path.

```
$ which aws_completer
/usr/local/bin/aws_completer
```

If the `which` command can't find the completer, then use the following steps to add the completer's folder to your path.

Step 1: Locate the AWS completer

The location of the AWS completer can vary depending on the installation method used.

- **Package Manager** - Programs such as `pip`, `yum`, `brew`, and `apt-get` typically install the AWS completer (or a symlink to it) to a standard path location.
 - If you used `pip` **without** the `--user` parameter, the default path is `/usr/local/bin/aws_completer`.
 - If you used `pip` **with** the `--user` parameter the default path is `/home/username/.local/bin/aws_completer`.
- **Bundled Installer** - If you used the bundled installer, the default path is `/usr/local/bin/aws_completer`.

If all else fails, you can use the `find` command to search your file system for the AWS completer.

```
$ find / -name aws_completer
/usr/local/bin/aws_completer
```

Step 2: Identify your shell

To identify which shell you're using, you can use one of the following commands.

- **echo \$SHELL** – Displays the shell's program file name. This usually matches the name of the in-use shell, unless you launched a different shell after logging in.

```
$ echo $SHELL
/bin/bash
```

- **ps** – Displays the processes running for the current user. One of them is the shell.

```
$ ps
  PID TTY          TIME CMD
 2148 pts/1    00:00:00 bash
 8756 pts/1    00:00:00 ps
```

Step 3: Add the completer to your path

1. Find your shell's profile script in your user folder.

```
$ ls -a ~/
.  ..  .bash_logout  .bash_profile  .bashrc  Desktop  Documents  Downloads
```

- **Bash**– `.bash_profile`, `.profile`, or `.bash_login`
 - **Zsh**– `.zshrc`
 - **Tcsh**– `.tcshrc`, `.cshrc`, or `.login`
2. Add an export command at the end of your profile script that's similar to the following example. Replace `/usr/local/bin/` with the folder that you discovered in the previous section.

```
export PATH=/usr/local/bin/:$PATH
```

3. Reload the profile into the current session to put those changes into effect. Replace `.bash_profile` with the name of the shell script you discovered in the first section.

```
$ source ~/.bash_profile
```

Enable command completion

After confirming the completer is in your path, enable command completion by running the appropriate command for the shell that you're using. You can add the command to your shell's profile to run it each time you open a new shell. In each command, replace the `/usr/local/bin/` path with the one found on your system in [Confirm the completer's folder is in your path](#).

- **bash** – Use the built-in command complete.

```
$ complete -C '/usr/local/bin/aws_completer' aws
```

Add the previous command to `~/.bashrc` to run it each time you open a new shell. Your `~/.bash_profile` should source `~/.bashrc` to ensure that the command is also run in login shells.

- **zsh** – To run command completion, you need to run `bashcompinit` by adding the following autoload line at the end of your `~/.zshrc` profile script.

```
$ autoload bashcompinit && bashcompinit  
$ autoload -Uz compinit && compinit
```

To enable command completion, use the built-in command complete.

```
$ complete -C '/usr/local/bin/aws_completer' aws
```

Add the previous commands to `~/.zshrc` to run it each time you open a new shell.

- **tcsh** – Complete for `tcsh` takes a word type and pattern to define the completion behavior.

```
> complete aws 'p/*/'aws_completer`/'
```

Add the previous command to `~/.tcshrc` to run it each time you open a new shell.

After you've enabled command completion, [Verify command completion](#) is working.

Verify command completion

After enabling command completion, reload your shell, enter a partial command, and press **Tab** to see the available commands.

```
$ aws sTAB
s3          ses          sqs          sts          swf
s3api       sns          storagegateway support
```

Configuring command completion on Windows

Note

For information on how PowerShell handles their completion, including their various completion keys, see [about_Tab_Expansion](#) in the *Microsoft PowerShell Docs*.

To enable command completion for PowerShell on Windows, complete the following steps in PowerShell.

1. Open your \$PROFILE with the following command.

```
PS C:\> Notepad $PROFILE
```

If you do not have a \$PROFILE, create a user profile using the following command.

```
PS C:\> if (!(Test-Path -Path $PROFILE ))
{ New-Item -Type File -Path $PROFILE -Force }
```

For more information on PowerShell profiles, see [How to Use Profiles in Windows PowerShell ISE](#) on the *Microsoft Docs* website.

2. To enable command completion, add the following code block to your profile, save, and then close the file.

```
Register-ArgumentCompleter -Native -CommandName aws -ScriptBlock {
    param($commandName, $wordToComplete, $cursorPosition)
    $env:COMP_LINE=$wordToComplete
    if ($env:COMP_LINE.Length -lt $cursorPosition){
        $env:COMP_LINE=$env:COMP_LINE + " "
```

```

    }
    $env:COMP_POINT=$cursorPosition
    aws_completer.exe | ForEach-Object {
        [System.Management.Automation.CompletionResult]::new($_, $_,
'ParameterValue', $_)
    }
    Remove-Item Env:\COMP_LINE
    Remove-Item Env:\COMP_POINT
}

```

3. After enabling command completion, reload your shell, enter a partial command, and press **Tab** to cycle through the available commands.

```
$ aws sTab
```

```
$ aws s3
```

To see all available commands available to your completion, enter a partial command and press **Ctrl + Space**.

```
$ aws sCtrl + Space
s3          ses          sqs          sts          swf
s3api       sns          storagegateway support
```

AWS CLI retries

This topic describes how the AWS CLI might see calls to AWS services fail due to unexpected issues. These issues can occur on the server side or might fail due to rate limiting from the AWS service you're attempting to call. These kinds of failures usually don't require special handling and the call is automatically made again, often after a brief waiting period. The AWS CLI provides many features to assist in retrying client calls to AWS services when these kinds of errors or exceptions are experienced.

Topics

- [Available retry modes](#)
- [Configuring a retry mode](#)
- [Viewing logs of retry attempts](#)

Available retry modes

The AWS CLI has multiple modes to choose from depending on your version:

- [Legacy retry mode](#)
- [Standard retry mode](#)
- [Adaptive retry mode](#)

Legacy retry mode

Legacy mode uses an older retry handler that has limited functionality which includes:

- A default value of 4 for maximum retry attempts, making a total of 5 call attempts. This value can be overwritten through the `max_attempts` configuration parameter.
- DynamoDB has a default value of 9 for maximum retry attempts, making a total of 10 call attempts. This value can be overwritten through the `max_attempts` configuration parameter.
- Retry attempts for the following limited number of errors/exceptions:
 - General socket/connection errors:
 - `ConnectionError`
 - `ConnectionClosedError`
 - `ReadTimeoutError`
 - `EndpointConnectionError`
 - Service-side throttling/limit errors and exceptions:
 - `Throttling`
 - `ThrottlingException`
 - `ThrottledException`
 - `RequestThrottledException`
 - `ProvisionedThroughputExceededException`
- Retry attempts on several HTTP status codes, including 429, 500, 502, 503, 504, and 509.
- Any retry attempt will include an exponential backoff by a base factor of 2.

Standard retry mode

Standard mode is a standard set of retry rules across the AWS SDKs with more functionality than legacy. This mode is the default for AWS CLI version 2. Standard mode was created for the AWS CLI version 2 and is backported to AWS CLI version 1. Standard mode's functionality includes:

- A default value of 2 for maximum retry attempts, making a total of 3 call attempts. This value can be overwritten through the `max_attempts` configuration parameter.
- Retry attempts for the following expanded list of errors/exceptions:
 - Transient errors/exceptions
 - RequestTimeout
 - RequestTimeoutException
 - PriorRequestNotComplete
 - ConnectionError
 - HTTPClientError
 - Service-side throttling/limit errors and exceptions:
 - Throttling
 - ThrottlingException
 - ThrottledException
 - RequestThrottledException
 - TooManyRequestsException
 - ProvisionedThroughputExceededException
 - TransactionInProgressException
 - RequestLimitExceeded
 - BandwidthLimitExceeded
 - LimitExceededException
 - RequestThrottled
 - SlowDown
 - EC2ThrottledException
- Retry attempts on nondescriptive, transient error codes. Specifically, these HTTP status codes:
500, 502, 503, 504.

- Any retry attempt will include an exponential backoff by a base factor of 2 for a maximum backoff time of 20 seconds.

Adaptive retry mode

Warning

Adaptive mode is an experimental mode and is subject to change, both in features and behavior.

Adaptive retry mode is an experimental retry mode that includes all the features of standard mode. In addition to the standard mode features, adaptive mode also introduces client-side rate limiting through the use of a token bucket and rate-limit variables that are dynamically updated with each retry attempt. This mode offers flexibility in client-side retries that adapts to the error/exception state response from an AWS service.

With each new retry attempt, adaptive mode modifies the rate-limit variables based on the error, exception, or HTTP status code presented in the response from the AWS service. These rate-limit variables are then used to calculate a new call rate for the client. Each exception/error or non-success HTTP response (provided in the list above) from an AWS service updates the rate-limit variables as retries occur until success is reached, the token bucket is exhausted, or the configured maximum attempts value is reached.

Configuring a retry mode

The AWS CLI includes a variety of both retry configurations as well as configuration methods to consider when creating your client object.

Available configuration methods

In the AWS CLI, users can configure retries in the following ways:

- Environment variables
- AWS CLI configuration file

Users can customize the following retry options:

- **Retry mode** - Specifies which retry mode the AWS CLI uses. As described previously, there are three retry modes available: legacy, standard, and adaptive. The default value for the AWS CLI version 2 is standard.
- **Max attempts** - Specifies the value of maximum retry attempts the AWS CLI retry handler uses, where the initial call counts toward the value that you provide. The default value is 5.

Defining a retry configuration in your environment variables

To define your retry configuration for the AWS CLI, update your operating system's environment variables.

The retry environment variables are:

- `AWS_RETRY_MODE`
- `AWS_MAX_ATTEMPTS`

For more information on environment variables, see [Environment variables to configure the AWS CLI](#).

Defining a retry configuration in your AWS configuration file

To change your retry configuration, update your global AWS configuration file. The default location for your AWS config file is `~/.aws/config`.

The following is an example of an AWS config file:

```
[default]
retry_mode = standard
max_attempts = 6
```

For more information on configuration files, see [Configuration and credential file settings](#).

Viewing logs of retry attempts

The AWS CLI uses Boto3's retry methodology and logging. You can use the `--debug` option on any command to receive debug logs. For more information on how to use the `--debug` option, see [Command line options](#).

If you search for "retry" in your debug logs, you'll find the retry information you need. The client log entries for retry attempts depend on which retry mode you've enabled.

Legacy mode:

Retry messages are generated by `botocore.retryhandler`. You'll see one of three messages:

- No retry needed
- Retry needed, action of: `<action_name>`
- Reached the maximum number of retry attempts: `<attempt_number>`

Standard or adaptive mode:

Retry messages are generated by `botocore.retries.standard`. You'll see one of three messages:

- No retrying request
- Retry needed, retrying request after delay of: `<delay_value>`
- Retry needed but retry quota reached, not retrying request

For the full definition file of botocore retries, see [_retry.json](#) on the *botocore GitHub Repository*.

Use an HTTP proxy

To access AWS through proxy servers, you can configure the `HTTP_PROXY` and `HTTPS_PROXY` environment variables with either the DNS domain names or IP addresses and port numbers that your proxy servers use.

Topics

- [Using the examples](#)
- [Authenticating to a proxy](#)
- [Using a proxy on Amazon EC2 instances](#)
- [Troubleshooting](#)

Using the examples

Note

The following examples show the environment variable name in all uppercase letters. However, if you specify a variable twice using different cases, the lowercase letters take precedence. We recommend that you define each variable only once to avoid system confusion and unexpected behavior.

The following examples show how you can use either the explicit IP address of your proxy or a DNS name that resolves to the IP address of your proxy. Either can be followed by a colon and the port number to which queries should be sent.

Linux or macOS

```
$ export HTTP_PROXY=http://10.15.20.25:1234
$ export HTTP_PROXY=http://proxy.example.com:1234
$ export HTTPS_PROXY=http://10.15.20.25:5678
$ export HTTPS_PROXY=http://proxy.example.com:5678
```

Windows Command Prompt

To set for all sessions

```
C:\> setx HTTP_PROXY http://10.15.20.25:1234
C:\> setx HTTP_PROXY http://proxy.example.com:1234
C:\> setx HTTPS_PROXY http://10.15.20.25:5678
C:\> setx HTTPS_PROXY http://proxy.example.com:5678
```

Using [setx](#) to set an environment variable changes the value used in both the current command prompt session and all command prompt sessions that you create after running the command. It does **not** affect other command shells that are already running at the time you run the command.

To set for current session only

Using [set](#) to set an environment variable changes the value used until the end of the current command prompt session, or until you set the variable to a different value.

```
C:\> set HTTP_PROXY=http://10.15.20.25:1234
C:\> set HTTP_PROXY=http://proxy.example.com:1234
C:\> set HTTPS_PROXY=http://10.15.20.25:5678
C:\> set HTTPS_PROXY=http://proxy.example.com:5678
```

Authenticating to a proxy

Note

The AWS CLI doesn't support NTLM proxies. If you use an NTLM or Kerberos protocol proxy, you might be able to connect through an authentication proxy like [Cntlm](#).

The AWS CLI supports HTTP Basic authentication. Specify the username and password in the proxy URL, as follows.

Linux or macOS

```
$ export HTTP_PROXY=http://username:password@proxy.example.com:1234
$ export HTTPS_PROXY=http://username:password@proxy.example.com:5678
```

Windows Command Prompt

To set for all sessions

```
C:\> setx HTTP_PROXY http://username:password@proxy.example.com:1234
C:\> setx HTTPS_PROXY http://username:password@proxy.example.com:5678
```

To set for current session only

```
C:\> set HTTP_PROXY=http://username:password@proxy.example.com:1234
C:\> set HTTPS_PROXY=http://username:password@proxy.example.com:5678
```

Using a proxy on Amazon EC2 instances

If you configure a proxy on an Amazon EC2 instance launched with an attached IAM role, ensure that you exempt the address used to access the [instance metadata](#). To do this, set the NO_PROXY

environment variable to the IP address of the instance metadata service, 169.254.169.254. This address does not vary.

Linux or macOS

```
$ export NO_PROXY=169.254.169.254
```

Windows Command Prompt

To set for all sessions

```
C:\> setx NO_PROXY 169.254.169.254
```

To set for current session only

```
C:\> set NO_PROXY=169.254.169.254
```

Troubleshooting

If you come across issues with the AWS CLI, see [Troubleshoot errors](#) for troubleshooting steps. For the most relevant troubleshooting steps, see [the section called "SSL certificate errors"](#).

Use endpoints in the AWS CLI

To connect programmatically to an AWS service, you use an endpoint. An *endpoint* is the URL of the entry point for an AWS web service. The AWS Command Line Interface (AWS CLI) automatically uses the default endpoint for each service in an AWS Region, but you can specify an alternate endpoint for your API requests.

Endpoint topics

- [Set endpoint for a single command](#)
- [Set global endpoint for all AWS services](#)
- [Set to use FIPs endpoints for all AWS services](#)
- [Set to use dual-stack endpoints for all AWS services](#)
- [Set service-specific endpoints](#)
 - [Service-specific endpoints: Environment variables](#)

- [Service-specific endpoints: Shared config file](#)
- [Service-specific endpoints: List of service-specific identifiers](#)
- [Endpoint configuration and settings precedence](#)

Set endpoint for a single command

To override any endpoint settings or environment variables for a single command, use the `--endpoint-url` command line option. The following command example uses a custom Amazon S3 endpoint URL.

```
$ aws s3 ls --endpoint-url http://localhost:4567
```

Set global endpoint for all AWS services

To route requests for all services to a custom endpoint URL, use one of the following settings:

- Environment variables:
 - [AWS_IGNORE_CONFIGURED_ENDPOINT_URLS](#) - Ignore configured endpoint URLs.

Linux or macOS

```
$ export AWS_IGNORE_CONFIGURED_ENDPOINT_URLS=true
```

Windows Command Prompt

To set for all sessions

```
C:\> setx AWS_IGNORE_CONFIGURED_ENDPOINT_URLS true
```

To set for current session only

```
C:\> set AWS_IGNORE_CONFIGURED_ENDPOINT_URLS=true
```

PowerShell

```
PS C:\> $Env:AWS_IGNORE_CONFIGURED_ENDPOINT_URLS="true"
```

- [AWS_ENDPOINT_URL](#) - Set global endpoint URL.

Linux or macOS

```
$ export AWS_ENDPOINT_URL=http://localhost:4567
```

Windows Command Prompt

To set for all sessions

```
C:\> setx AWS_ENDPOINT_URL http://localhost:4567
```

To set for current session only

```
C:\> set AWS_ENDPOINT_URL=http://localhost:4567
```

PowerShell

```
PS C:\> $Env:AWS_ENDPOINT_URL="http://localhost:4567"
```

- The config file:
 - [ignore_configure_endpoint_urls](#) - Ignore configured endpoint URLs.

```
ignore_configure_endpoint_urls = true
```

- [endpoint_url](#) - Set global endpoint URL.

```
endpoint_url = http://localhost:4567
```

Service-specific endpoints and the `--endpoint-url` command line option override any global endpoints.

Set to use FIPs endpoints for all AWS services

To route requests for all services to use FIPs endpoints, use one of the following:

- [AWS_USE_FIPS_ENDPOINT](#) environment variable.

Linux or macOS

```
$ export AWS_USE_FIPS_ENDPOINT=true
```

Windows Command Prompt

To set for all sessions

```
C:\> setx AWS_USE_FIPS_ENDPOINT true
```

To set for current session only

```
C:\> set AWS_USE_FIPS_ENDPOINT=true
```

PowerShell

```
PS C:\> $Env:AWS_USE_FIPS_ENDPOINT="true"
```

- [use_fips_endpoint](#) file setting.

```
use_fips_endpoint = true
```

Some AWS services offer endpoints that support [Federal Information Processing Standard \(FIPS\) 140-2](#) in some AWS Regions. When the AWS service supports FIPS, this setting specifies what FIPS endpoint the AWS CLI should use. Unlike standard AWS endpoints, FIPS endpoints use a TLS software library that complies with FIPS 140-2. These endpoints might be required by enterprises that interact with the United States government.

If this setting is enabled, but a FIPS endpoint does not exist for the service in your AWS Region, the AWS command may fail. In this case, manually specify the endpoint to use in the command using the [--endpoint-url](#) option or use [service-specific endpoints](#).

For more information on specifying FIPS endpoints by AWS Region, see [FIPS Endpoints by Service](#).

Set to use dual-stack endpoints for all AWS services

To route requests for all services to use dual-stack endpoints when available, use one of the following settings:

- [AWS_USE_DUALSTACK_ENDPOINT](#) environment variable.

Linux or macOS

```
$ export AWS_USE_DUALSTACK_ENDPOINT=true
```

Windows Command Prompt

To set for all sessions

```
C:\> setx AWS_USE_DUALSTACK_ENDPOINT true
```

To set for current session only

```
C:\> set AWS_USE_DUALSTACK_ENDPOINT=true
```

PowerShell

```
PS C:\> $Env:AWS_USE_DUALSTACK_ENDPOINT="true"
```

- [use_dualstack_endpoint](#) file setting.

```
use_dualstack_endpoint = true
```

Enables the use of dual-stack endpoints to send AWS requests. To learn more about dual-stack endpoints, which support both IPv4 and IPv6 traffic, see [Using Amazon S3 dual-stack endpoints](#) in the *Amazon Simple Storage Service User Guide*. Dual-stack endpoints are available for some services in some regions. If a dual-stack endpoint does not exist for the service or AWS Region, the request fails. This is disabled by default.

Set service-specific endpoints

Service-specific endpoint configuration provides the option to use a persistent endpoint of your choosing for AWS CLI requests. These settings provide flexibility to support local endpoints, VPC endpoints, and third-party local AWS development environments. Different endpoints can be used for testing and production environments. You can specify an endpoint URL for individual AWS services.

Service-specific endpoints can be specified in the following ways:

- The command line option [--endpoint-url](#) for a single command.
- Environment variables:
 - [AWS_IGNORE_CONFIGURED_ENDPOINT_URLS](#) - Ignore all configured endpoint URLs, unless specified on the command line.
 - [AWS_ENDPOINT_URL_<SERVICE>](#) - Specifies a custom endpoint that is used for a specific service, where <SERVICE> is replace with the AWS service identifier. For all service-specific variables, see [the section called "List of service-specific identifiers"](#).
- config file:
 - [ignore_configure_endpoint_urls](#) - Ignore all configured endpoint URLs, unless specified using environment variables or on the command line.
 - The [services](#) section of the config file combined with the [endpoint_url](#) file setting.

Service-specific endpoints topics:

- [Service-specific endpoints: Environment variables](#)
- [Service-specific endpoints: Shared config file](#)
- [Service-specific endpoints: List of service-specific identifiers](#)

Service-specific endpoints: Environment variables

Environment variables override settings in your config file, but do not override options specified on the command line. Use environment variables if you want all profiles to use the same endpoints on your device.

The following are service-specific environment variables:

- [AWS_IGNORE_CONFIGURED_ENDPOINT_URLS](#) - Ignore all configured endpoint URLs, unless specified on the command line.

Linux or macOS

```
$ export AWS_IGNORE_CONFIGURED_ENDPOINT_URLS=true
```

Windows Command Prompt

To set for all sessions

```
C:\> setx AWS_IGNORE_CONFIGURED_ENDPOINT_URLS true
```

To set for current session only

```
C:\> set AWS_IGNORE_CONFIGURED_ENDPOINT_URLS=true
```

PowerShell

```
PS C:\> $Env:AWS_IGNORE_CONFIGURED_ENDPOINT_URLS="true"
```

- [AWS_ENDPOINT_URL_<SERVICE>](#) - Specifies a custom endpoint that is used for a specific service, where <SERVICE> is replaced with the AWS service identifier. For all service-specific variables, see [the section called "List of service-specific identifiers"](#).

The following environment variable examples sets an endpoint for AWS Elastic Beanstalk:

Linux or macOS

```
$ export AWS_ENDPOINT_URL_ELASTIC_BEANSTALK=http://localhost:4567
```

Windows Command Prompt

To set for all sessions

```
C:\> setx AWS_ENDPOINT_URL_ELASTIC_BEANSTALK http://localhost:4567
```

To set for current session only

```
C:\> set AWS_ENDPOINT_URL_ELASTIC_BEANSTALK=http://localhost:4567
```

PowerShell

```
PS C:\> $Env:AWS_ENDPOINT_URL_ELASTIC_BEANSTALK="http://localhost:4567"
```

For more information on setting environment variables, see [the section called "Environment Variables"](#).

Service-specific endpoints: Shared config file

In the shared config file, `endpoint_url` is used in multiple sections. To set a service-specific endpoint, use the `endpoint_url` setting nested under a service identifier key within a `services` section. For details on defining a `services` section in your shared config file, see [the section called "services"](#).

The following example uses a `services` section to configure a service-specific endpoint URL for Amazon S3 and a custom global endpoint used for all other services:

```
[profile dev1]
endpoint_url = http://localhost:1234
services = s3-specific

[services testing-s3]
s3 =
  endpoint_url = http://localhost:4567
```

A single profile can configure endpoints for multiple services. The following example sets the service-specific endpoint URLs for Amazon S3 and AWS Elastic Beanstalk in the same profile.

For a list of all service identifier keys to use in the `services` section, see [List of service-specific identifiers](#).

```
[profile dev1]
services = testing-s3-and-eb

[services testing-s3-and-eb]
s3 =
  endpoint_url = http://localhost:4567
elastic_beanstalk =
  endpoint_url = http://localhost:8000
```

The service configuration section can be used in multiple profiles. The following example has two profiles use the same `services` definition:

```
[profile dev1]
output = json
services = testing-s3

[profile dev2]
```

```
output = text
services = testing-s3

[services testing-s3]
s3 =
  endpoint_url = https://localhost:4567
```

Service-specific endpoints: List of service-specific identifiers

The AWS service identifier is based on the API model's `serviceId` by replacing all spaces with underscores and lowercasing all letters.

The following service identifier example uses AWS Elastic Beanstalk. AWS Elastic Beanstalk has a `serviceId` of [Elastic Beanstalk](#), therefore the service identifier key is `elastic_beanstalk`.

The following table lists all service-specific identifiers, config file keys, and environment variables.

<code>serviceId</code>	Service identifier key for the config file	environment variable
AccessAnalyzer	<code>AWS_ENDPOINT_URL_ACCESSANALYZER</code>	
Account	<code>AWS_ENDPOINT_URL_ACCOUNT</code>	
ACM	<code>AWS_ENDPOINT_URL_ACM</code>	
ACM PCA	<code>AWS_ENDPOINT_URL_ACM_PCA</code>	
Alexa For Business	<code>AWS_ENDPOINT_URL_ALEXA_FOR_BUSINESS</code>	

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
amp	awscli AWS_ENDPOINT_URL_AMP
Amplify	awscli AWS_ENDPOINT_URL_AMPLIFY
AmplifyBackend	awscli AWS_ENDPOINT_URL_AMPLIFYBACKEND
AmplifyUIBuilder	awscli AWS_ENDPOINT_URL_AMPLIFYUIBUILDER
API Gateway	awscli AWS_ENDPOINT_URL_API_GATEWAY
ApiGatewayManagementApi	awscli AWS_ENDPOINT_URL_APIGATEWAYMANAGEMENTAPI
ApiGatewayV2	awscli AWS_ENDPOINT_URL_APIGATEWAYV2
AppConfig	awscli AWS_ENDPOINT_URL_APPCONFIG
AppConfigData	awscli AWS_ENDPOINT_URL_APPCONFIGDATA
AppFabric	awscli AWS_ENDPOINT_URL_APPFABRIC

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
Appflow	awscli AWS_ENDPOINT_URL_APPFLOW
AppIntegrations	awscli AWS_ENDPOINT_URL_APPINTEGRATIONS
Application Auto Scaling	awscli AWS_ENDPOINT_URL_APPLICATION_AUTO_SCALING
Application Insights	awscli AWS_ENDPOINT_URL_APPLICATION_INSIGHTS
ApplicationCostProfiler	awscli AWS_ENDPOINT_URL_APPLICATIONCOSTPROFILER
App Mesh	awscli AWS_ENDPOINT_URL_APP_MESH
AppRunner	awscli AWS_ENDPOINT_URL_APPRUNNER
AppStream	awscli AWS_ENDPOINT_URL_APPSTREAM
AppSync	awscli AWS_ENDPOINT_URL_APPSVC
ARC Zonal Shift	awscli AWS_ENDPOINT_URL_ARC_ZONAL_SHIFT

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
Artifact	aws: AWS_ENDPOINT_URL_ARTIFACT
Athena	aws: AWS_ENDPOINT_URL_ATHENA
AuditManager	aws: AWS_ENDPOINT_URL_AUDITMANAGER
Auto Scaling	aws: AWS_ENDPOINT_URL_AUTO_SCALING
Auto Scaling Plans	aws: AWS_ENDPOINT_URL_AUTO_SCALING_PLANS
b2bi	aws: AWS_ENDPOINT_URL_B2BI
Backup	aws: AWS_ENDPOINT_URL_BACKUP
Backup Gateway	aws: AWS_ENDPOINT_URL_BACKUP_GATEWAY
BackupStorage	aws: AWS_ENDPOINT_URL_BACKUPSTORAGE
Batch	aws: AWS_ENDPOINT_URL_BATCH
BCM Data Exports	aws: AWS_ENDPOINT_URL_BCM_DATA_EXPORTS

serviceId	Service ID	AWS_ENDPOINT_URL_<SERVICE>	environment variable
Bedrock	bedrock	AWS_ENDPOINT_URL_BEDROCK	
Bedrock Agent	bedrock-agent	AWS_ENDPOINT_URL_BEDROCK_AGENT	
Bedrock Agent Runtime	bedrock-agent-runtime	AWS_ENDPOINT_URL_BEDROCK_AGENT_RUNTIME	
Bedrock Runtime	bedrock-runtime	AWS_ENDPOINT_URL_BEDROCK_RUNTIME	
billingconductor	billingconductor	AWS_ENDPOINT_URL_BILLINGCONDUCTOR	
Braket	braket	AWS_ENDPOINT_URL_BRAKET	
Budgets	budgets	AWS_ENDPOINT_URL_BUDGETS	
Cost Explorer	cost-explorer	AWS_ENDPOINT_URL_COST_EXPLORER	
chatbot	chatbot	AWS_ENDPOINT_URL_CHATBOT	
Chime	chime	AWS_ENDPOINT_URL_CHIME	
Chime SDK Identity	chime-sdk-identity	AWS_ENDPOINT_URL_CHIME_SDK_IDENTITY	

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
Chime SDK Media Pipelines	cli AWS_ENDPOINT_URL_CHIME_SDK_MEDIA_PIPELINES _r ke fo sh Al co fil
Chime SDK Meetings	cli AWS_ENDPOINT_URL_CHIME_SDK_MEETINGS _r
Chime SDK Messaging	cli AWS_ENDPOINT_URL_CHIME_SDK_MESSAGING _r g
Chime SDK Voice	cli AWS_ENDPOINT_URL_CHIME_SDK_VOICE _r
CleanRooms	cli AWS_ENDPOINT_URL_CLEANROOMS s
CleanRoomsML	cli AWS_ENDPOINT_URL_CLEANROOMSML sr
Cloud9	cli AWS_ENDPOINT_URL_CLOUD9
CloudControl	cli AWS_ENDPOINT_URL_CLOUDCONTROL r
CloudDirectory	cli AWS_ENDPOINT_URL_CLOUDDIRECTORY cli

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
CloudFormation	c:\AWS_ENDPOINT_URL_CLOUDFORMATION at
CloudFront	c:\AWS_ENDPOINT_URL_CLOUDFRONT t
CloudFront KeyValuesStore	c:\AWS_ENDPOINT_URL_CLOUDFRONT_KEYVALUESTORE t_ e:
CloudHSM	c:\AWS_ENDPOINT_URL_CLOUDHSM
CloudHSM V2	c:\AWS_ENDPOINT_URL_CLOUDHSM_V2 v:
CloudSearch	c:\AWS_ENDPOINT_URL_CLOUDSEARCH cl
CloudSearch Domain	c:\AWS_ENDPOINT_URL_CLOUDSEARCH_DOMAIN cl
CloudTrail	c:\AWS_ENDPOINT_URL_CLOUDTRAIL l
CloudTrail Data	c:\AWS_ENDPOINT_URL_CLOUDTRAIL_DATA l_

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
CloudWatch	CloudWatch
codeartifact	codeartifact
CodeBuild	CodeBuild
CodeCatalyst	CodeCatalyst
CodeCommit	CodeCommit
CodeDeploy	CodeDeploy
CodeGuru Reviewer	CodeGuru Reviewer
CodeGuru Security	CodeGuru Security
CodeGuruProfiler	CodeGuruProfiler
CodePipeline	CodePipeline

serviceId	Service ID	environment variable
CodeStar	code	AWS_ENDPOINT_URL_CODESTAR
CodeStar connections	code	AWS_ENDPOINT_URL_CODESTAR_CONNECTIONS
codestar notificat ions	code	AWS_ENDPOINT_URL_CODESTAR_NOTIFICATIONS
Cognito Identity	code	AWS_ENDPOINT_URL_COGNITO_IDENTITY
Cognito Identity Provider	code	AWS_ENDPOINT_URL_COGNITO_IDENTITY_PROVIDER
Cognito Sync	code	AWS_ENDPOINT_URL_COGNITO_SYNC
Comprehend	code	AWS_ENDPOINT_URL_COMPREHEND
ComprehendMedical	code	AWS_ENDPOINT_URL_COMPREHENDMEDICAL
Compute Optimizer	code	AWS_ENDPOINT_URL_COMPUTE_OPTIMIZER

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
Config Service	<code>setenv AWS_ENDPOINT_URL_CONFIG_SERVICE</code>
Connect	<code>setenv AWS_ENDPOINT_URL_CONNECT</code>
Connect Contact Lens	<code>setenv AWS_ENDPOINT_URL_CONNECT_CONTACT_LENS</code>
ConnectCampaigns	<code>setenv AWS_ENDPOINT_URL_CONNECTCAMPAIGNS</code>
ConnectCases	<code>setenv AWS_ENDPOINT_URL_CONNECTCASES</code>
ConnectParticipant	<code>setenv AWS_ENDPOINT_URL_CONNECTPARTICIPANT</code>
ControlTower	<code>setenv AWS_ENDPOINT_URL_CONTROLTOWER</code>
Cost Optimization Hub	<code>setenv AWS_ENDPOINT_URL_COST_OPTIMIZATION_HUB</code>

serviceId	Set the AWS_ENDPOINT_URL_<SERVICE> environment variable
Cost and Usage Report Service	<code>costandusage AWS_ENDPOINT_URL_COST_AND_USAGE_REPORT_SERVICE</code>
Customer Profiles	<code>customerprofiles AWS_ENDPOINT_URL_CUSTOMER_PROFILES</code>
DataBrew	<code>databrew AWS_ENDPOINT_URL_DATABREW</code>
DataExchange	<code>dataexchange AWS_ENDPOINT_URL_DATAEXCHANGE</code>
Data Pipeline	<code>datapipeline AWS_ENDPOINT_URL_DATA_PIPELINE</code>
DataSync	<code>datasync AWS_ENDPOINT_URL_DATASYNC</code>
DataZone	<code>datazone AWS_ENDPOINT_URL_DATAZONE</code>
DAX	<code>dax AWS_ENDPOINT_URL_DAX</code>
Detective	<code>detective AWS_ENDPOINT_URL_DETECTIVE</code>
Device Farm	<code>devicefarm AWS_ENDPOINT_URL_DEVICE_FARM</code>

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
DevOps Guru	devops-guru: AWS_ENDPOINT_URL_DEVOPS_GURU
Direct Connect	direct-connect: AWS_ENDPOINT_URL_DIRECT_CONNECT
Application Discovery Service	application-discovery-service: AWS_ENDPOINT_URL_APPLICATION_DISCOVERY_SERVICE
DLM	dlm: AWS_ENDPOINT_URL_DLM
Database Migration Service	database-migration: AWS_ENDPOINT_URL_DATABASE_MIGRATION_SERVICE
DocDB	docdb: AWS_ENDPOINT_URL_DOCDB
DocDB Elastic	docdb-elastic: AWS_ENDPOINT_URL_DOCDB_ELASTIC
drs	drs: AWS_ENDPOINT_URL_DRS
Directory Service	directory-service: AWS_ENDPOINT_URL_DIRECTORY_SERVICE
DynamoDB	dynamodb: AWS_ENDPOINT_URL_DYNAMODB

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
DynamoDB Streams	Set the <code>AWS_ENDPOINT_URL_DYNAMODB_STREAMS</code> environment variable.
EBS	Set the <code>AWS_ENDPOINT_URL_EBS</code> environment variable.
EC2	Set the <code>AWS_ENDPOINT_URL_EC2</code> environment variable.
EC2 Instance Connect	Set the <code>AWS_ENDPOINT_URL_EC2_INSTANCE_CONNECT</code> environment variable.
ECR	Set the <code>AWS_ENDPOINT_URL_ECR</code> environment variable.
ECR PUBLIC	Set the <code>AWS_ENDPOINT_URL_ECR_PUBLIC</code> environment variable.
ECS	Set the <code>AWS_ENDPOINT_URL_ECS</code> environment variable.
EFS	Set the <code>AWS_ENDPOINT_URL_EFS</code> environment variable.
EKS	Set the <code>AWS_ENDPOINT_URL_EKS</code> environment variable.
EKS Auth	Set the <code>AWS_ENDPOINT_URL_EKS_AUTH</code> environment variable.
Elastic Inference	Set the <code>AWS_ENDPOINT_URL_ELASTIC_INFERENCE</code> environment variable.
ElastiCache	Set the <code>AWS_ENDPOINT_URL_ELASTICACHE</code> environment variable.

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
Elastic Beanstalk	environment variable id r ke fo sh Al co fil
Elastic Transcoder	e: AWS_ENDPOINT_URL_ELASTIC_BEANSTALK e:
Elastic Load Balancing	e: AWS_ENDPOINT_URL_ELASTIC_TRANSCODER r:
Elastic Load Balancing v2	e: AWS_ENDPOINT_URL_ELASTIC_LOAD_BALANCING o: c:
EMR	e: AWS_ENDPOINT_URL_ELASTIC_LOAD_BALANCING_V2 o: c:
EMR containers	e: AWS_ENDPOINT_URL_EMR er
EMR Serverless	e: AWS_ENDPOINT_URL_EMR_CONTAINERS i:
EntityResolution	e: AWS_ENDPOINT_URL_EMR_SERVERLESS r:
	e: AWS_ENDPOINT_URL_ENTITYRESOLUTION o:

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
Elasticsearch Service	environment variable: AWS_ENDPOINT_URL_ELASTICSEARCH_SERVICE
EventBridge	environment variable: AWS_ENDPOINT_URL_EVENTBRIDGE
Evidently	environment variable: AWS_ENDPOINT_URL_EVIDENTLY
finspace	environment variable: AWS_ENDPOINT_URL_FINSPLACE
finspace data	environment variable: AWS_ENDPOINT_URL_FINSPLACE_DATA
Firehose	environment variable: AWS_ENDPOINT_URL_FIREHOSE
fis	environment variable: AWS_ENDPOINT_URL_FIS
FMS	environment variable: AWS_ENDPOINT_URL_FMS
forecast	environment variable: AWS_ENDPOINT_URL_FORECAST
forecastquery	environment variable: AWS_ENDPOINT_URL_FORECASTQUERY
FraudDetector	environment variable: AWS_ENDPOINT_URL_FRAUDETECTOR

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
FreeTier	f: AWS_ENDPOINT_URL_FREETIER
FSx	f: AWS_ENDPOINT_URL_FSX
GameLift	g: AWS_ENDPOINT_URL_GAMELIFT
Glacier	g: AWS_ENDPOINT_URL_GLACIER
Global Accelerator	g: AWS_ENDPOINT_URL_GLOBAL_ACCELERATOR c:
Glue	g: AWS_ENDPOINT_URL_GLUE
grafana	g: AWS_ENDPOINT_URL_GRAFANA
Greengrass	g: AWS_ENDPOINT_URL_GREENGRASS s
GreengrassV2	g: AWS_ENDPOINT_URL_GREENGRASSV2 s\
GroundStation	g: AWS_ENDPOINT_URL_GROUNDSTATION t:
GuardDuty	g: AWS_ENDPOINT_URL_GUARDDUTY
Health	h: AWS_ENDPOINT_URL_HEALTH

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
HealthLake	healthlake AWS_ENDPOINT_URL_HEALTHLAKE
Honeycode	honeycode AWS_ENDPOINT_URL_HONEYCODE
IAM	iam AWS_ENDPOINT_URL_IAM
identitystore	identitystore AWS_ENDPOINT_URL_IDENTITYSTORE
imagebuilder	imagebuilder AWS_ENDPOINT_URL_IMAGEBUILDER
ImportExport	importexport AWS_ENDPOINT_URL_IMPORTEXPORT
Inspector	inspector AWS_ENDPOINT_URL_INSPECTOR
Inspector Scan	inspector-scan AWS_ENDPOINT_URL_INSPECTOR_SCAN
Inspector2	inspector2 AWS_ENDPOINT_URL_INSPECTOR2
InternetMonitor	internetmonitor AWS_ENDPOINT_URL_INTERNETMONITOR

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
IoT	iot AWS_ENDPOINT_URL_IOT
IoT Data Plane	iot AWS_ENDPOINT_URL_IOT_DATA_PLANE p.
IoT Jobs Data Plane	iot AWS_ENDPOINT_URL_IOT_JOBS_DATA_PLANE d: e
IoT 1Click Devices Service	iot AWS_ENDPOINT_URL_IOT_1CLICK_DEVICES_ k_ SERVICE _:
IoT 1Click Projects	iot AWS_ENDPOINT_URL_IOT_1CLICK_PROJECTS k_ s
IoTAnalytics	iot AWS_ENDPOINT_URL_IOTANALYTICS i:
IotDeviceAdvisor	iot AWS_ENDPOINT_URL_IOTDEVICEADVISOR a:
IoT Events	iot AWS_ENDPOINT_URL_IOT_EVENTS s
IoT Events Data	iot AWS_ENDPOINT_URL_IOT_EVENTS_DATA s_

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
IoT FleetHub	iot AWS_ENDPOINT_URL_IOTFLEETHUB url
IoT FleetWise	iot AWS_ENDPOINT_URL_IOTFLEETWISE i:
IoT Secure Tunneling	iot AWS_ENDPOINT_URL_IOTSECURETUNNELING t:
IoT SiteWise	iot AWS_ENDPOINT_URL_IOTSITWISE s:
IoT ThingsGraph	iot AWS_ENDPOINT_URL_IOTTHINGSGRAPH g:
IoT TwinMaker	iot AWS_ENDPOINT_URL_IOTTWINMAKER k:
IoT Wireless	iot AWS_ENDPOINT_URL_IOT_WIRELESS e:
ivs	ivs AWS_ENDPOINT_URL_IVS
IVS RealTime	ivs AWS_ENDPOINT_URL_IVS_REALTIME ir
ivschat	ivs AWS_ENDPOINT_URL_IVSCHAT

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
Kafka	ka: AWS_ENDPOINT_URL_KAFKA
KafkaConnect	ka: AWS_ENDPOINT_URL_KAFKACONNECT
kendra	ka: AWS_ENDPOINT_URL_KENDRA
Kendra Ranking	ka: AWS_ENDPOINT_URL_KENDRA_RANKING
Keyspaces	ka: AWS_ENDPOINT_URL_KEYSPACES
Kinesis	ka: AWS_ENDPOINT_URL_KINESIS
Kinesis Video Archived Media	ka: AWS_ENDPOINT_URL_KINESIS_VIDEO_ARCHIVED_MEDIA
Kinesis Video Media	ka: AWS_ENDPOINT_URL_KINESIS_VIDEO_MEDIA
Kinesis Video Signaling	ka: AWS_ENDPOINT_URL_KINESIS_VIDEO_SIGNALING

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
Kinesis Video WebRTC Storage	k: AWS_ENDPOINT_URL_KINESIS_VIDEO_WEBRTC_STORAGE
Kinesis Analytics	k: AWS_ENDPOINT_URL_KINESIS_ANALYTICS
Kinesis Analytics V2	k: AWS_ENDPOINT_URL_KINESIS_ANALYTICS_V2
Kinesis Video	k: AWS_ENDPOINT_URL_KINESIS_VIDEO
KMS	k: AWS_ENDPOINT_URL_KMS
LakeFormation	l: AWS_ENDPOINT_URL_LAKEFORMATION
Lambda	l: AWS_ENDPOINT_URL_LAMBDA
Launch Wizard	l: AWS_ENDPOINT_URL_LAUNCH_WIZARD
Lex Model Building Service	l: AWS_ENDPOINT_URL_LEX_MODEL_BUILDING_SERVICE

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
Lex Runtime Service	1: AWS_ENDPOINT_URL_LEX_RUNTIME_SERVICE
Lex Models V2	1: AWS_ENDPOINT_URL_LEX_MODELS_V2
Lex Runtime V2	1: AWS_ENDPOINT_URL_LEX_RUNTIME_V2
License Manager	1: AWS_ENDPOINT_URL_LICENSE_MANAGER
License Manager Linux Subscriptions	1: AWS_ENDPOINT_URL_LICENSE_MANAGER_LINUX_SUBSCRIPTIONS
License Manager User Subscriptions	1: AWS_ENDPOINT_URL_LICENSE_MANAGER_USER_SUBSCRIPTIONS
Lightsail	1: AWS_ENDPOINT_URL_LIGHTSAIL
Location	1: AWS_ENDPOINT_URL_LOCATION

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
CloudWatch Logs	Set the environment variable <code>AWS_ENDPOINT_URL_CLOUDWATCH_LOGS</code> to the endpoint URL for CloudWatch Logs.
CloudWatch Logs	Set the environment variable <code>AWS_ENDPOINT_URL_CLOUDWATCH_LOGS</code> to the endpoint URL for CloudWatch Logs.
LookoutEquipment	Set the environment variable <code>AWS_ENDPOINT_URL_LOOKOUTEQUIPMENT</code> to the endpoint URL for Lookout for Equipment.
LookoutMetrics	Set the environment variable <code>AWS_ENDPOINT_URL_LOOKOUTMETRICS</code> to the endpoint URL for Lookout for Metrics.
LookoutVision	Set the environment variable <code>AWS_ENDPOINT_URL_LOOKOUTVISION</code> to the endpoint URL for Lookout for Vision.
m2	Set the environment variable <code>AWS_ENDPOINT_URL_M2</code> to the endpoint URL for Macie2.
Machine Learning	Set the environment variable <code>AWS_ENDPOINT_URL_MACHINE_LEARNING</code> to the endpoint URL for Amazon SageMaker.
Macie2	Set the environment variable <code>AWS_ENDPOINT_URL_MACIE2</code> to the endpoint URL for Macie2.
ManagedBlockchain	Set the environment variable <code>AWS_ENDPOINT_URL_MANAGEDBLOCKCHAIN</code> to the endpoint URL for Amazon Managed Blockchain.
ManagedBlockchain Query	Set the environment variable <code>AWS_ENDPOINT_URL_MANAGEDBLOCKCHAIN_QUERY</code> to the endpoint URL for Amazon Managed Blockchain Query.

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
Marketplace Agreement	<code>marketplace-agreement</code> <code>AWS_ENDPOINT_URL_MARKETPLACE_AGREEMENT</code>
Marketplace Catalog	<code>marketplace-catalog</code> <code>AWS_ENDPOINT_URL_MARKETPLACE_CATALOG</code>
Marketplace Deployment	<code>marketplace-deployment</code> <code>AWS_ENDPOINT_URL_MARKETPLACE_DEPLOYMENT</code>
Marketplace Entitlement Service	<code>marketplace-entitlement-service</code> <code>AWS_ENDPOINT_URL_MARKETPLACE_ENTITLEMENT_SERVICE</code>
Marketplace Commerce Analytics	<code>marketplace-commerce-analytics</code> <code>AWS_ENDPOINT_URL_MARKETPLACE_COMMERCE_ANALYTICS</code>
MediaConnect	<code>mediaconnect</code> <code>AWS_ENDPOINT_URL_MEDIACONNECT</code>
MediaConvert	<code>mediaconvert</code> <code>AWS_ENDPOINT_URL_MEDIACONVERT</code>

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
MediaLive	media-live
MediaPackage	media-package
MediaPackage Vod	media-package-vod
MediaPackageV2	media-packagev2
MediaStore	media-store
MediaStore Data	media-store-data
MediaTailor	media-tailor
Medical Imaging	medical-imaging
MemoryDB	memorydb
Marketplace Metering	marketplace-metering

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
Migration Hub	m: AWS_ENDPOINT_URL_MIGRATION_HUB _HUB
mgn	m: AWS_ENDPOINT_URL_MGN
Migration Hub Refactor Spaces	m: AWS_ENDPOINT_URL_MIGRATION_HUB_REFAC _HUB_REFAC TOR_SPACES CTOR_SPACES
MigrationHub Config	m: AWS_ENDPOINT_URL_MIGRATIONHUB_CONFIG MIGRATIONHUB_CONFIG
MigrationHubOrchestrator	m: AWS_ENDPOINT_URL_MIGRATIONHUBORCHESTRATOR MIGRATIONHUBORCHESTRATOR
MigrationHubStrategy	m: AWS_ENDPOINT_URL_MIGRATIONHUBSTRATEGY MIGRATIONHUBSTRATEGY
Mobile	m: AWS_ENDPOINT_URL_MOBILE
mq	m: AWS_ENDPOINT_URL_MQ
MTurk	m: AWS_ENDPOINT_URL_MTURK
MWAA	m: AWS_ENDPOINT_URL_MWAA

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
Neptune	ne AWS_ENDPOINT_URL_NEPTUNE
Neptune Graph	ne AWS_ENDPOINT_URL_NEPTUNE_GRAPH ra
neptunedata	ne AWS_ENDPOINT_URL_NEPTUNEDATA ta
Network Firewall	ne AWS_ENDPOINT_URL_NETWORK_FIREWALL i:
NetworkManager	ne AWS_ENDPOINT_URL_NETWORKMANAGER na
NetworkMonitor	ne AWS_ENDPOINT_URL_NETWORKMONITOR n:
nimble	n: AWS_ENDPOINT_URL_NIMBLE
OAM	o: AWS_ENDPOINT_URL_OAM
Omics	or AWS_ENDPOINT_URL_OMICS
OpenSearch	op AWS_ENDPOINT_URL_OPENSEARCH h
OpenSearchServerless	op AWS_ENDPOINT_URL_OPENSEARCHSERVERLESS h: s:

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
OpsWorks	o: AWS_ENDPOINT_URL_OPSWORKS
OpsWorksCM	o: AWS_ENDPOINT_URL_OPSWORKSCM
Organizations	o: AWS_ENDPOINT_URL_ORGANIZATIONS
OSIS	o: AWS_ENDPOINT_URL_OSIS
Outposts	o: AWS_ENDPOINT_URL_OUTPOSTS
p8data	p: AWS_ENDPOINT_URL_P8DATA
p8data	p: AWS_ENDPOINT_URL_P8DATA
Panorama	p: AWS_ENDPOINT_URL_PANORAMA
Payment Cryptography	p: AWS_ENDPOINT_URL_PAYMENT_CRYPTOGRAPHY
Payment Cryptography Data	p: AWS_ENDPOINT_URL_PAYMENT_CRYPTOGRAPHY_DATA
Pca Connector Ad	p: AWS_ENDPOINT_URL_PCA_CONNECTOR_AD

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
Personalize	p: AWS_ENDPOINT_URL_PERSONALIZE
Personalize Events	p: AWS_ENDPOINT_URL_PERSONALIZE_EVENTS
Personalize Runtime	p: AWS_ENDPOINT_URL_PERSONALIZE_RUNTIME
PI	p: AWS_ENDPOINT_URL_PI
Pinpoint	p: AWS_ENDPOINT_URL_PINPOINT
Pinpoint Email	p: AWS_ENDPOINT_URL_PINPOINT_EMAIL
Pinpoint SMS Voice	p: AWS_ENDPOINT_URL_PINPOINT_SMS_VOICE
Pinpoint SMS Voice V2	p: AWS_ENDPOINT_URL_PINPOINT_SMS_VOICE_V2
Pipes	p: AWS_ENDPOINT_URL_PIPES
Polly	p: AWS_ENDPOINT_URL_POLLY

serviceId	Service endpoint URL	environment variable
Pricing	pricing	AWS_ENDPOINT_URL_PRICING
PrivateNetworks	privatenetworks	AWS_ENDPOINT_URL_PRIVATENETWORKS
Proton	proton	AWS_ENDPOINT_URL_PROTON
QBusiness	qbusiness	AWS_ENDPOINT_URL_QBUSINESS
QConnect	qconnect	AWS_ENDPOINT_URL_QCONNECT
QLDB	qldb	AWS_ENDPOINT_URL_QLDB
QLDB Session	qldb-session	AWS_ENDPOINT_URL_QLDB_SESSION
QuickSight	quicksight	AWS_ENDPOINT_URL_QUICKSIGHT
RAM	ram	AWS_ENDPOINT_URL_RAM
rbn	rbn	AWS_ENDPOINT_URL_RBN
RDS	rds	AWS_ENDPOINT_URL_RDS
RDS Data	rds-data	AWS_ENDPOINT_URL_RDS_DATA
Redshift	redshift	AWS_ENDPOINT_URL_REDSHIFT

serviceId	Service ID	Environment Variable
Redshift Data	redshift-data	AWS_ENDPOINT_URL_REDSHIFT_DATA
Redshift Serverless	redshift-serverless	AWS_ENDPOINT_URL_REDSHIFT_SERVERLESS
Rekognition	rekognition	AWS_ENDPOINT_URL_REKOGNITION
repostspace	repostspace	AWS_ENDPOINT_URL_REPOSTSPACE
resiliencehub	resiliencehub	AWS_ENDPOINT_URL_RESILIENCEHUB
Resource Explorer 2	resource-explorer-2	AWS_ENDPOINT_URL_RESOURCE_EXPLORER_2
Resource Groups	resource-groups	AWS_ENDPOINT_URL_RESOURCE_GROUPS
Resource Groups Tagging API	resource-groups-tagging-api	AWS_ENDPOINT_URL_RESOURCE_GROUPS_TAGGING_API

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
RoboMaker	<code>AWS_ENDPOINT_URL_ROBOMAKER</code>
RolesAnywhere	<code>AWS_ENDPOINT_URL_ROLESEANYWHERE</code>
Route 53	<code>AWS_ENDPOINT_URL_ROUTE_53</code>
Route53 Recovery Cluster	<code>AWS_ENDPOINT_URL_ROUTE53_RECOVERY_CLUSTER</code>
Route53 Recovery Control Config	<code>AWS_ENDPOINT_URL_ROUTE53_RECOVERY_CONTROL_CONFIG</code>
Route53 Recovery Readiness	<code>AWS_ENDPOINT_URL_ROUTE53_RECOVERY_READINESS</code>
Route 53 Domains	<code>AWS_ENDPOINT_URL_ROUTE_53_DOMAINS</code>
Route53Resolver	<code>AWS_ENDPOINT_URL_ROUTE53RESOLVER</code>
RUM	<code>AWS_ENDPOINT_URL_RUM</code>

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
S3	s: AWS_ENDPOINT_URL_S3
S3 Control	s: AWS_ENDPOINT_URL_S3_CONTROL
S3Outposts	s: AWS_ENDPOINT_URL_S3OUTPOSTS
SageMaker	s: AWS_ENDPOINT_URL_SAGEMAKER
SageMaker A2I Runtime	s: AWS_ENDPOINT_URL_SAGEMAKER_A2I_RUNTIME
Sagemaker Edge	s: AWS_ENDPOINT_URL_SAGEMAKER_EDGE
SageMaker FeatureStore Runtime	s: AWS_ENDPOINT_URL_SAGEMAKER_FEATURESTORE_RUNTIME
SageMaker Geospatial	s: AWS_ENDPOINT_URL_SAGEMAKER_GEOSPATIAL

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
SageMaker Metrics	set AWS_ENDPOINT_URL_SAGEMAKER_METRICS
SageMaker Runtime	set AWS_ENDPOINT_URL_SAGEMAKER_RUNTIME
savingsplans	set AWS_ENDPOINT_URL_SAVINGSPLANS
Scheduler	set AWS_ENDPOINT_URL_SCHEDULER
schemas	set AWS_ENDPOINT_URL_SCHEMAS
SimpleDB	set AWS_ENDPOINT_URL_SIMPLEDB
Secrets Manager	set AWS_ENDPOINT_URL_SECRETS_MANAGER
SecurityHub	set AWS_ENDPOINT_URL_SECURITYHUB
SecurityLake	set AWS_ENDPOINT_URL_SECURITYLAKE
ServerlessApplicationRepository	set AWS_ENDPOINT_URL_SERVERLESSAPPLICATIONREPOSITORY

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
Service Quotas	set AWS_ENDPOINT_URL_SERVICE_QUOTAS
Service Catalog	set AWS_ENDPOINT_URL_SERVICE_CATALOG
Service Catalog AppRegistry	set AWS_ENDPOINT_URL_SERVICE_CATALOG_APP_REGISTRY
ServiceDiscovery	set AWS_ENDPOINT_URL_SERVICEDISCOVERY
SES	set AWS_ENDPOINT_URL_SES
SESV2	set AWS_ENDPOINT_URL_SESV2
Shield	set AWS_ENDPOINT_URL_SHIELD
signer	set AWS_ENDPOINT_URL_SIGNER
SimSpaceWeaver	set AWS_ENDPOINT_URL_SIMSPACEWEAVER
SMS	set AWS_ENDPOINT_URL_SMS

serviceId	Service ID	Environment Variable
	Service ID	Environment Variable
Snow Device Management	SNOW_DEVICE_MANAGEMENT	AWS_ENDPOINT_URL_SNOW_DEVICE_MANAGEMENT
Snowball	SNOWBALL	AWS_ENDPOINT_URL_SNOWBALL
SNS	SNS	AWS_ENDPOINT_URL_SNS
SQS	SQS	AWS_ENDPOINT_URL_SQS
SSM	SSM	AWS_ENDPOINT_URL_SSM
SSM Contacts	SSM_CONTACTS	AWS_ENDPOINT_URL_SSM_CONTACTS
SSM Incidents	SSM_INCIDENTS	AWS_ENDPOINT_URL_SSM_INCIDENTS
Ssm Sap	SSM_SAP	AWS_ENDPOINT_URL_SSM_SAP
SSO	SSO	AWS_ENDPOINT_URL_SSO
SSO Admin	SSO_ADMIN	AWS_ENDPOINT_URL_SSO_ADMIN
SSO OIDC	SSO_OIDC	AWS_ENDPOINT_URL_SSO_OIDC
SFN	SFN	AWS_ENDPOINT_URL_SFN

serviceId	Service ID	AWS_ENDPOINT_URL_<SERVICE>	environment variable
Storage Gateway	storage-gateway	AWS_ENDPOINT_URL_STORAGE_GATEWAY	
STS	sts	AWS_ENDPOINT_URL_STS	
SupplyChain	supply-chain	AWS_ENDPOINT_URL_SUPPLYCHAIN	
Support	support	AWS_ENDPOINT_URL_SUPPORT	
Support App	support-app	AWS_ENDPOINT_URL_SUPPORT_APP	
SWF	swf	AWS_ENDPOINT_URL_SWF	
synthetics	synthetics	AWS_ENDPOINT_URL_SYNTHETICS	
Textract	textract	AWS_ENDPOINT_URL_TEXTRACT	
Timestream InfluxDB	timestream-influxdb	AWS_ENDPOINT_URL_TIMESTREAM_INFLUXDB	
Timestream Query	timestream-query	AWS_ENDPOINT_URL_TIMESTREAM_QUERY	
Timestream Write	timestream-write	AWS_ENDPOINT_URL_TIMESTREAM_WRITE	

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
tnb	t: AWS_ENDPOINT_URL_TNB
Transcribe	t: AWS_ENDPOINT_URL_TRANSCRIBE
Transfer	t: AWS_ENDPOINT_URL_TRANSFER
Translate	t: AWS_ENDPOINT_URL_TRANSLATE
TrustedAdvisor	t: AWS_ENDPOINT_URL_TRUSTEDADVISOR v:
VerifiedPermissions	v: AWS_ENDPOINT_URL_VERIFIEDPERMISSIONS e: s:
Voice ID	v: AWS_ENDPOINT_URL_VOICE_ID
VPC Lattice	v: AWS_ENDPOINT_URL_VPC_LATTICE c:
WAF	w: AWS_ENDPOINT_URL_WAF
WAF Regional	w: AWS_ENDPOINT_URL_WAF_REGIONAL n:
WAFV2	w: AWS_ENDPOINT_URL_WAFV2

serviceId	Set AWS_ENDPOINT_URL_<SERVICE> environment variable
WellArchitected	wc: AWS_ENDPOINT_URL_WELLARCHITECTED
Wisdom	w: AWS_ENDPOINT_URL_WISDOM
WorkDocs	wc: AWS_ENDPOINT_URL_WORKDOCS
WorkLink	wc: AWS_ENDPOINT_URL_WORKLINK
WorkMail	wc: AWS_ENDPOINT_URL_WORKMAIL
WorkMailMessageFlow	wc: AWS_ENDPOINT_URL_WORKMAILMESSAGEFLOW
WorkSpaces	wc: AWS_ENDPOINT_URL_WORKSPACES
WorkSpaces Thin Client	wc: AWS_ENDPOINT_URL_WORKSPACES_THIN_CLIENT
WorkSpaces Web	wc: AWS_ENDPOINT_URL_WORKSPACES_WEB
XRay	x: AWS_ENDPOINT_URL_XRAY

Endpoint configuration and settings precedence

Endpoint configuration settings are located in multiple places, such as the system or user environment variables, local AWS configuration files, or explicitly declared on the command line as a parameter. The AWS CLI endpoint configuration settings take precedence in the following order:

1. The `--endpoint-url` command line option.
2. If enabled, the `AWS_IGNORE_CONFIGURED_ENDPOINT_URLS` global endpoint environment variable or profile setting `ignore_configure_endpoint_urls` to ignore custom endpoints.
3. The value provided by a service-specific environment variable `AWS_ENDPOINT_URL_<SERVICE>`, such as `AWS_ENDPOINT_URL_DYNAMODB`.
4. The values provided by the `AWS_USE_DUALSTACK_ENDPOINT`, `AWS_USE_FIPS_ENDPOINT`, and `AWS_ENDPOINT_URL` environment variables.
5. The service-specific endpoint value provided by the `endpoint_url` setting within a services section of the shared config file.
6. The value provided by the `endpoint_url` setting within a profile of the shared config file.
7. `use_dualstack_endpoint`, `use_fips_endpoint`, and `endpoint_url` settings.
8. Any default endpoint URL for the respective AWS service is used last. For a list of the standard service endpoints available in each Region, see [AWS Regions and Endpoints](#) in the *Amazon Web Services General Reference*.

Authentication and access credentials

You must establish how the AWS CLI authenticates with AWS when you develop with AWS services. To configure credentials for programmatic access for the AWS CLI, choose one of the following options. The options are in order of recommendation.

Which user needs programmatic access?	Purpose	Instructions
Workforce identity (AWS IAM Identity Center users)	(Recommended) Use short-term credentials.	the section called "IAM Identity Center authentication"
IAM	Use short-term credentials.	the section called "Short-term credentials"
IAM or Workforce identity (AWS IAM Identity Center users)	Use Amazon EC2 instance metadata for credentials.	the section called "Use credentials for Amazon EC2 instance metadata"
IAM or Workforce identity (AWS IAM Identity Center users)	Pair another credential method and assume a role for permissions.	the section called "IAM roles"
IAM	(Not recommended) Use long-term credentials.	the section called "IAM users"
IAM or Workforce identity (AWS IAM Identity Center users)	(Not recommended) Pair another credential method but use credential values stored in a location outside of the AWS CLI.	the section called "External credentials"

Configuration and credential precedence

Credentials and configuration settings are located in multiple places, such as the system or user environment variables, local AWS configuration files, or explicitly declared on the command line as a parameter. Certain authentication take precedence over others. The AWS CLI authentication settings take precedence in the following order:

1. **[Command line options](#)** – Overrides settings in any other location, such as the `--region`, `--output`, and `--profile` parameters.
2. **[Environment variables](#)** – You can store values in your system's environment variables.
3. **[Assume role](#)** – Assume the permissions of an IAM role through configuration or the `aws sts assume-role` command.
4. **[Assume role with web identity](#)** – Assume the permissions of an IAM role using web identity through configuration or the `aws sts assume-role` command.
5. **[AWS IAM Identity Center](#)** – The IAM Identity Center configuration settings stored in the `config` file are updated when you run the `aws configure sso` command. Credentials are then authenticated when you run the `aws sso login` command. The `config` file is located at `~/.aws/config` on Linux or macOS, or at `C:\Users\USERNAME\.aws\config` on Windows.
6. **[Credentials file](#)** – The credentials and `config` file are updated when you run the command `aws configure`. The credentials file is located at `~/.aws/credentials` on Linux or macOS, or at `C:\Users\USERNAME\.aws\credentials` on Windows.
7. **[Custom process](#)** – Get your credentials from an external source.
8. **[Configuration file](#)** – The credentials and `config` file are updated when you run the command `aws configure`. The `config` file is located at `~/.aws/config` on Linux or macOS, or at `C:\Users\USERNAME\.aws\config` on Windows.
9. **[Container credentials](#)** – You can associate an IAM role with each of your Amazon Elastic Container Service (Amazon ECS) task definitions. Temporary credentials for that role are then available to that task's containers. For more information, see [IAM Roles for Tasks](#) in the *Amazon Elastic Container Service Developer Guide*.
10. **[Amazon EC2 instance profile credentials](#)** – You can associate an IAM role with each of your Amazon Elastic Compute Cloud (Amazon EC2) instances. Temporary credentials for that role are then available to code running in the instance. The credentials are delivered through the Amazon EC2 metadata service. For more information, see [IAM Roles for Amazon EC2](#) in the *Amazon EC2 User Guide* and [Using Instance Profiles](#) in the *IAM User Guide*.

Additional topics in this section

- [the section called “IAM Identity Center authentication”](#)
- [the section called “Short-term credentials”](#)
- [the section called “IAM roles”](#)
- [the section called “IAM users”](#)
- [the section called “Use credentials for Amazon EC2 instance metadata”](#)
- [the section called “External credentials”](#)

Configure the AWS CLI to use AWS IAM Identity Center

There are primarily two ways to authenticate users with AWS IAM Identity Center (IAM Identity Center) to get credentials to run AWS Command Line Interface (AWS CLI) commands through the config file:

- **(Recommended)** [SSO token provider configuration](#). The SSO token provider configuration, your AWS SDK or tool can automatically retrieve refreshed authentication tokens.
- [Legacy non-refreshable configuration](#). When using the legacy non-refreshable configuration, you need to manually refresh the token as it periodically expires.

When using IAM Identity Center, you can login to Active Directory, a built-in IAM Identity Center directory, or [another IdP connected to IAM Identity Center](#). You can map these credentials to an AWS Identity and Access Management (IAM) role for you to run AWS CLI commands.

Regardless of which IdP you use, IAM Identity Center abstracts those distinctions away. For example, you can connect Microsoft Azure AD as described in the blog article [The Next Evolution in IAM Identity Center](#).

Note

For information on using bearer auth, which uses no account ID and role, see [Setting up to use the AWS CLI with CodeCatalyst](#) in the *Amazon CodeCatalyst User Guide*.

Topics in this section

- [Configure the AWS CLI to use IAM Identity Center token provider credentials with automatic authentication refresh](#)
- [Legacy non-refreshable configuration for AWS IAM Identity Center](#)
- [Use an IAM Identity Center named profile](#)

Configure the AWS CLI to use IAM Identity Center token provider credentials with automatic authentication refresh

This topic describes how to configure the AWS CLI to authenticate users with the AWS IAM Identity Center (IAM Identity Center) token provider configuration. Using this SSO token provider configuration, your AWS SDK or tool can automatically retrieve refreshed authentication tokens.

When using IAM Identity Center, you can login to Active Directory, a built-in IAM Identity Center directory, or [another IdP connected to IAM Identity Center](#). You can map these credentials to an AWS Identity and Access Management (IAM) role for you to run AWS CLI commands.

Regardless of which IdP you use, IAM Identity Center abstracts those distinctions away. For example, you can connect Microsoft Azure AD as described in the blog article [The Next Evolution in IAM Identity Center](#).

Note

For information on using bearer auth, which uses no account ID and role, see [Setting up to use the AWS CLI with CodeCatalyst](#) in the *Amazon CodeCatalyst User Guide*.

You can use the SSO token provider configuration to automatically refresh authentication tokens as needed for your application, and to use extended [session duration options](#). You can configure this in the following ways:

- Automatically, using the `aws configure sso` and `aws configure sso-session` commands. The following commands are wizards that guide you through configuring your profile and `sso-session` information are the following:
 - Use [aws configure sso](#) to create or edit both your config profiles and `sso-session` sections.
 - Use [aws configure sso-session](#) to create or edit only `sso-session` sections.
- [Manually](#), by editing the `config` file that stores the named profiles.

Prerequisites

- Install the AWS CLI. For more information, see [the section called “Install/Update”](#).
- You must first have access to SSO authentication within IAM Identity Center. Choose one of the following methods to access your AWS credentials.

I do not have established access through IAM Identity Center

Follow the instructions in [Getting started](#) in the *AWS IAM Identity Center User Guide*. This process activates IAM Identity Center, creates an administrative user, and adds an appropriate least-privilege permission set.

Note

Create a permission set that applies least-privilege permissions. We recommend using the predefined `PowerUserAccess` permission set, unless your employer has created a custom permission set for this purpose.

Exit the portal and sign in again to see your AWS accounts and options for Administrator or `PowerUserAccess`. Select `PowerUserAccess` when working with the SDK. This also helps you find details about programmatic access.

I already have access to AWS through a federated identity provider managed by my employer (such as Azure AD or Okta)

Sign in to AWS through your identity provider’s portal. If your Cloud Administrator has granted you `PowerUserAccess` (developer) permissions, you see the AWS accounts that you have access to and your permission set. Next to the name of your permission set, you see options to access the accounts manually or programmatically using that permission set.

Custom implementations might result in different experiences, such as different permission set names. If you're not sure which permission set to use, contact your IT team for help.

I already have access to AWS through the AWS access portal managed by my employer

Sign in to AWS through the AWS access portal. If your Cloud Administrator has granted you `PowerUserAccess` (developer) permissions, you see the AWS accounts that you have access to

and your permission set. Next to the name of your permission set, you see options to access the accounts manually or programmatically using that permission set.

I already have access to AWS through a federated custom identity provider managed by my employer

Contact your IT team for help.

Configure your profile with the `aws configure sso` wizard

To configure both an IAM Identity Center profile and `sso-session` to your AWS CLI

1. Gather your IAM Identity Center information by performing the following:
 1. In your AWS access portal, select the permission set you use for development, and select the **Access keys** link.
 2. In the **Get credentials** dialog box, choose the tab that matches your operating system.
 3. Choose the **IAM Identity Center credentials** method to get the SSO Start URL and SSO Region values that you need to run `aws configure sso`.
 4. For information on which scopes value to register, see [OAuth 2.0 Access scopes](#) in the *IAM Identity Center User Guide*.
2. In your preferred terminal, run the `aws configure sso` command and provide your IAM Identity Center start URL and the AWS Region that hosts the Identity Center directory.

```
$ aws configure sso
SSO session name (Recommended): my-sso
SSO start URL [None]: https://my-sso-portal.awsapps.com/start
SSO region [None]: us-east-1
SSO registration scopes [None]: sso:account:access
```

3. The AWS CLI attempts to open your default browser and begin the login process for your IAM Identity Center account.

```
Attempting to automatically open the SSO authorization page in your default browser.
```

If the AWS CLI cannot open the browser, the following message appears with instructions on how to manually start the login process.


If the browser does not open or you wish to use a different device to authorize this request, open the following URL:

<https://device.sso.us-west-2.amazonaws.com/>

Then enter the code:

QCFK-N451

IAM Identity Center uses the code to associate the IAM Identity Center session with your current AWS CLI session. The IAM Identity Center browser page prompts you to log in with your IAM Identity Center credentials. This gives permissions to the AWS CLI to retrieve and display the AWS accounts and roles that you are authorized to use with IAM Identity Center.

 **Note**

The sign in process may prompt you to allow the AWS CLI access to your data. Since the AWS CLI is built on top of the SDK for Python, permission messages may contain variations of the `botocore` name.

4. The AWS CLI displays the AWS accounts available for you to use. If you are authorized to use only one account, the AWS CLI selects that account for you automatically and skips the prompt. The AWS accounts that are available for you to use are determined by your user configuration in IAM Identity Center.

There are 2 AWS accounts available to you.

```
> DeveloperAccount, developer-account-admin@example.com (123456789011)
   ProductionAccount, production-account-admin@example.com (123456789022)
```

Use the arrow keys to select the account you want to use. The ">" character on the left points to the current choice. Press ENTER to make your selection.

5. The AWS CLI confirms your account choice, and displays the IAM roles that are available to you in the selected account. If the selected account lists only one role, the AWS CLI selects that role for you automatically and skips the prompt. The roles that are available for you to use are determined by your user configuration in IAM Identity Center.

Using the account ID **123456789011**
There are 2 roles available to you.

```
> ReadOnly
   FullAccess
```

Use the arrow keys to select the IAM role you want to use and press <ENTER>.

- Specify the [default output format](#), the [default AWS Region](#) to send commands to, and providing a [name for the profile](#) so you can reference this profile from among all those defined on the local computer. In the following example, the user enters a default Region, default output format, and the name of the profile. If you have a previously existing configuration, you can alternatively press <ENTER> to select any default values that are shown between the square brackets. The suggested profile name is the account ID number followed by an underscore followed by the role name.

```
CLI default client Region [None]: us-west-2<ENTER>
CLI default output format [None]: json<ENTER>
CLI profile name [123456789011_ReadOnly]: my-dev-profile<ENTER>
```

Note

If you specify default as the profile name, this profile becomes the one used whenever you run an AWS CLI command and do not specify a profile name.

- A final message describes the completed profile configuration.

To use this profile, specify the profile name using `--profile`, as shown:

```
aws s3 ls --profile my-dev-profile
```

- This results in creating the `sso-session` section and named profile in `~/.aws/config` that looks like the following:

```
[profile my-dev-profile]
sso_session = my-sso
sso_account_id = 123456789011
sso_role_name = readOnly
region = us-west-2
output = json

[sso-session my-sso]
sso_region = us-east-1
```

```
sso_start_url = https://my-sso-portal.awsapps.com/start
sso_registration_scopes = sso:account:access
```

You can now use this `sso-session` and profile to request refreshed credentials. Use the `aws sso login` command to request and retrieve the credentials needed to run commands. For instructions, see [Use an IAM Identity Center named profile](#).

Configure only your `sso-session` section with `aws configure sso-session wizard`

The `aws configure sso-session` command only updates the `sso-session` sections in the `~/.aws/config` file. This command can be used to create or update your sessions. This is useful if you already have existing configuration settings and would like to create new or edit existing `sso-session` configuration.

Run the `aws configure sso-session` command and provide your IAM Identity Center start URL and the AWS Region that hosts the Identity Center directory.

```
$ aws configure sso-session
SSO session name: my-sso
SSO start URL [None]: https://my-sso-portal.awsapps.com/start
SSO region [None]: us-east-1
SSO registration scopes [None]: sso:account:access
```

After entering in your information a message describes the completed profile configuration.

```
Completed configuring SSO session: my-sso
Run the following to login and refresh access token for this session:

aws sso login --sso-session my-sso
```

Note

If you are signed in to the `sso-session` you are updating, refresh your token by running the `aws sso login` command.

Manual configuration using the config file

The `sso-session` section of the config file is used to group configuration variables for acquiring SSO access tokens, which can then be used to acquire AWS credentials. The following settings are used:

- **(Required)** [`sso_start_url`](#)
- **(Required)** [`sso_region`](#)
- [`sso_account_id`](#)
- [`sso_role_name`](#)
- [`sso_registration_scopes`](#)

You define an `sso-session` section and associate it to a profile. `sso_region` and `sso_start_url` must be set within the `sso-session` section. Typically, `sso_account_id` and `sso_role_name` must be set in the profile section so that the SDK can request SSO credentials.

The following example configures the SDK to request SSO credentials and supports automated token refresh:

```
[profile dev]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole

[sso-session my-sso]
sso_region = us-east-1
sso_start_url = https://my-sso-portal.awsapps.com/start
```

This also allows `sso-session` configurations to be reused across multiple profiles:

```
[profile dev]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole

[profile prod]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole2
```

```
[sso-session my-sso]
sso_region = us-east-1
sso_start_url = https://my-sso-portal.awsapps.com/start
```

However, `sso_account_id` and `sso_role_name` aren't required for all scenarios of SSO token configuration. If your application only uses AWS services that support bearer authentication, then traditional AWS credentials are not needed. Bearer authentication is an HTTP authentication scheme that uses security tokens called bearer tokens. In this scenario, `sso_account_id` and `sso_role_name` aren't required. See the individual guide for your AWS service to determine if it supports bearer token authorization.

Additionally, registration scopes can be configured as part of a `sso-session`. Scope is a mechanism in OAuth 2.0 to limit an application's access to a user's account. An application can request one or more scopes, and the access token issued to the application will be limited to the scopes granted. These scopes define the permissions requested to be authorized for the registered OIDC client and access tokens retrieved by the client. The following example sets `sso_registration_scopes` to provide access for listing accounts/roles:

```
[sso-session my-sso]
sso_region = us-east-1
sso_start_url = https://my-sso-portal.awsapps.com/start
sso_registration_scopes = sso:account:access
```

The authentication token is cached to disk under the `~/ .aws/sso/cache` directory with a filename based on the session name.

Legacy non-refreshable configuration for AWS IAM Identity Center

This topic describes how to configure the AWS CLI to authenticate users with AWS IAM Identity Center (IAM Identity Center) to get credentials to run AWS CLI commands using the legacy method. When using the legacy non-refreshable configuration, you need to manually refresh the token as it periodically expires.

When using IAM Identity Center, you can login to Active Directory, a built-in IAM Identity Center directory, or [another IdP connected to IAM Identity Center](#). You can map these credentials to an AWS Identity and Access Management (IAM) role where you can run AWS CLI commands.

Regardless of which IdP you use, IAM Identity Center abstracts those distinctions away. For example, you can connect Microsoft Azure AD as described in the blog article [The Next Evolution in IAM Identity Center](#).

Note

For information on using bearer auth, which uses no account ID and role, see [Setting up to use the AWS CLI with CodeCatalyst](#) in the *Amazon CodeCatalyst User Guide*.

You can configure one or more of your AWS CLI [named profiles](#) to use a role from a legacy IAM Identity Center in the following ways:

- [Automatically](#), using the `aws configure sso` command.
- [Manually](#), by editing the `config` file that stores the named profiles.

Prerequisites

- Install the AWS CLI. For more information, see [the section called “Install/Update”](#).
- You must first have access to SSO authentication within IAM Identity Center. Choose one of the following methods to access your AWS credentials.

I do not have established access through IAM Identity Center

Follow the instructions in [Getting started](#) in the *AWS IAM Identity Center User Guide*. This process activates IAM Identity Center, creates an administrative user, and adds an appropriate least-privilege permission set.

Note

Create a permission set that applies least-privilege permissions. We recommend using the predefined `PowerUserAccess` permission set, unless your employer has created a custom permission set for this purpose.

Exit the portal and sign in again to see your AWS accounts and options for Administrator or `PowerUserAccess`. Select `PowerUserAccess` when working with the SDK. This also helps you find details about programmatic access.

I already have access to AWS through a federated identity provider managed by my employer (such as Azure AD or Okta)

Sign in to AWS through your identity provider's portal. If your Cloud Administrator has granted you `PowerUserAccess` (developer) permissions, you see the AWS accounts that you have access to and your permission set. Next to the name of your permission set, you see options to access the accounts manually or programmatically using that permission set.

Custom implementations might result in different experiences, such as different permission set names. If you're not sure which permission set to use, contact your IT team for help.

I already have access to AWS through the AWS access portal managed by my employer

Sign in to AWS through the AWS access portal. If your Cloud Administrator has granted you `PowerUserAccess` (developer) permissions, you see the AWS accounts that you have access to and your permission set. Next to the name of your permission set, you see options to access the accounts manually or programmatically using that permission set.

I already have access to AWS through a federated custom identity provider managed by my employer

Contact your IT team for help.

Automatic configuration for legacy configuration

To configure an IAM Identity Center profile to your AWS CLI

1. Run the `aws configure sso` command and provide your IAM Identity Center start URL and the AWS Region that hosts the Identity Center directory.

```
$ aws configure sso
SSO session name (Recommended):
SSO start URL [None]: https://my-sso-portal.awsapps.com/start
SSO region [None]:us-east-1
```

2. The AWS CLI attempts to open your default browser and begin the login process for your IAM Identity Center account.

```
SSO authorization page has automatically been opened in your default browser.
Follow the instructions in the browser to complete this authorization request.
```

If the AWS CLI cannot open the browser, the following message appears with instructions on how to manually start the login process.

Using a browser, open the following URL:

<https://device.sso.us-west-2.amazonaws.com/>

and enter the following code:

`QCFK-N451`

IAM Identity Center uses the code to associate the IAM Identity Center session with your current AWS CLI session. The IAM Identity Center browser page prompts you to sign in with your IAM Identity Center credentials. This gives permissions to the AWS CLI to retrieve and display the AWS accounts and roles that you are authorized to use with IAM Identity Center.

3. Next, the AWS CLI displays the AWS accounts available for you to use. If you are authorized to use only one account, the AWS CLI selects that account for you automatically and skips the prompt. The AWS accounts that are available for you to use are determined by your user configuration in IAM Identity Center.

There are 2 AWS accounts available to you.

```
> DeveloperAccount, developer-account-admin@example.com (123456789011)
  ProductionAccount, production-account-admin@example.com (123456789022)
```

Use the arrow keys to select the account you want to use with this profile. The ">" character on the left points to the current choice. Press ENTER to make your selection.

4. Next, the AWS CLI confirms your account choice, and displays the IAM roles that are available to you in the selected account. If the selected account lists only one role, the AWS CLI selects that role for you automatically and skips the prompt. The roles that are available for you to use are determined by your user configuration in IAM Identity Center.

Using the account ID `123456789011`

There are 2 roles available to you.

```
> ReadOnly
  FullAccess
```

Use the arrow keys to select the IAM role you want to use with this profile and press <ENTER>.

5. The AWS CLI confirms your role selection.

```
Using the role name "ReadOnly"
```

6. Finish the configuration of your profile by specifying the default output format, the default AWS Region to send commands to, and providing a [name for the profile](#) so you can reference this profile from among all those defined on the local computer. In the following example, the user enters a default Region, default output format, and the name of the profile. You can alternatively press <ENTER> to select any default values that are shown between the square brackets. The suggested profile name is the account ID number followed by an underscore followed by the role name.

```
CLI default client Region [None]: us-west-2<ENTER>  
CLI default output format [None]: json<ENTER>  
CLI profile name [123456789011_ReadOnly]: my-dev-profile<ENTER>
```

Note

If you specify `default` as the profile name, this profile becomes the one used whenever you run an AWS CLI command and do not specify a profile name.

7. A final message describes the completed profile configuration.

To use this profile, specify the profile name using `--profile`, as shown:

```
aws s3 ls --profile my-dev-profile
```

8. The previous example entries would result in a named profile in `~/.aws/config` that looks like the following example.

```
[profile my-dev-profile]  
sso_start_url = https://my-sso-portal.awsapps.com/start  
sso_region = us-east-1  
sso_account_id = 123456789011  
sso_role_name = readOnly  
region = us-west-2  
output = json
```

At this point, you have a profile that you can use to request temporary credentials. You must use the `aws sso login` command to actually request and retrieve the temporary credentials needed to run commands. For instructions, see [Use an IAM Identity Center named profile](#).

Manual configuration for legacy configuration

Automated token refresh isn't supported using the legacy non-refreshable configuration. We recommend using the SSO token configuration.

To manually add IAM Identity Center support to a named profile, you must add the following keys and values to the profile definition in the file `~/.aws/config` (Linux or macOS) or `%USERPROFILE%/.aws/config` (Windows).

- [sso_start_url](#)
- [sso_region](#)
- [sso_account_id](#)
- [sso_role_name](#)

You can include any other keys and values that are valid in the `.aws/config` file, such as [region](#), [output](#), or [s3](#). To prevent errors, don't include any credential related values, such as [role_arn](#) or [aws_secret_access_key](#).

The following is an example IAM Identity Center profile in `.aws/config`:

```
[profile my-sso-profile]
sso_start_url = https://my-sso-portal.awsapps.com/start
sso_region = us-west-2
sso_account_id = 111122223333
sso_role_name = SSOReadOnlyRole
region = us-west-2
output = json
```

Your profile for temporary credentials is complete.

To run commands, you must first use the `aws sso login` command to request and retrieve your temporary credentials. For instructions, see the next section, [Use an IAM Identity Center named profile](#). The authentication token is cached to disk under the `~/.aws/sso/cache` directory with a filename based on the `sso_start_url`.

Use an IAM Identity Center named profile

This topic describes how to use the AWS CLI to authenticate users with AWS IAM Identity Center (IAM Identity Center) to get credentials to run AWS CLI commands.

Note

Whether your credentials are temporary or automatically refreshing depends on how you previously configured your profile.

Topics

- [Prerequisites](#)
- [Signing in and getting credentials](#)
- [Running a command with your IAM Identity Center profile](#)
- [Signing out of your IAM Identity Center sessions](#)

Prerequisites

You've configured an IAM Identity Center profile. See [the section called "Configure automatic token refresh"](#) and [the section called "Configure legacy non-refreshable"](#) for more information.

Signing in and getting credentials

Note

The sign in process may prompt you to allow the AWS CLI access to your data. Since the AWS CLI is built on top of the SDK for Python, permission messages may contain variations of the `botocore` name.

After you configure a named profile, you can invoke it to request credentials from AWS. Before you can run an AWS CLI service command, you must retrieve and cache a set of credentials. To get these credentials, run the following command.

```
$ aws sso login --profile my-dev-profile
```


The AWS CLI opens your default browser and verifies your IAM Identity Center log in.

SSO authorization page has automatically been opened in your default browser. Follow the instructions in the browser to complete this authorization request. Successfully logged into Start URL: <https://my-sso-portal.awsapps.com/start>

If you are not currently signed into IAM Identity Center, you must provide your IAM Identity Center credentials.

If the AWS CLI can't open your browser, it prompts you to open it yourself and enter the specified code.

```
$ aws sso login --profile my-dev-profile
```

Using a browser, open the following URL:

<https://device.sso.us-west-2.amazonaws.com/>

and enter the following code:

QCFK-N451

The AWS CLI opens your default browser (or you manually open the browser of your choice) to the specified page, and enter the provided code. The webpage then prompts you for your IAM Identity Center credentials.

Your IAM Identity Center session credentials are cached. If these credentials are temporary, it includes an expiration timestamp and when they expire, the AWS CLI requests you to sign in to IAM Identity Center again.

If your IAM Identity Center credentials are valid, the AWS CLI uses them to securely retrieve AWS credentials for the IAM role specified in the profile.

Welcome, you have successfully signed-in to the AWS-CLI.

You can also specify which sso-session profile to use when logging in using the `--sso-session` parameter of the `aws sso login` command.

```
$ aws sso login --sso-session my-dev-session
```

Attempting to automatically open the SSO authorization page in your default browser. If the browser does not open or you wish to use a different device to authorize this request, open the following URL:

```
https://device.sso.us-west-2.amazonaws.com/
```

and enter the following code:

```
QCFK-N451
```

Successfully logged into Start URL: <https://cli-reinvent.awsapps.com/start>

Running a command with your IAM Identity Center profile

You can use these credentials to invoke an AWS CLI command with the associated named profile. The following example shows that the command was run under an assumed role that is part of the specified account.

```
$ aws sts get-caller-identity --profile my-dev-profile
{
  "UserId": "AROA12345678901234567:test-user@example.com",
  "Account": "123456789011",
  "Arn": "arn:aws:sts::123456789011:assumed-role/
AWSPeregrine_readOnly_12321abc454d123/test-user@example.com"
}
```

As long as you signed in to IAM Identity Center and those cached credentials are not expired, the AWS CLI automatically renews expired AWS credentials when needed. However, if your IAM Identity Center credentials expire, you must explicitly renew them by logging in to your IAM Identity Center account again.

```
$ aws s3 ls --profile my-sso-profile
```

Your short-term credentials have expired. Please sign-in to renew your credentials
SSO authorization page has automatically been opened in your default browser.
Follow the instructions in the browser to complete this authorization request.

Signing out of your IAM Identity Center sessions

When you are done using your IAM Identity Center profiles, you can choose to do nothing and let the AWS temporary credentials and your IAM Identity Center credentials expire. However, you can also choose to run the following command to immediately delete all cached credentials in the SSO credential cache folder and all AWS temporary credentials that were based on the IAM Identity Center credentials. This makes those credentials unavailable to be used for any future command.

```
$ aws sso logout
```



```
[profile user1]
region=us-east-1
output=text
```

When the SDK creates a service client, it will access these temporary credentials and use them for each request. The settings for the IAM role chosen in step 2a determine [how long the temporary credentials are valid](#). The maximum duration is twelve hours.

Repeat these steps each time your credentials expire.

Use an IAM role in the AWS CLI

An [AWS Identity and Access Management \(IAM\) role](#) is an authorization tool that lets a user gain additional (or different) permissions, or get permissions to perform actions in a different AWS account.

Topics

- [Prerequisites](#)
- [Overview of using IAM roles](#)
- [Configuring and using a role](#)
- [Using multi-factor authentication](#)
- [Cross-account roles and external ID](#)
- [Specifying a role session name for easier auditing](#)
- [Assume role with web identity](#)
- [Clearing cached credentials](#)

Prerequisites

To run the `iam` commands, you need to install and configure the AWS CLI. For more information, see [the section called “Install/Update”](#).

Overview of using IAM roles

You can configure the AWS Command Line Interface (AWS CLI) to use an IAM role by defining a profile for the role in the `~/.aws/config` file.

The following example shows a role profile named `marketingadmin`. If you run commands with `--profile marketingadmin` (or specify it with the [AWS_PROFILE environment variable](#)), the AWS CLI uses the credentials defined in a separate profile `user1` to assume the role with the Amazon Resource Name (ARN) `arn:aws:iam::123456789012:role/marketingadminrole`. You can run any operations that are allowed by the permissions assigned to that role.

```
[profile marketingadmin]
role_arn = arn:aws:iam::123456789012:role/marketingadminrole
source_profile = user1
```

You can then specify a `source_profile` that points to a separate named profile that contains user credentials with permission to use the role. In the previous example, the `marketingadmin` profile uses the credentials in the `user1` profile. When you specify that an AWS CLI command is to use the profile `marketingadmin`, the AWS CLI automatically looks up the credentials for the linked `user1` profile and uses them to request temporary credentials for the specified IAM role. The CLI uses the [sts:AssumeRole](#) operation in the background to accomplish this. Those temporary credentials are then used to run the requested AWS CLI command. The specified role must have attached IAM permission policies that allow the requested AWS CLI command to run.

To run a AWS CLI command from within an Amazon Elastic Compute Cloud (Amazon EC2) instance or an Amazon Elastic Container Service (Amazon ECS) container, you can use an IAM role attached to the instance profile or the container. If you specify no profile or set no environment variables, that role is used directly. This enables you to avoid storing long-lived access keys on your instances. You can also use those instance or container roles only to get credentials for another role. To do this, you use `credential_source` (instead of `source_profile`) to specify how to find the credentials. The `credential_source` attribute supports the following values:

- `Environment` – Retrieves the source credentials from environment variables.
- `Ec2InstanceMetadata` – Uses the IAM role attached to the Amazon EC2 instance profile.
- `EcsContainer` – Uses the IAM role attached to the Amazon ECS container.

The following example shows the same `marketingadminrole` role used by referencing an Amazon EC2 instance profile.

```
[profile marketingadmin]
role_arn = arn:aws:iam::123456789012:role/marketingadminrole
credential_source = Ec2InstanceMetadata
```

When you invoke a role, you have additional options that you can require, such as the use of multi-factor authentication and an External ID (used by third-party companies to access their clients' resources). You can also specify unique role session names that can be more easily audited in AWS CloudTrail logs.

Configuring and using a role

When you run commands using a profile that specifies an IAM role, the AWS CLI uses the source profile's credentials to call AWS Security Token Service (AWS STS) and request temporary credentials for the specified role. The user in the source profile must have permission to call `sts:assume-role` for the role in the specified profile. The role must have a trust relationship that allows the user in the source profile to use the role. The process of retrieving and then using temporary credentials for a role is often referred to as *assuming the role*.

You can create a role in IAM with the permissions that you want users to assume by following the procedure under [Creating a Role to Delegate Permissions to an IAM user](#) in the *AWS Identity and Access Management User Guide*. If the role and the source profile's user are in the same account, you can enter your own account ID when configuring the role's trust relationship.

After creating the role, modify the trust relationship to allow the user to assume it.

The following example shows a trust policy that you could attach to a role. This policy allows the role to be assumed by any user in the account 123456789012, *if* the administrator of that account explicitly grants the `sts:AssumeRole` permission to the user.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:root"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

The trust policy doesn't actually grant permissions. The administrator of the account must delegate the permission to assume the role to individual users by attaching a policy with the appropriate

permissions. The following example shows a policy that you can attach to a user that allows the user to assume only the `marketingadminrole` role. For more information about granting a user access to assume a role, see [Granting a User Permission to Switch Roles](#) in the *IAM User Guide*.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "arn:aws:iam::123456789012:role/marketingadminrole"
    }
  ]
}
```

The user doesn't need to have additional permissions to run the AWS CLI commands using the role profile. Instead, the permissions to run the command come from those attached to the *role*. You attach permission policies to the role to specify which actions can be performed against which AWS resources. For more information about attaching permissions to a role (which works identically to a user), see [Changing Permissions for an IAM user](#) in the *IAM User Guide*.

Now that you have the role profile, role permissions, role trust relationship, and user permissions correctly configured, you can use the role at the command line by invoking the `--profile` option. For example, the following calls the Amazon S3 `ls` command using the permissions attached to the `marketingadmin` role as defined by the example at the beginning of this topic.

```
$ aws s3 ls --profile marketingadmin
```

To use the role for several calls, you can set the `AWS_PROFILE` environment variable for the current session from the command line. While that environment variable is defined, you don't have to specify the `--profile` option on each command.

Linux or macOS

```
$ export AWS_PROFILE=marketingadmin
```

Windows

```
C:\> setx AWS_PROFILE marketingadmin
```

For more information about configuring users and roles, see [Users and Groups](#) and [Roles](#) in the *IAM User Guide*.

Using multi-factor authentication

For additional security, you can require that users provide a one-time key generated from a multi-factor authentication (MFA) device, a U2F device, or mobile app when they attempt to make a call using the role profile.

First, you can choose to modify the trust relationship on the IAM role to require MFA. This prevents anyone from using the role without first authenticating by using MFA. For an example, see the `Condition` line in the following example. This policy allows the user named `anika` to assume the role the policy is attached to, but only if they authenticate by using MFA.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::123456789012:user/anika" },
      "Action": "sts:AssumeRole",
      "Condition": { "Bool": { "aws:multifactorAuthPresent": true } }
    }
  ]
}
```

Next, add a line to the role profile that specifies the ARN of the user's MFA device. The following sample config file entries show two role profiles that both use the access keys for the user `anika` to request temporary credentials for the role `cli-role`. The user `anika` has permissions to assume the role, granted by the role's trust policy.

```
[profile role-without-mfa]
region = us-west-2
role_arn= arn:aws:iam::128716708097:role/cli-role
source_profile=cli-user

[profile role-with-mfa]
region = us-west-2
role_arn= arn:aws:iam::128716708097:role/cli-role
source_profile = cli-user
```



```
mfa_serial = arn:aws:iam::128716708097:mfa/cli-user

[profile cli-user]
region = us-west-2
output = json
```

The `mfa_serial` setting can take an ARN, as shown, or the serial number of a hardware MFA token.

The first profile, `role-without-mfa`, doesn't require MFA. However, because the previous example trust policy attached to the role requires MFA, any attempt to run a command with this profile fails.

```
$ aws iam list-users --profile role-without-mfa
```

```
An error occurred (AccessDenied) when calling the AssumeRole operation: Access denied
```

The second profile entry, `role-with-mfa`, identifies an MFA device to use. When the user attempts to run a AWS CLI command with this profile, the AWS CLI prompts the user to enter the one-time password (OTP) that the MFA device provides. If the MFA authentication succeeds, the command performs the requested operation. The OTP is not displayed on the screen.

```
$ aws iam list-users --profile role-with-mfa
Enter MFA code for arn:aws:iam::123456789012:mfa/cli-user:
{
  "Users": [
    {
      ...
    }
  ]
}
```

Cross-account roles and external ID

You can enable users to use roles that belong to different accounts by configuring the role as a cross-account role. During role creation, set the role type to **Another AWS account**, as described in [Creating a Role to Delegate Permissions to an IAM user](#). Optionally, select **Require MFA**. **Require MFA** configures the appropriate condition in the trust relationship, as described in [Using multi-factor authentication](#).

If you use an [external ID](#) to provide additional control over who can use a role across accounts, you must also add the `external_id` parameter to the role profile. You typically use this only when the other account is controlled by someone outside your company or organization.

```
[profile crossaccountrole]
role_arn = arn:aws:iam::234567890123:role/SomeRole
source_profile = default
mfa_serial = arn:aws:iam::123456789012:mfa/saanvi
external_id = 123456
```

Specifying a role session name for easier auditing

When many individuals share a role, auditing becomes more of a challenge. You want to associate each operation invoked with the individual who invoked the action. However, when the individual uses a role, the assumption of the role by the individual is a separate action from the invoking of an operation, and you must manually correlate the two.

You can simplify this by specifying unique role session names when users assume a role. You do this by adding a `role_session_name` parameter to each named profile in the config file that specifies a role. The `role_session_name` value is passed to the `AssumeRole` operation and becomes part of the ARN for the role session. It is also included in the AWS CloudTrail logs for all logged operations.

For example, you could create a role-based profile as follows.

```
[profile namedsessionrole]
role_arn = arn:aws:iam::234567890123:role/SomeRole
source_profile = default
role_session_name = Session_Maria_Garcia
```

This results in the role session having the following ARN.

```
arn:aws:iam::234567890123:assumed-role/SomeRole/Session_Maria_Garcia
```

Also, all AWS CloudTrail logs include the role session name in the information captured for each operation.

Assume role with web identity

You can configure a profile to indicate that the AWS CLI should assume a role using [web identity federation and Open ID Connect \(OIDC\)](#). When you specify this in a profile, the AWS CLI automatically makes the corresponding AWS STS `AssumeRoleWithWebIdentity` call for you.

Note

When you specify a profile that uses an IAM role, the AWS CLI makes the appropriate calls to retrieve temporary credentials. These credentials are stored in `~/.aws/cli/` cache. Subsequent AWS CLI commands that specify the same profile use the cached temporary credentials until they expire. At that point, the AWS CLI automatically refreshes the credentials.

To retrieve and use temporary credentials using web identity federation, you can specify the following configuration values in a shared profile.

[role_arn](#)

Specifies the ARN of the role to assume.

[web_identity_token_file](#)

Specifies the path to a file which contains an OAuth 2.0 access token or OpenID Connect ID token that is provided by the identity provider. The AWS CLI loads this file and passes its content as the `WebIdentityToken` argument of the `AssumeRoleWithWebIdentity` operation.

[role_session_name](#)

Specifies an optional name applied to this assume-role session.

The following is an example of the minimal amount of configuration needed to configure an assume role with web identity profile.

```
# In ~/.aws/config

[profile web-identity]
role_arn=arn:aws:iam:123456789012:role/RoleNameToAssume
web_identity_token_file=/path/to/a/token
```

You can also provide this configuration by using [environment variables](#).

AWS_ROLE_ARN

The ARN of the role to assume.

AWS_WEB_IDENTITY_TOKEN_FILE

The path to the web identity token file.

AWS_ROLE_SESSION_NAME

The name applied to this assume-role session.

Note

These environment variables currently apply only to the assume role with web identity provider. They don't apply to the general assume role provider configuration.

Clearing cached credentials

When you use a role, the AWS CLI caches the temporary credentials locally until they expire. The next time you try to use them, the AWS CLI attempts to renew them on your behalf.

If your role's temporary credentials are [revoked](#), they are not renewed automatically, and attempts to use them fail. However, you can delete the cache to force the AWS CLI to retrieve new credentials.

Linux or macOS

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\aws\cli\cache
```

Authenticate with IAM user credentials

Warning

To avoid security risks, don't use IAM users for authentication when developing purpose-built software or working with real data. Instead, use federation with an identity provider such as [AWS IAM Identity Center](#).

This section explains how to configure basic settings with an IAM user. These include your security credentials using the `config` and `credentials` files. To instead see configuration instructions for AWS IAM Identity Center, see [the section called "IAM Identity Center authentication"](#).

Topics

- [Step 1: Create your IAM user](#)
- [Step 2: Get your access keys](#)
- [Configure the AWS CLI](#)
 - [Using aws configure](#)
 - [Importing access keys via .CSV file](#)
 - [Directly editing the config and credentials files](#)

Step 1: Create your IAM user

Create your IAM user by following the [Creating IAM users \(console\)](#) procedure in the *IAM User Guide*.

- For **Permission options**, choose **Attach policies directly** for how you want to assign permissions to this user.
- Most "Getting Started" SDK tutorials use the Amazon S3 service as an example. To provide your application with full access to Amazon S3, select the `AmazonS3FullAccess` policy to attach to this user.

Step 2: Get your access keys

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane of the IAM console, select **Users** and then select the **User name** of the user that you created previously.
3. On the user's page, select the **Security credentials** page. Then, under **Access keys**, select **Create access key**.
4. For **Create access key Step 1**, choose **Command Line Interface (CLI)**.
5. For **Create access key Step 2**, enter an optional tag and select **Next**.

6. For **Create access key Step 3**, select **Download .csv file** to save a .csv file with your IAM user's access key and secret access key. You need this information for later.
7. Select Done.

Configure the AWS CLI

For general use, the AWS CLI needs the following pieces of information:

- Access key ID
- Secret access key
- AWS Region
- Output format

The AWS CLI stores this information in a *profile* (a collection of settings) named `default` in the `credentials` file. By default, the information in this profile is used when you run an AWS CLI command that doesn't explicitly specify a profile to use. For more information on the `credentials` file, see [Configuration and credential file settings](#).

To configure the AWS CLI, use one of the following procedures:

Topics

- [Using `aws configure`](#)
- [Importing access keys via .CSV file](#)
- [Directly editing the config and credentials files](#)

Using `aws configure`

For general use, the `aws configure` command is the fastest way to set up your AWS CLI installation. This configure wizard prompts you for each piece of information you need to get started. Unless otherwise specified by using the `--profile` option, the AWS CLI stores this information in the `default` profile.

The following example configures a `default` profile using sample values. Replace them with your own values as described in the following sections.

```
$ aws configure
```

```
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
Default region name [None]: us-west-2
Default output format [None]: json
```

The following example configures a profile named `userprod` using sample values. Replace them with your own values as described in the following sections.

```
$ aws configure --profile userprod
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
Default region name [None]: us-west-2
Default output format [None]: json
```

Importing access keys via .CSV file

Instead of using `aws configure` to enter in access keys, you can import the plain text `.csv` file you downloaded after you created your access keys.

The `.csv` file must contain the following headers.

- User Name - This column must be added to your `.csv`. This is used to create the profile name used in the the config and `credentials` files when you import.
- Access key ID
- Secret access key

Note

During initial access keys creation, once you close the **Download .csv file** dialog box, you cannot access your secret access key after you close the dialog box. If you need a `.csv` file, you'll need to create one yourself with the required headers and your stored access keys information. If you do not have access to your access keys information, you need to create a new access keys.

To import the `.csv` file, use the `aws configure import` command with the `--csv` option as follows:

```
$ aws configure import --csv file://credentials.csv
```

For more information, see [aws_configure_import](#).

Directly editing the config and credentials files

To directly edit the config and credentials files, perform the following.

1. Create or open the shared AWS credentials file. This file is `~/.aws/credentials` on Linux and macOS systems, and `%USERPROFILE%\.aws\credentials` on Windows. For more information, see [the section called “Configuration and credential file settings”](#).
2. Add the following text to the shared credentials file. Replace the sample values in the `.csv` file that you downloaded earlier and save the file.

```
[default]
aws_access_key_id = AKIAIOSFODNN7EXAMPLE
aws_secret_access_key = wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
```

Use credentials for Amazon EC2 instance metadata

When you run the AWS CLI from within an Amazon Elastic Compute Cloud (Amazon EC2) instance, you can simplify providing credentials to your commands. Each Amazon EC2 instance contains metadata that the AWS CLI can directly query for temporary credentials. When an IAM role is attached to the instance, the AWS CLI automatically and securely retrieves the credentials from the instance metadata.

To disable this service, use the [AWS_EC2_METADATA_DISABLED](#) environment variable.

Topics

- [Prerequisites](#)
- [Configuring a profile for Amazon EC2 metadata](#)

Prerequisites

To use Amazon EC2 credentials with the AWS CLI, you need to complete the following:

- Install and configure the AWS CLI. For more information, see [the section called “Install/Update” and Authentication and access credentials](#).
- You understand configuration files and named profiles. For more information, see [Configuration and credential file settings](#).
- You've created an AWS Identity and Access Management (IAM) role that has access to the resources needed, and attached that role to the Amazon EC2 instance when you launch it. For more information, see [IAM policies for Amazon EC2](#) in the *Amazon EC2 User Guide* and [Granting Applications That Run on Amazon EC2 Instances Access to AWS Resources](#) in the *IAM User Guide*.

Configuring a profile for Amazon EC2 metadata

To specify that you want to use the credentials available in the hosting Amazon EC2 instance profile, use the following syntax in the named profile in your configuration file. See the following steps for more instructions.

```
[profile profilename]  
role_arn = arn:aws:iam::123456789012:role/rolename  
credential_source = Ec2InstanceMetadata  
region = region
```

1. Create a profile in your configuration file.

```
[profile profilename]
```

2. Add your IAM arn role that has access to the resources needed.

```
role_arn = arn:aws:iam::123456789012:role/rolename
```

3. Specify Ec2InstanceMetadata as your credential source.

```
credential_source = Ec2InstanceMetadata
```

4. Set your Region.

```
region = region
```

Example

The following example assumes the *marketingadminrole* role and uses the *us-west-2* Region in an Amazon EC2 instance profile named *marketingadmin*.

```
[profile marketingadmin]
role_arn = arn:aws:iam::123456789012:role/marketingadminrole
credential_source = Ec2InstanceMetadata
region = us-west-2
```

Source credentials with an external process

Warning

This topic discusses sourcing credentials from an external process. This could be a security risk if the command to generate the credentials becomes accessible by non-approved processes or users. We recommend that you use the supported, secure alternatives provided by the AWS CLI and AWS to reduce the risk of compromising your credentials. Ensure that you secure the `config` file and any supporting files and tools to prevent disclosure.

Ensure that your custom credential tool does not write any secret information to `StdErr` because the SDKs and AWS CLI can capture and log such information, potentially exposing it to unauthorized users.

If you have a method to generate or look up credentials that isn't directly supported by the AWS CLI, you can configure the AWS CLI to use it by configuring the `credential_process` setting in the `config` file.

For example, you might include an entry similar to the following in the `config` file.

```
[profile developer]
credential_process = /opt/bin/awscreds-custom --username helen
```

Syntax

To create this string in a way that is compatible with any operating system, follow these rules:

- If the path or file name contains a space, surround the complete path and file name with double-quotation marks (" "). The path and file name can consist of only the characters: A-Z a-z 0-9 - _ . space

- If a parameter name or a parameter value contains a space, surround that element with double-quotation marks (" "). Surround only the name or value, not the pair.
- Do not include any environment variables in the strings. For example, you can't include \$HOME or %USERPROFILE%.
- Do not specify the home folder as ~. You must specify the full path.

Example for Windows

```
credential_process = "C:\Path\To\credentials.cmd" parameterWithoutSpaces "parameter with spaces"
```

Example for Linux or macOS

```
credential_process = "/Users/Dave/path/to/credentials.sh" parameterWithoutSpaces "parameter with spaces"
```

Expected output from the Credentials program

The AWS CLI runs the command as specified in the profile and then reads data from STDOUT. The command you specify must generate JSON output on STDOUT that matches the following syntax.


```
{
  "Version": 1,
  "AccessKeyId": "an AWS access key",
  "SecretAccessKey": "your AWS secret access key",
  "SessionToken": "the AWS session token for temporary credentials",
  "Expiration": "ISO8601 timestamp when the credentials expire"
}
```

Note

As of this writing, the `Version` key must be set to 1. This might increment over time as the structure evolves.

The `Expiration` key is an [ISO8601](#) formatted timestamp. If the `Expiration` key is not present in the tool's output, the CLI assumes that the credentials are long-term credentials that do

not refresh. Otherwise the credentials are considered temporary credentials and are refreshed automatically by rerunning the `credential_process` command before they expire.

 **Note**

The AWS CLI does **not** cache external process credentials the way it does assume-role credentials. If caching is required, you must implement it in the external process.

The external process can return a non-zero return code to indicate that an error occurred while retrieving the credentials.

Use the AWS CLI

This section provides information about general use, common features, and options available in the AWS Command Line Interface (AWS CLI), beyond what is written in the Configuration [the section called “Endpoints”](#) section. Instructions include how to write a command, basic structure, formatting, filtering, and locating the help content or documentation for a command.

For AWS service specific examples, see [Code examples](#) or the [AWS CLI version 2 reference guide](#).

Note

By default, the AWS CLI sends requests to AWS services by using HTTPS on TCP port 443. To use the AWS CLI successfully, you must be able to make outbound connections on TCP port 443.

Topics in this guide

- [Get help with the AWS CLI](#)
- [Command structure in the AWS CLI](#)
- [Specify parameter values for the AWS CLI](#)
- [Have the AWS CLI prompt you for commands](#)
- [Control command output from the AWS CLI](#)
- [Return codes from the AWS CLI](#)
- [Interactive commands using the AWS CLI wizards](#)
- [Create and use AWS CLI command shortcuts called aliases](#)

Get help with the AWS CLI

This topic describes how to access help content for the AWS Command Line Interface (AWS CLI).

Topics

- [The built-in AWS CLI help command](#)
- [AWS CLI reference guide](#)
- [API documentation](#)

- [Troubleshooting errors](#)
- [Additional help](#)

The built-in AWS CLI help command

You can get help with any command when using the AWS Command Line Interface (AWS CLI). To do so, simply type `help` at the end of a command name.

For example, the following command displays help for the general AWS CLI options and the available top-level commands.

```
$ aws help
```

The following command displays the available Amazon Elastic Compute Cloud (Amazon EC2) specific commands.

```
$ aws ec2 help
```

The following example displays detailed help for the Amazon EC2 `DescribeInstances` operation. The help includes descriptions of its input parameters, available filters, and what is included as output. It also includes examples showing how to type common variations of the command.

```
$ aws ec2 describe-instances help
```

The help for each command is divided into six sections:

Name

The name of the command.

```
NAME
    describe-instances -
```

Description

A description of the API operation that the command invokes.

DESCRIPTION

Describes one or more of your instances.

If you specify one or more instance IDs, Amazon EC2 returns information for those instances. If you do not specify instance IDs, Amazon EC2 returns information for all relevant instances. If you specify an instance ID that is not valid, an error is returned. If you specify an instance that you do not own, it is not included in the returned results.

...

Synopsis

The basic syntax for using the command and its options. If an option is shown in square brackets, it's optional, has a default value, or has an alternative option that you can use.

SYNOPSIS

```
describe-instances
[--dry-run | --no-dry-run]
[--instance-ids <value>]
[--filters <value>]
[--cli-input-json <value>]
[--starting-token <value>]
[--page-size <value>]
[--max-items <value>]
[--generate-cli-skeleton]
```

For example, `describe-instances` has a default behavior that describes **all** instances in the current account and AWS Region. You can optionally specify a list of `instance-ids` to describe one or more instances; `dry-run` is an optional Boolean flag that doesn't take a value. To use a Boolean flag, specify either shown value, in this case `--dry-run` or `--no-dry-run`. Likewise, `--generate-cli-skeleton` doesn't take a value. If there are conditions on an option's use, they are described in the **OPTIONS** section, or shown in the examples.

Options

A description of each of the options shown in the synopsis.

OPTIONS

```
--dry-run | --no-dry-run (boolean)
    Checks whether you have the required permissions for the action,
```

```
without actually making the request, and provides an error response.
If you have the required permissions, the error response is DryRun-
Operation . Otherwise, it is UnauthorizedOperation .
```

```
--instance-ids (list)
```

```
One or more instance IDs.
```

```
Default: Describes all your instances.
```

```
...
```

Examples

Examples showing the usage of the command and its options. If no example is available for a command or use case that you need, request one using the feedback link on this page, or in the AWS CLI command reference on the help page for the command.

EXAMPLES

To describe an Amazon EC2 instance

Command:

```
aws ec2 describe-instances --instance-ids i-5203422c
```

To describe all instances with the instance type m1.small

Command:

```
aws ec2 describe-instances --filters "Name=instance-type,Values=m1.small"
```

To describe all instances with an Owner tag

Command:

```
aws ec2 describe-instances --filters "Name=tag-key,Values=Owner"
```

```
...
```

Output

Descriptions of each of the fields and data types included in the response from AWS.

For `describe-instances`, the output is a list of reservation objects, each of which contains several fields and objects that contain information about the instances associated with it. This

information comes from the [API documentation for the reservation data type](#) used by Amazon EC2.

OUTPUT

Reservations -> (list)

One or more reservations.

(structure)

Describes a reservation.

ReservationId -> (string)

The ID of the reservation.

OwnerId -> (string)

The ID of the AWS account that owns the reservation.

RequesterId -> (string)

The ID of the requester that launched the instances on your behalf (for example, AWS Management Console or Auto Scaling).

Groups -> (list)

One or more security groups.

(structure)

Describes a security group.

GroupName -> (string)

The name of the security group.

GroupId -> (string)

The ID of the security group.

Instances -> (list)

One or more instances.

(structure)

Describes an instance.

InstanceId -> (string)

The ID of the instance.

ImageId -> (string)

The ID of the AMI used to launch the instance.

```
State -> (structure)
    The current state of the instance.

Code -> (integer)
    The low byte represents the state. The high byte
    is an opaque internal value and should be ignored.
```

...

When the AWS CLI renders the output into JSON, it becomes an array of reservation objects, similar to the following example.

```
{
  "Reservations": [
    {
      "OwnerId": "012345678901",
      "ReservationId": "r-4c58f8a0",
      "Groups": [],
      "RequesterId": "012345678901",
      "Instances": [
        {
          "Monitoring": {
            "State": "disabled"
          },
          "PublicDnsName": "ec2-52-74-16-12.us-
west-2.compute.amazonaws.com",
          "State": {
            "Code": 16,
            "Name": "running"
          }
        }
      ]
    }
  ]
}
```

...

Each reservation object contains fields describing the reservation and an array of instance objects, each with its own fields (for example, `PublicDnsName`) and objects (for example, `State`) that describe it.

Windows users

You can *pipe* (`|`) the output of the help command to the more command to view the help file one page at a time. Press the space bar or **PgDn** to view more of the document, and **q** to quit.

```
C:\> aws ec2 describe-instances help | more
```

AWS CLI reference guide

The help files contain links that cannot be viewed or navigated to from the command line. You can view and interact with these links by using the online [AWS CLI version 2 reference guide](#). The reference also contains the help content for all AWS CLI commands. The descriptions are presented for easy navigation and viewing on mobile, tablet, or desktop screens.

API documentation

All commands in the AWS CLI correspond to requests made to an AWS service's public API. Each service with a public API has an API reference that can be found on the service's home page on the [AWS Documentation website](#). The content for an API reference varies based on how the API is constructed and which protocol is used. Typically, an API reference contains detailed information about the operations supported by the API, the data sent to and from the service, and any error conditions that the service can report.

API Documentation Sections

- **Actions** – Detailed information on each operation and its parameters (including constraints on length or content, and default values). It lists the errors that can occur for this operation. Each operation corresponds to a subcommand in the AWS CLI.
- **Data Types** – Detailed information about structures that a command might require as a parameter, or return in response to a request.
- **Common Parameters** – Detailed information about the parameters that are shared by all of action for the service.
- **Common Errors** – Detailed information about errors that can be returned by any of the service's operations.

The name and availability of each section can vary, depending on the service.

Service-specific CLIs

Some services have a separate CLI that dates from before a single AWS CLI was created to work with all services. These service-specific CLIs have separate documentation that is linked from the service's documentation page. Documentation for service-specific CLIs do not apply to the AWS CLI.

Troubleshooting errors

For help diagnosing and fixing AWS CLI errors, see [Troubleshoot errors](#).

Additional help

For additional help with your AWS CLI issues, visit the [AWS CLI community](#) on *GitHub*.

Command structure in the AWS CLI

This topic covers how AWS Command Line Interface (AWS CLI) command is structured, and how to use wait commands.

Topics

- [Command structure](#)
- [Wait commands](#)

Command structure

The AWS CLI uses a multipart structure on the command line that must be specified in this order:

1. The base call to the `aws` program.
2. The top-level *command*, which typically corresponds to an AWS service supported by the AWS CLI.
3. The *subcommand* that specifies which operation to perform.
4. General AWS CLI options or parameters required by the operation. You can specify these in any order as long as they follow the first three parts. If an exclusive parameter is specified multiple times, only the *last value* applies.

```
$ aws <command> <subcommand> [options and parameters]
```

Parameters can take various types of input values, such as numbers, strings, lists, maps, and JSON structures. What is supported is dependent upon the command and subcommand you specify.

Examples

Amazon S3

The following example lists all of your Amazon S3 buckets.

```
$ aws s3 ls
2018-12-11 17:08:50 my-bucket
2018-12-14 14:55:44 my-bucket2
```

For more information on the Amazon S3 commands, see [aws s3](#) in the *AWS CLI Command Reference*.

AWS CloudFormation

The following [create-change-set](#) command example changes the cloudformation stack name to *my-change-set*.

```
$ aws cloudformation create-change-set --stack-name my-stack --change-set-name my-change-set
```

For more information on the AWS CloudFormation commands, see [aws cloudformation](#) in the *AWS CLI Command Reference*.

Wait commands

Some AWS services have wait commands available. Any command that uses `aws wait` usually waits until a command is complete before it moves on to the next step. This is especially useful for multipart commands or scripting, as you can use a wait command to prevent moving to subsequent steps if the wait command fails.

The AWS CLI uses a multipart structure on the command line for the `wait` command that must be specified in this order:

1. The base call to the `aws` program.

2. The top-level *command*, which typically corresponds to an AWS service supported by the AWS CLI.
3. The *wait* command.
4. The *subcommand* that specifies which operation to perform.
5. General CLI options or parameters required by the operation. You can specify these in any order as long as they follow the first three parts. If an exclusive parameter is specified multiple times, only the *last value* applies.

```
$ aws <command> wait <subcommand> [options and parameters]
```

Parameters can take various types of input values, such as numbers, strings, lists, maps, and JSON structures. What is supported is dependent upon the command and subcommand you specify.

Note

Not every AWS service supports *wait* commands. See the [AWS CLI version 2 reference guide](#) to see if your service supports *wait* commands.

Examples

AWS CloudFormation

The following [wait change-set-create-complete](#) command examples pauses and resumes only after it can confirm that the *my-change-set* change set in the *my-stack* stack is ready to run.

```
$ aws cloudformation wait change-set-create-complete --stack-name my-stack --change-set-name my-change-set
```

For more information on the AWS CloudFormation *wait* commands, see [wait](#) in the *AWS CLI Command Reference*.

AWS CodeDeploy

The following [wait deployment-successful](#) command examples pauses until the *d-A1B2C3111* deployment completes successfully.

```
$ aws deploy wait deployment-successful --deployment-id d-A1B2C3111
```

For more information on the AWS CodeDeploy wait commands, see [wait](#) in the *AWS CLI Command Reference*.

Specify parameter values for the AWS CLI

Many parameters used in the AWS Command Line Interface (AWS CLI) are simple string or numeric values, such as the key-pair name `my-key-pair` in the following example.

```
$ aws ec2 create-key-pair --key-name my-key-pair
```

Formatting between terminals can vary. For example, most terminals are case sensitive but Powershell is case insensitive. This means the two following command examples would yield different results for case sensitive terminals as they view `MyFile*.txt` and `myfile*.txt` as **different** parameters.

However, PowerShell would process these requests as the same as it sees `MyFile*.txt` and `myfile*.txt` as the **same** parameters.

```
$ aws s3 cp . s3://my-bucket/path --include "MyFile*.txt"  
$ aws s3 cp . s3://my-bucket/path --include "myfile*.txt"
```

For more information on PowerShell's case insensitivity, see [about_Case-Sensitivity](#) in the *PowerShell documentation*.

Sometimes you need to use quotation marks or literals around strings that include special or space characters. The rules around this formatting can also vary between terminals. For more information about using quotation marks around complex parameters, see [Quotation marks with strings in the AWS CLI](#).

Parameter topics

- [Common AWS CLI parameter types](#)
- [Quotation marks with strings in the AWS CLI](#)
- [Load AWS CLI parameters from a file](#)
- [AWS CLI skeletons and input files](#)
- [Use shorthand syntax with the AWS CLI](#)

Common AWS CLI parameter types

This section describes some of the common parameter types and the typical required format.

If you are having trouble formatting a parameter for a specific command, check the help by entering **help** after the command name. The help for each subcommand includes an option's name and description. The option's parameter type is listed in parentheses. For more information on viewing help, see [the section called "Get Help"](#).

Parameter types include:

- [String](#)
- [Timestamp](#)
- [List](#)
- [Boolean](#)
- [Integer](#)
- [Binary / blob \(binary large object\) and streaming blob](#)
- [Map](#)
- [Document](#)

String

String parameters can contain alphanumeric characters, symbols, and white spaces from the [ASCII](#) character set. Strings that contain white spaces must be surrounded by quotation marks. We recommend that you don't use symbols or white spaces other than the standard space character and to observe your terminal's [quoting rules](#) to prevent unexpected results.

Some string parameters can accept binary data from a file. See [Binary files](#) for an example.

Timestamp

Timestamps are formatted according to the [ISO 8601](#) standard. These are often referred to as "DateTime" or "Date" parameters.

```
$ aws ec2 describe-spot-price-history --start-time 2014-10-13T19:00:00Z
```

Acceptable formats include:

- *YYYY-MM-DDThh:mm:ss.sssTZD (UTC)*, for example, 2014-10-01T20:30:00.000Z

- *YYYY-MM-DDThh:mm:ss.sssTZD (with offset)*, for example, 2014-10-01T12:30:00.000-08:00
- *YYYY-MM-DD*, for example, 2014-10-01
- Unix time in seconds, for example, 1412195400. This is sometimes referred to as [Unix Epoch time](#) and represents the number of seconds since midnight, January 1, 1970 UTC.

By default, the AWS CLI version 2 translates all *response* DateTime values to ISO 8601 format.

You can set the timestamp format by using the [cli_timestamp_format](#) file setting.

List

One or more strings separated by spaces. If any of the string items contain a space, you must put quotation marks around that item. Observe your terminal's [quoting rules](#) to prevent unexpected results.

```
$ aws ec2 describe-spot-price-history --instance-types m1.xlarge m1.medium
```

Boolean

Binary flag that turns an option on or off. For example, `ec2 describe-spot-price-history` has a Boolean `--dry-run` parameter that, when specified, validates the query with the service without actually running the query.

```
$ aws ec2 describe-spot-price-history --dry-run
```

The output indicates whether the command was well formed. This command also includes a `--no-dry-run` version of the parameter that you can use to explicitly indicate that the command should be run normally. Including it isn't necessary because this is the default behavior.

Integer

An unsigned, whole number.

```
$ aws ec2 describe-spot-price-history --max-items 5
```

Binary / blob (binary large object) and streaming blob

In the AWS CLI, you can pass a binary value as a string directly on the command line. There are two types of blobs:

- [Blob](#)
- [Streaming blob](#)

Blob

To pass a value to a parameter with type `blob`, you must specify a path to a local file that contains the binary data using the `fileb://` prefix. Files referenced using the `fileb://` prefix are always treated as raw unencoded binary. The specified path is interpreted as being relative to the current working directory. For example, the `--plaintext` parameter for `aws kms encrypt` is a blob.

```
$ aws kms encrypt \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --plaintext fileb://ExamplePlaintextFile \  
  --output text \  
  --query CiphertextBlob | base64 \  
  --decode > ExampleEncryptedFile
```

Note

For backwards compatibility, you can use the `file://` prefix. There are two formats used based on the file setting [cli_binary_format](#) or `--cli-binary-format` command line option:

- Default for the AWS CLI version 2. If the setting's value is `base64`, files referenced using the `file://` prefix are treated as base64-encoded text.
- Default for the AWS CLI version 1. If the setting's value is `raw-in-base64-out`, files referenced using the `file://` prefix is read as text and then the AWS CLI attempts to encode it to binary.

For more information, see the file setting [cli_binary_format](#) or `--cli-binary-format` command line option.

Streaming blob

Streaming blobs such as `aws cloudsearchdomain upload-documents` do not use prefixes. Instead, streaming blob parameters are formatted using the direct file path. The following example uses the direct file path `document-batch.json` for the `aws cloudsearchdomain upload-documents` command:

```
$ aws cloudsearchdomain upload-documents \  
  --endpoint-url https://doc-my-domain.us-west-1.cloudsearch.amazonaws.com \  
  --content-type application/json \  
  --documents document-batch.json
```

Map

A set of key-value pairs specified in JSON or by using the CLI's [shorthand syntax](#). The following JSON example reads an item from an Amazon DynamoDB table named *my-table* with a map parameter, `--key`. The parameter specifies the primary key named *id* with a number value of *1* in a nested JSON structure.

For more advanced JSON usage in a command line, consider using a command line JSON processor, like `jq`, to create JSON strings. For more information on `jq`, see the [jq repository](#) on *GitHub*.

```
$ aws dynamodb get-item --table-name my-table --key '{"id": {"N": "1"}}'  
  
{  
  "Item": {  
    "name": {  
      "S": "John"  
    },  
    "id": {  
      "N": "1"  
    }  
  }  
}
```

Document

Note

[Shorthand syntax](#) is not compatible with document types.

Document types are used to send data without needing to embed JSON inside strings. The document type enables services to provide arbitrary schemas for you to use more flexible data types.

This allows for sending JSON data without needing to escape values. For example, instead of using the following escaped JSON input:

```
{"document": "{\"key\":true}"}
```

You can use the following document type:

```
{"document": {"key": true}}
```

Valid values for document types

Due to the flexible nature of document types, there are multiple valid value types. Valid values include the following:

String

```
--option "value"
```

Number

```
--option 123  
--option 123.456
```

Boolean

```
--option true
```

Null

```
--option null
```

Array

```
--option ["value1", "value2", "value3"]
```

```
--option '["value", 1, true, null, ["key1", 2.34], {"key2": "value2"}]'
```

Object

```
--option '{"key": "value"}'  
--option '{"key1": "value1", "key2": 123, "key3": true, "key4": null, "key5":  
["value3", "value4"], "key6": {"value5": "value6"}}'
```

Quotation marks with strings in the AWS CLI

There are primarily two ways single and double quotes are used in the AWS CLI.

- [Using quotation marks around strings that contain white spaces](#)
- [Using quotation marks inside strings](#)

Using quotation marks around strings that contain white spaces

Parameter names and their values are separated by spaces on the command line. If a string value contains an embedded space, then you must surround the entire string with quotation marks to prevent the AWS CLI from misinterpreting the space as a divider between the value and the next parameter name. Which type of quotation mark you use depends on the operating system you are running the AWS CLI on.

Linux and macOS

Use single quotation marks ' '.

```
$ aws ec2 create-key-pair --key-name 'my key pair'
```

For more information on using quotes, see the user documentation for your preferred shell.

PowerShell

Single quotations (recommended)

Single quotation marks ' ' are called verbatim strings. The string is passed to the command exactly as you type it, which means PowerShell variables will not pass through.

```
PS C:\> aws ec2 create-key-pair --key-name 'my key pair'
```

Double quotations

Double quotation marks " " are called expandable string. Variables can be passed in expandable strings.

```
PS C:\> aws ec2 create-key-pair --key-name "my key pair"
```

For more information on using quotes, see [About Quoting Rules](#) in the *Microsoft PowerShell Docs*.

Windows command prompt

Use double quotation marks " " .

```
C:\> aws ec2 create-key-pair --key-name "my key pair"
```

Optionally, you can separate the parameter name from the value with an equals sign = instead of a space. This is typically necessary only if the value of the parameter starts with a hyphen.

```
$ aws ec2 delete-key-pair --key-name=-mykey
```

Using quotation marks inside strings

Strings might contain quotation marks, and your shell might require escaping quotations for them to work properly. One of the common parameter value types is a JSON string. This is complex since it includes spaces and double quotation marks " " around each element name and value in the JSON structure. The way you enter JSON-formatted parameters on the command line differs depending on your operating system.

For more advanced JSON usage in the command line, consider using a command line JSON processor, like `jq`, to create JSON strings. For more information on `jq`, see the [jq repository](#) on *GitHub*.

Linux and macOS

For Linux and macOS to interpret strings literally use single quotation marks ' ' to enclose the JSON data structure, as in the following example. You do not need to escape double quotation marks embedded in the JSON string, as they are being treated literally. Since the JSON is

enclosed in single quotation marks, any single quotation marks in the string will need to be escaped, this is usually accomplished using a backslash before the single quote `\'`.

```
$ aws ec2 run-instances \  
  --image-id ami-12345678 \  
  --block-device-mappings '[{"DeviceName":"/dev/sdb","Ebs":  
{"VolumeSize":20,"DeleteOnTermination":false,"VolumeType":"standard"}}]'
```

For more information on using quotes, see the user documentation for your preferred shell.

PowerShell

Use single quotation marks `' '` or double quotation marks `" "`.

Single quotations (recommended)

Single quotation marks `' '` are called `verbatim` strings. The string is passed to the command exactly as you type it, which means PowerShell variables will not pass through.

Since JSON data structures include double quotes, we suggest **single** quotation marks `' '` to enclose it. If you use **single** quotation marks, you do not need to escape **double** quotation marks embedded in the JSON string. However, you need to escape each **single** quotation mark with a backtick ``` within the JSON structure.

```
PS C:\> aws ec2 run-instances `  
  --image-id ami-12345678 `  
  --block-device-mappings '[{"DeviceName":"/dev/sdb","Ebs":  
{"VolumeSize":20,"DeleteOnTermination":false,"VolumeType":"standard"}}]'
```

Double quotations

Double quotation marks `" "` are called `expandable` strings. Variables can be passed in expandable strings.

If you use **double** quotation marks, you do not need to escape **single** quotation marks embedded in the JSON string. However, you need to escape each **double** quotation mark with a backtick ``` within the JSON structure, as with the following example.

```
PS C:\> aws ec2 run-instances `  
  --image-id ami-12345678 `
```

```
--block-device-mappings "[{"DeviceName `": `"/dev/sdb `", "Ebs `":
{"VolumeSize `":20, "DeleteOnTermination `":false, "VolumeType `": "standard `"}]"
```

For more information on using quotes, see [About Quoting Rules](#) in the *Microsoft PowerShell Docs*.

Warning

Before PowerShell sends a command to the AWS CLI, it determines if your command is interpreted using typical PowerShell or CommandLineToArgvW quoting rules. When PowerShell processes using CommandLineToArgvW, you must escape characters with a backslash \.

For more information on CommandLineToArgvW in PowerShell, see [What's up with the strange treatment of quotation marks and backslashes by CommandLineToArgvW](#) in the *Microsoft DevBlogs*, [Everyone quotes command line arguments the wrong way](#) in the *Microsoft Docs Blog*, and [CommandLineToArgvW function](#) in the *Microsoft Docs*.

Single quotations

Single quotation marks ' ' are called verbatim strings. The string is passed to the command exactly as you type it, which means PowerShell variables will not pass through. Escape characters with a backslash \.

```
PS C:\> aws ec2 run-instances `
  --image-id ami-12345678 `
  --block-device-mappings '[{"DeviceName\`":\`"/dev/sdb\`",\`"Ebs\`":
{"VolumeSize\`":20,\`"DeleteOnTermination\`":false,\`"VolumeType\`":\`"standard\`"}]'
```

Double quotations

Double quotation marks " " are called expandable strings. Variables can be passed in expandable strings. For double quoted strings you have to escape twice using ` \ for each quote instead of only using a backtick. The backtick escapes the backslash, and then the backslash is used as an escape character for the CommandLineToArgvW process.

```
PS C:\> aws ec2 run-instances `
  --image-id ami-12345678 `
  --block-device-mappings "[{"DeviceName `": `"/dev/sdb `", `Ebs `":
{"VolumeSize `":20, `DeleteOnTermination `":false, `VolumeType `": `
"standard `"}]"
```


Blobs (recommended)

To bypass PowerShell quoting rules for JSON data input, use Blobs to pass your JSON data directly to the AWS CLI. For more information on Blobs, see [Blob](#).

Windows command prompt

The Windows command prompt requires double quotation marks " " to enclose the JSON data structure. Also, to prevent the command processor from misinterpreting the double quotation marks embedded in the JSON, you must also escape (precede with a backslash \ character) each double quotation mark " within the JSON data structure itself, as in the following example.

```
C:\> aws ec2 run-instances ^
  --image-id ami-12345678 ^
  --block-device-mappings "[{\\"DeviceName\\":\\"/dev/sdb\\",\\"Ebs\\":
  {\\"VolumeSize\\":20,\\"DeleteOnTermination\\":false,\\"VolumeType\\":\\"standard\\"}"}]"
```

Only the outermost double quotation marks are not escaped.

Load AWS CLI parameters from a file

Some parameters expect file names as arguments, from which the AWS CLI loads the data. Other parameters enable you to specify the parameter value as either text typed on the command line or read from a file. Whether a file is required or optional, you must encode the file correctly so that the AWS CLI can understand it. The file's encoding must match the reading system's default locale. You can determine this by using the Python `locale.getpreferredencoding()` method.

Note

By default, Windows PowerShell outputs text as UTF-16, which conflicts with the UTF-8 encoding used by JSON files and many Linux systems. We recommend that you use `Encoding ascii` with your PowerShell `Out-File` commands to ensure the AWS CLI can read the resulting file.

Topics

- [How to load parameters from a file](#)

- [Binary files](#)

How to load parameters from a file

Sometimes it's convenient to load a parameter value from a file instead of trying to type it all as a command line parameter value, such as when the parameter is a complex JSON string. To specify a file that contains the value, specify a file URL in the following format.

```
file://complete/path/to/file
```

- The first two slash '/' characters are part of the specification. If the required path begins with a '/', the result is three slash characters: `file:///folder/file`.
- The URL provides the path to the file that contains the actual parameter content.
- When using files with spaces or special characters, following the [quoting and escaping rules](#) for your terminal.

The file paths in the following examples are interpreted to be relative to the current working directory.

Linux or macOS

```
// Read from a file in the current directory
$ aws ec2 describe-instances --filters file://filter.json

// Read from a file in /tmp
$ aws ec2 describe-instances --filters file:///tmp/filter.json

// Read from a file with a filename with whitespaces
$ aws ec2 describe-instances --filters 'file://filter content.json'
```

Windows command prompt

```
// Read from a file in C:\temp
C:\> aws ec2 describe-instances --filters file://C:\temp\filter.json

// Read from a file with a filename with whitespaces
C:\> aws ec2 describe-instances --filters "file://C:\temp\filter content.json"
```

The `file://` prefix option supports Unix-style expansions, including "`~/`", "`./`", and "`../`". On Windows, the "`~/`" expression expands to your user directory, stored in the `%USERPROFILE%` environment variable. For example, on Windows 10 you would typically have a user directory under `C:\Users\UserName\`.

You must still escape JSON documents that are embedded as the value of another JSON document.

```
$ aws sqs create-queue --queue-name my-queue --attributes file://attributes.json
```

attributes.json

```
{
  "RedrivePolicy": "{\"deadLetterTargetArn\":\"arn:aws:sqs:us-
west-2:0123456789012:deadletter\", \"maxReceiveCount\":\"5\"}"
}
```

Binary files

For commands that take binary data as a parameter, specify that the data is binary content by using the `fileb://` prefix. Commands that accept binary data include:

- **aws ec2 run-instances:** `--user-data` parameter.
- **aws s3api put-object:** `--sse-customer-key` parameter.
- **aws kms decrypt:** `--ciphertext-blob` parameter.

The following example generates a binary 256-bit AES key using a Linux command line tool, and then provides it to Amazon S3 to encrypt an uploaded file server-side.

```
$ dd if=/dev/urandom bs=1 count=32 > sse.key
32+0 records in
32+0 records out
32 bytes (32 B) copied, 0.000164441 s, 195 kB/s
$ aws s3api put-object \
  --bucket my-bucket \
  --key test.txt \
  --body test.txt \
  --sse-customer-key fileb://sse.key \
  --sse-customer-algorithm AES256
```

```
{
  "SSECustomerKeyMD5": "iVg8oWa8sy714+FjtesrJg==",
  "SSECustomerAlgorithm": "AES256",
  "ETag": "\"a6118e84b76cf98bf04bbe14b6045c6c\""
}
```

For another example referencing a file containing JSON-formatted parameters, see [Attach an IAM managed policy to a user](#).

AWS CLI skeletons and input files

Most of the AWS CLI commands accept all parameter inputs from a file. These templates can be generated using the `generate-cli-skeleton` option.

Topics

- [About AWS CLI skeletons and input files](#)
- [Generating a command skeleton](#)

About AWS CLI skeletons and input files

Most of the AWS Command Line Interface (AWS CLI) commands support the ability to accept all parameter inputs from a file using the `--cli-input-json` and `--cli-input-yaml` parameters.

Those same commands helpfully provide the `--generate-cli-skeleton` parameter to generate a file in either JSON or YAML format with all of the parameters that you can edit and fill in.

Then you can run the command with the relevant `--cli-input-json` or `--cli-input-yaml` parameter and point to the filled-in file.

Important

Several AWS CLI commands don't map directly to individual AWS API operations, such as the [aws s3 commands](#). Such commands don't support either the `--generate-cli-skeleton` or `--cli-input-json` and `--cli-input-yaml` parameters described in this topic. If you don't know whether a specific command supports these parameters, run the following command, replacing the *service* and *command* names with the ones you're interested in.

```
$ aws service command help
```

The output includes a `Synopsis` section that shows the parameters that the specified command supports.

```
$ aws iam list-users help
...
SYNOPSIS
    list-users
    ...
    [--cli-input-json | --cli-input-yaml]
    ...
    [--generate-cli-skeleton <value>]
...
```

The `--generate-cli-skeleton` parameter causes the command not to run, but instead to generate and display a parameter template that you can customize and use as input on a later command. The generated template includes all of the parameters that the command supports.

The `--generate-cli-skeleton` parameter accepts one of the following values:

- `input` – The generated template includes all input parameters formatted as JSON. This is the default value.
- `yaml-input` – The generated template includes all input parameters formatted as YAML.
- `output` – The generated template includes all output parameters formatted as JSON. You can't currently request the output parameters as YAML.

Because the AWS CLI is essentially a "wrapper" around the service's API, the skeleton file expects you to reference all parameters by their underlying API parameter names. This is likely different from the AWS CLI parameter name. For example, an AWS CLI parameter named `user-name` might map to the AWS service's API parameter named `UserName` (notice the altered capitalization and missing dash). We recommend that you use the `--generate-cli-skeleton` option to generate the template with the "correct" parameter names to avoid errors. You can also reference the API Reference Guide for the service to see the expected parameter names. You can delete any parameters from the template that are not required and for which you don't want to supply a value.

For example, if you run the following command, it generates the parameter template for the Amazon Elastic Compute Cloud (Amazon EC2) command **run-instances**.

JSON

The following example shows how to generate a template formatted in JSON by using the default value (input) for the `--generate-cli-skeleton` parameter.

```
$ aws ec2 run-instances --generate-cli-skeleton
```

```
{
  "DryRun": true,
  "ImageId": "",
  "MinCount": 0,
  "MaxCount": 0,
  "KeyName": "",
  "SecurityGroups": [
    ""
  ],
  "SecurityGroupIds": [
    ""
  ],
  "UserData": "",
  "InstanceType": "",
  "Placement": {
    "AvailabilityZone": "",
    "GroupName": "",
    "Tenancy": ""
  },
  "KernelId": "",
  "RamdiskId": "",
  "BlockDeviceMappings": [
    {
      "VirtualName": "",
      "DeviceName": "",
      "Ebs": {
        "SnapshotId": "",
        "VolumeSize": 0,
        "DeleteOnTermination": true,
        "VolumeType": "",
        "Iops": 0,
        "Encrypted": true
      }
    }
  ],
}
```

```
        "NoDevice": ""
      }
    ],
    "Monitoring": {
      "Enabled": true
    },
    "SubnetId": "",
    "DisableApiTermination": true,
    "InstanceInitiatedShutdownBehavior": "",
    "PrivateIpAddress": "",
    "ClientToken": "",
    "AdditionalInfo": "",
    "NetworkInterfaces": [
      {
        "NetworkInterfaceId": "",
        "DeviceIndex": 0,
        "SubnetId": "",
        "Description": "",
        "PrivateIpAddress": "",
        "Groups": [
          ""
        ],
        "DeleteOnTermination": true,
        "PrivateIpAddresses": [
          {
            "PrivateIpAddress": "",
            "Primary": true
          }
        ],
        "SecondaryPrivateIpAddressCount": 0,
        "AssociatePublicIpAddress": true
      }
    ],
    "IamInstanceProfile": {
      "Arn": "",
      "Name": ""
    },
    "EbsOptimized": true
  }
}
```

YAML

The following example shows how to generate a template formatted in YAML by using the value `yaml-input` for the `--generate-cli-skeleton` parameter.

```
$ aws ec2 run-instances --generate-cli-skeleton yaml-input
```

```
BlockDeviceMappings: # The block device mapping entries.
- DeviceName: '' # The device name (for example, /dev/sdh or xvdh).
  VirtualName: '' # The virtual device name (ephemeralN).
  Ebs: # Parameters used to automatically set up Amazon EBS volumes when the
instance is launched.
    DeleteOnTermination: true # Indicates whether the EBS volume is deleted on
instance termination.
    Iops: 0 # The number of I/O operations per second (IOPS) that the volume
supports.
    SnapshotId: '' # The ID of the snapshot.
    VolumeSize: 0 # The size of the volume, in GiB.
    VolumeType: st1 # The volume type. Valid values are: standard, io1, gp2, sc1,
st1.
    Encrypted: true # Indicates whether the encryption state of an EBS volume is
changed while being restored from a backing snapshot.
    KmsKeyId: '' # Identifier (key ID, key alias, ID ARN, or alias ARN) for a
customer managed KMS key under which the EBS volume is encrypted.
    NoDevice: '' # Suppresses the specified device included in the block device
mapping of the AMI.
ImageId: '' # The ID of the AMI.
InstanceType: c4.4xlarge # The instance type. Valid values are: t1.micro, t2.nano,
t2.micro, t2.small, t2.medium, t2.large, t2.xlarge, t2.2xlarge, t3.nano, t3.micro,
t3.small, t3.medium, t3.large, t3.xlarge, t3.2xlarge, t3a.nano, t3a.micro,
t3a.small, t3a.medium, t3a.large, t3a.xlarge, t3a.2xlarge, m1.small, m1.medium,
m1.large, m1.xlarge, m3.medium, m3.large, m3.xlarge, m3.2xlarge, m4.large,
m4.xlarge, m4.2xlarge, m4.4xlarge, m4.10xlarge, m4.16xlarge, m2.xlarge, m2.2xlarge,
m2.4xlarge, cr1.8xlarge, r3.large, r3.xlarge, r3.2xlarge, r3.4xlarge, r3.8xlarge,
r4.large, r4.xlarge, r4.2xlarge, r4.4xlarge, r4.8xlarge, r4.16xlarge, r5.large,
r5.xlarge, r5.2xlarge, r5.4xlarge, r5.8xlarge, r5.12xlarge, r5.16xlarge,
r5.24xlarge, r5.metal, r5a.large, r5a.xlarge, r5a.2xlarge, r5a.4xlarge,
r5a.8xlarge, r5a.12xlarge, r5a.16xlarge, r5a.24xlarge, r5d.large, r5d.xlarge,
r5d.2xlarge, r5d.4xlarge, r5d.8xlarge, r5d.12xlarge, r5d.16xlarge, r5d.24xlarge,
r5d.metal, r5ad.large, r5ad.xlarge, r5ad.2xlarge, r5ad.4xlarge, r5ad.8xlarge,
r5ad.12xlarge, r5ad.16xlarge, r5ad.24xlarge, x1.16xlarge, x1.32xlarge, x1e.xlarge,
x1e.2xlarge, x1e.4xlarge, x1e.8xlarge, x1e.16xlarge, x1e.32xlarge, i2.xlarge,
i2.2xlarge, i2.4xlarge, i2.8xlarge, i3.large, i3.xlarge, i3.2xlarge, i3.4xlarge,
i3.8xlarge, i3.16xlarge, i3.metal, i3en.large, i3en.xlarge, i3en.2xlarge,
i3en.3xlarge, i3en.6xlarge, i3en.12xlarge, i3en.24xlarge, i3en.metal, hi1.4xlarge,
hs1.8xlarge, c1.medium, c1.xlarge, c3.large, c3.xlarge, c3.2xlarge, c3.4xlarge,
c3.8xlarge, c4.large, c4.xlarge, c4.2xlarge, c4.4xlarge, c4.8xlarge, c5.large,
c5.xlarge, c5.2xlarge, c5.4xlarge, c5.9xlarge, c5.12xlarge, c5.18xlarge,
```


c5.24xlarge, c5.metal, c5d.large, c5d.xlarge, c5d.2xlarge, c5d.4xlarge, c5d.9xlarge, c5d.18xlarge, c5n.large, c5n.xlarge, c5n.2xlarge, c5n.4xlarge, c5n.9xlarge, c5n.18xlarge, cc1.4xlarge, cc2.8xlarge, g2.2xlarge, g2.8xlarge, g3.4xlarge, g3.8xlarge, g3.16xlarge, g3s.xlarge, g4dn.xlarge, g4dn.2xlarge, g4dn.4xlarge, g4dn.8xlarge, g4dn.12xlarge, g4dn.16xlarge, cg1.4xlarge, p2.xlarge, p2.8xlarge, p2.16xlarge, p3.2xlarge, p3.8xlarge, p3.16xlarge, p3dn.24xlarge, d2.xlarge, d2.2xlarge, d2.4xlarge, d2.8xlarge, f1.2xlarge, f1.4xlarge, f1.16xlarge, m5.large, m5.xlarge, m5.2xlarge, m5.4xlarge, m5.8xlarge, m5.12xlarge, m5.16xlarge, m5.24xlarge, m5.metal, m5a.large, m5a.xlarge, m5a.2xlarge, m5a.4xlarge, m5a.8xlarge, m5a.12xlarge, m5a.16xlarge, m5a.24xlarge, m5d.large, m5d.xlarge, m5d.2xlarge, m5d.4xlarge, m5d.8xlarge, m5d.12xlarge, m5d.16xlarge, m5d.24xlarge, m5d.metal, m5ad.large, m5ad.xlarge, m5ad.2xlarge, m5ad.4xlarge, m5ad.8xlarge, m5ad.12xlarge, m5ad.16xlarge, m5ad.24xlarge, h1.2xlarge, h1.4xlarge, h1.8xlarge, h1.16xlarge, z1d.large, z1d.xlarge, z1d.2xlarge, z1d.3xlarge, z1d.6xlarge, z1d.12xlarge, z1d.metal, u-6tb1.metal, u-9tb1.metal, u-12tb1.metal, u-18tb1.metal, u-24tb1.metal, a1.medium, a1.large, a1.xlarge, a1.2xlarge, a1.4xlarge, a1.metal, m5dn.large, m5dn.xlarge, m5dn.2xlarge, m5dn.4xlarge, m5dn.8xlarge, m5dn.12xlarge, m5dn.16xlarge, m5dn.24xlarge, m5n.large, m5n.xlarge, m5n.2xlarge, m5n.4xlarge, m5n.8xlarge, m5n.12xlarge, m5n.16xlarge, m5n.24xlarge, r5dn.large, r5dn.xlarge, r5dn.2xlarge, r5dn.4xlarge, r5dn.8xlarge, r5dn.12xlarge, r5dn.16xlarge, r5dn.24xlarge, r5n.large, r5n.xlarge, r5n.2xlarge, r5n.4xlarge, r5n.8xlarge, r5n.12xlarge, r5n.16xlarge, r5n.24xlarge.

Ipv6AddressCount: 0 # [EC2-VPC] The number of IPv6 addresses to associate with the primary network interface.

Ipv6Addresses: # [EC2-VPC] The IPv6 addresses from the range of the subnet to associate with the primary network interface.

- Ipv6Address: '' # The IPv6 address.

KernelId: '' # The ID of the kernel.

KeyName: '' # The name of the key pair.

MaxCount: 0 # [REQUIRED] The maximum number of instances to launch.

MinCount: 0 # [REQUIRED] The minimum number of instances to launch.

Monitoring: # Specifies whether detailed monitoring is enabled for the instance.

Enabled: true # [REQUIRED] Indicates whether detailed monitoring is enabled.

Placement: # The placement for the instance.

AvailabilityZone: '' # The Availability Zone of the instance.

Affinity: '' # The affinity setting for the instance on the Dedicated Host.

GroupName: '' # The name of the placement group the instance is in.

PartitionNumber: 0 # The number of the partition the instance is in.

HostId: '' # The ID of the Dedicated Host on which the instance resides.

Tenancy: dedicated # The tenancy of the instance (if the instance is running in a VPC). Valid values are: default, dedicated, host.

SpreadDomain: '' # Reserved for future use.

RamdiskId: '' # The ID of the RAM disk to select.

SecurityGroupIds: # The IDs of the security groups.

```
- ''
SecurityGroups: # [default VPC] The names of the security groups.
- ''
SubnetId: '' # [EC2-VPC] The ID of the subnet to launch the instance into.
UserData: '' # The user data to make available to the instance.
AdditionalInfo: '' # Reserved.
ClientToken: '' # Unique, case-sensitive identifier you provide to ensure the
  idempotency of the request.
DisableApiTermination: true # If you set this parameter to true, you can't terminate
  the instance using the Amazon EC2 console, CLI, or API; otherwise, you can.
DryRun: true # Checks whether you have the required permissions for the action,
  without actually making the request, and provides an error response.
EbsOptimized: true # Indicates whether the instance is optimized for Amazon EBS I/O.
IamInstanceProfile: # The IAM instance profile.
  Arn: '' # The Amazon Resource Name (ARN) of the instance profile.
  Name: '' # The name of the instance profile.
InstanceInitiatedShutdownBehavior: stop # Indicates whether an instance stops or
  terminates when you initiate shutdown from the instance (using the operating system
  command for system shutdown). Valid values are: stop, terminate.
NetworkInterfaces: # The network interfaces to associate with the instance.
- AssociatePublicIpAddress: true # Indicates whether to assign a public IPv4
  address to an instance you launch in a VPC.
  DeleteOnTermination: true # If set to true, the interface is deleted when the
  instance is terminated.
  Description: '' # The description of the network interface.
  DeviceIndex: 0 # The position of the network interface in the attachment order.
  Groups: # The IDs of the security groups for the network interface.
- ''
  Ipv6AddressCount: 0 # A number of IPv6 addresses to assign to the network
  interface.
  Ipv6Addresses: # One or more IPv6 addresses to assign to the network interface.
  - Ipv6Address: '' # The IPv6 address.
  NetworkInterfaceId: '' # The ID of the network interface.
  PrivateIpAddress: '' # The private IPv4 address of the network interface.
  PrivateIpAddresses: # One or more private IPv4 addresses to assign to the network
  interface.
  - Primary: true # Indicates whether the private IPv4 address is the primary
  private IPv4 address.
    PrivateIpAddress: '' # The private IPv4 addresses.
  SecondaryPrivateIpAddressCount: 0 # The number of secondary private IPv4
  addresses.
  SubnetId: '' # The ID of the subnet associated with the network interface.
  InterfaceType: '' # The type of network interface.
PrivateIpAddress: '' # [EC2-VPC] The primary IPv4 address.
```

```
ElasticGpuSpecification: # An elastic GPU to associate with the instance.
- Type: '' # [REQUIRED] The type of Elastic Graphics accelerator.
ElasticInferenceAccelerators: # An elastic inference accelerator to associate with
the instance.
- Type: '' # [REQUIRED] The type of elastic inference accelerator.
TagSpecifications: # The tags to apply to the resources during launch.
- ResourceType: network-interface # The type of resource to tag. Valid values
are: client-vpn-endpoint, customer-gateway, dedicated-host, dhcp-options, elastic-
ip, fleet, fpga-image, host-reservation, image, instance, internet-gateway,
launch-template, natgateway, network-acl, network-interface, reserved-instances,
route-table, security-group, snapshot, spot-instances-request, subnet, traffic-
mirror-filter, traffic-mirror-session, traffic-mirror-target, transit-gateway,
transit-gateway-attachment, transit-gateway-route-table, volume, vpc, vpc-peering-
connection, vpn-connection, vpn-gateway.
  Tags: # The tags to apply to the resource.
  - Key: '' # The key of the tag.
  Value: '' # The value of the tag.
LaunchTemplate: # The launch template to use to launch the instances.
  LaunchTemplateId: '' # The ID of the launch template.
  LaunchTemplateName: '' # The name of the launch template.
  Version: '' # The version number of the launch template.
InstanceMarketOptions: # The market (purchasing) option for the instances.
  MarketType: spot # The market type. Valid values are: spot.
  SpotOptions: # The options for Spot Instances.
  MaxPrice: '' # The maximum hourly price you're willing to pay for the Spot
Instances.
  SpotInstanceType: one-time # The Spot Instance request type. Valid values are:
one-time, persistent.
  BlockDurationMinutes: 0 # The required duration for the Spot Instances (also
known as Spot blocks), in minutes.
  ValidUntil: 1970-01-01 00:00:00 # The end date of the request.
  InstanceInterruptionBehavior: terminate # The behavior when a Spot Instance is
interrupted. Valid values are: hibernate, stop, terminate.
CreditSpecification: # The credit option for CPU usage of the T2 or T3 instance.
  CpuCredits: '' # [REQUIRED] The credit option for CPU usage of a T2 or T3
instance.
CpuOptions: # The CPU options for the instance.
  CoreCount: 0 # The number of CPU cores for the instance.
  ThreadsPerCore: 0 # The number of threads per CPU core.
CapacityReservationSpecification: # Information about the Capacity Reservation
targeting option.
  CapacityReservationPreference: none # Indicates the instance's Capacity
Reservation preferences. Valid values are: open, none.
  CapacityReservationTarget: # Information about the target Capacity Reservation.
```

```

CapacityReservationId: '' # The ID of the Capacity Reservation.
HibernationOptions: # Indicates whether an instance is enabled for hibernation.
  Configured: true # If you set this parameter to true, your instance is enabled
  for hibernation.
LicenseSpecifications: # The license configurations.
- LicenseConfigurationArn: '' # The Amazon Resource Name (ARN) of the license
  configuration.

```

Generating a command skeleton

To generate and use a parameter skeleton file

1. Run the command with the `--generate-cli-skeleton` parameter to produce either JSON or YAML and direct the output to a file to save it.

JSON

```
$ aws ec2 run-instances --generate-cli-skeleton input > ec2runinst.json
```

YAML

```
$ aws ec2 run-instances --generate-cli-skeleton yaml-input > ec2runinst.yaml
```

2. Open the parameter skeleton file in your text editor and remove any of the parameters that you don't need. For example, you might strip the template down to the following. Be sure that the file is still valid JSON or YAML after you remove the elements you don't need.

JSON

```
{
  "DryRun": true,
  "ImageId": "",
  "KeyName": "",
  "SecurityGroups": [
    ""
  ],
  "InstanceType": "",
  "Monitoring": {
    "Enabled": true
  }
}
```

```
}
```

YAML

```
DryRun: true
ImageId: ''
KeyName: ''
SecurityGroups:
- ''
InstanceType:
Monitoring:
  Enabled: true
```

In this example, we leave the `DryRun` parameter set to `true` to use the Amazon EC2 dry run feature. This feature lets you safely test the command without actually creating or modifying any resources.

3. Fill in the remaining values with values appropriate for your scenario. In this example, we provide the instance type, key name, security group, and identifier of the Amazon Machine Image (AMI) to use. This example assumes the default AWS Region. The AMI `ami-dfc39aef` is a 64-bit Amazon Linux image hosted in the `us-west-2` Region. If you use a different Region, you must [find the correct AMI ID to use](#).

JSON

```
{
  "DryRun": true,
  "ImageId": "ami-dfc39aef",
  "KeyName": "mykey",
  "SecurityGroups": [
    "my-sg"
  ],
  "InstanceType": "t2.micro",
  "Monitoring": {
    "Enabled": true
  }
}
```

YAML

```
DryRun: true
ImageId: 'ami-dfc39aef'
KeyName: 'mykey'
SecurityGroups:
- 'my-sg'
InstanceType: 't2.micro'
Monitoring:
  Enabled: true
```

4. Run the command with the completed parameters by passing the completed template file to either the `--cli-input-json` or `--cli-input-yaml` parameter by using the `file://` prefix. The AWS CLI interprets the path to be relative to your current working directory, so in the following example that displays only the file name with no path, it looks for the file directly in the current working directory.

JSON

```
$ aws ec2 run-instances --cli-input-json file://ec2runinst.json
```

```
A client error (DryRunOperation) occurred when calling the RunInstances
operation: Request would have succeeded, but DryRun flag is set.
```

YAML

```
$ aws ec2 run-instances --cli-input-yaml file://ec2runinst.yaml
```

```
A client error (DryRunOperation) occurred when calling the RunInstances
operation: Request would have succeeded, but DryRun flag is set.
```

The dry run error indicates that the JSON or YAML is formed correctly and that the parameter values are valid. If other issues are reported in the output, fix them and repeat the previous step until the "Request would have succeeded" message is displayed.

5. Now you can set the `DryRun` parameter to `false` to disable dry run.

JSON

```
{
  "DryRun": false,
  "ImageId": "ami-dfc39aef",
  "KeyName": "mykey",
  "SecurityGroups": [
    "my-sg"
  ],
  "InstanceType": "t2.micro",
  "Monitoring": {
    "Enabled": true
  }
}
```

YAML

```
DryRun: false
ImageId: 'ami-dfc39aef'
KeyName: 'mykey'
SecurityGroups:
- 'my-sg'
InstanceType: 't2.micro'
Monitoring:
  Enabled: true
```

6. Run the command, and `run-instances` actually launches an Amazon EC2 instance and displays the details generated by the successful launch. The format of the output is controlled by the `--output` parameter, separately from the format of your input parameter template.

JSON

```
$ aws ec2 run-instances --cli-input-json file://ec2runinst.json --output json
```

```
{
  "OwnerId": "123456789012",
  "ReservationId": "r-d94a2b1",
  "Groups": [],
  "Instances": [
    ...
  ]
}
```

YAML

```
$ aws ec2 run-instances --cli-input-yaml file://ec2runinst.yaml --output yaml
```

```
OwnerId: '123456789012'  
ReservationId: 'r-d94a2b1',  
Groups":  
- ''  
Instances:  
...
```

Use shorthand syntax with the AWS CLI

The AWS Command Line Interface (AWS CLI) can accept many of its option parameters in JSON format. However, it can be tedious to enter large JSON lists or structures on the command line. To make this easier, the AWS CLI also supports a shorthand syntax that enables a simpler representation of your option parameters than using the full JSON format.

Topics

- [Structure parameters](#)
- [Using shorthand syntax with the AWS Command Line Interface](#)

Structure parameters

The shorthand syntax in the AWS CLI makes it easier for users to input parameters that are flat (non-nested structures). The format is a comma-separated list of key-value pairs. Be sure to use the [quoting](#) and escaping rules appropriate for your terminal as shorthand syntax are strings.

Linux or macOS

```
--option key1=value1,key2=value2,key3=value3
```

PowerShell

```
--option "key1=value1,key2=value2,key3=value3"
```


These are both equivalent to the following example, formatted in JSON.

```
--option '{"key1":"value1","key2":"value2","key3":"value3"}
```

There must be no white space between each comma-separated key-value pair. Here is an example of the Amazon DynamoDB `update-table` command with the `--provisioned-throughput` option specified in shorthand.

```
$ aws dynamodb update-table \  
  --provisioned-throughput ReadCapacityUnits=15,WriteCapacityUnits=10 \  
  --table-name MyDDBTable
```

This is equivalent to the following example formatted in JSON.

```
$ aws dynamodb update-table \  
  --provisioned-throughput '{"ReadCapacityUnits":15,"WriteCapacityUnits":10}' \  
  --table-name MyDDBTable
```

Using shorthand syntax with the AWS Command Line Interface

You can specify input parameters in a list form in two ways: JSON or shorthand. The AWS CLI shorthand syntax is designed to make it easier to pass in lists with number, string, or non-nested structures.

The basic format is shown here, where values in the list are separated by a single space.

```
--option value1 value2 value3
```

This is equivalent to the following example, formatted in JSON.

```
--option '[value1,value2,value3]'
```

As previously mentioned, you can specify a list of numbers, a list of strings, or a list of non-nested structures in shorthand. The following is an example of the `stop-instances` command for Amazon Elastic Compute Cloud (Amazon EC2), where the input parameter (list of strings) for the `--instance-ids` option is specified in shorthand.

```
$ aws ec2 stop-instances \  
  --instance-ids ["i-12345678","i-87654321"]
```

```
--instance-ids i-1486157a i-1286157c i-ec3a7e87
```

This is equivalent to the following example formatted in JSON.

```
$ aws ec2 stop-instances \  
  --instance-ids '["i-1486157a","i-1286157c","i-ec3a7e87"]'
```

The following example shows the Amazon EC2 `create-tags` command, which takes a list of non-nested structures for the `--tags` option. The `--resources` option specifies the ID of the instance to tag.

```
$ aws ec2 create-tags \  
  --resources i-1286157c \  
  --tags Key=My1stTag,Value=Value1 Key=My2ndTag,Value=Value2  
Key=My3rdTag,Value=Value3
```

This is equivalent to the following example, formatted in JSON. The JSON parameter is written over multiple lines for readability.

```
$ aws ec2 create-tags \  
  --resources i-1286157c \  
  --tags '[  
    {"Key": "My1stTag", "Value": "Value1"},  
    {"Key": "My2ndTag", "Value": "Value2"},  
    {"Key": "My3rdTag", "Value": "Value3"}  
]'
```

Have the AWS CLI prompt you for commands

You can have the AWS CLI version 2 prompt you commands, parameters, and resources when you run an `aws` command.

Topics

- [How it works](#)
- [Auto-prompt features](#)
- [Auto-prompt modes](#)
- [Configure auto-prompt](#)

How it works

If enabled, the auto-prompt enables you to use the **ENTER** key to complete a partially entered command. After pressing the **ENTER** key, commands, parameters, and resources are suggested based on what you continue to type. The suggestions list the name of the command, parameter, or resource on the left and a description of it on the right. To select and use a suggestion, use the arrows keys to highlight a row, and then press the **SPACE** key. When you've finished entering in your command, press **ENTER** to use the command. The following example demonstrates what a suggested list from auto-prompt looks like.

```
$ aws
> aws a
    accessanalyzer      Access Analyzer
    acm                  AWS Certificate Manager
    acm-pca              AWS Certificate Manager Private Certificate
Authority
    alexaforbusiness    Alexa For Business
    amplify              AWS Amplify
```

Auto-prompt features

The auto-prompt contains the following useful features:

Documentation panel

Provides the help documentation for the current command. To open the documentation, press the **F3** key.

Command completion

Suggests aws commands to use. To see a list, partially enter the command. The following example is searching for a service starting with the letter a.

```
$ aws
> aws a
    accessanalyzer      Access Analyzer
    acm                  AWS Certificate Manager
    acm-pca              AWS Certificate Manager Private Certificate
Authority
    alexaforbusiness    Alexa For Business
    amplify              AWS Amplify
```

Parameter completion

After a command is typed, auto-prompt starts to suggest parameters. The descriptions for the parameters include the value type, and a description of what the parameter is. Required parameters are listed first, and are labeled as required. The following example shows the auto-prompt list of parameters for `aws dynamodb describe-table`.

```
$ aws dynamodb describe-table
> aws dynamodb describe-table
      --table-name (required) [string] The name of the
table to describe.
      --cli-input-json       [string] Reads arguments
from the JSON string provided. The JSON string follows the format provide...
      --cli-input-yaml       [string] Reads arguments
from the YAML string provided. The YAML string follows the format provide...
      --generate-cli-skeleton [string] Prints a JSON
skeleton to standard output without sending an API request. If provided wit...
```

Resource completion

The auto-prompt makes AWS API calls using available AWS resource properties to suggest resource values. This allows for auto-prompt to suggest possible resources you own when entering in parameters. In the following example auto-prompt lists your table names when filling in the `--table-name` parameter for the `aws dynamodb describe-table` command.

```
$ aws dynamodb describe-table
> aws dynamodb describe-table --table-name
Table1
Table2
Table3
```

Shorthand completion

For parameters that use shorthand syntax, auto-prompt suggests values to use. In the following example, auto-prompt lists shorthand syntax values for the `--placement` parameter in the `aws ec2 run-instances` command.

```
$ aws ec2 run-instances
> aws ec2 run-instances --placement
AvailabilityZone= [string] The Availability Zone of the instance. If not
specified, an Availability Zone wil...
```

```
Affinity=           [string] The affinity setting for the instance on the
Dedicated Host. This parameter is no...
GroupName=         [string] The name of the placement group the instance is in.
PartitionNumber=   [integer] The number of the partition the instance is in.
Valid only if the placement grou...
```

File completion

When filling out parameters in `aws` commands, auto-complete suggests local filenames after using the prefix `file://` or `fileb://`. In the following example, auto-prompt suggests local files after entering in `--item file://` for the `aws ec2 run-instances` command.

```
$ aws ec2 run-instances
> aws ec2 run-instances --item file://
                               item1.txt
                               file1.json
                               file2.json
```

Region completion

When using the global parameter `--region`, auto-prompt lists possible Regions to select from. In the following example, auto-prompt suggests Regions in alphabetical order after entering in `--region` for the `aws dynamodb list-tables` command.

```
$ aws dynamodb list-tables
> aws dynamodb list-tables --region
                               af-south-1
                               ap-east-1
                               ap-northeast-1
                               ap-northeast-2
```

Profile completion

When using the global parameter `--profile`, auto-prompt lists your profiles. In the following example, auto-prompt suggests your profiles after entering in `--profile` for the `aws dynamodb list-tables` command.

```
$ aws dynamodb list-tables
> aws dynamodb list-tables --profile
                               profile1
```

```
profile2
profile3
```

Fuzzy searching

Complete commands and values that contain a specific set of characters. In the following example, auto-prompt suggests Regions that contain eu after entering in `--region eu` for the `aws dynamodb list-tables` command.

```
$ aws dynamodb list-tables
> aws dynamodb list-tables --region west
                                eu-west-1
                                eu-west-2
                                eu-west-3
                                us-west-1
```

History

To view and run previously used commands in auto-prompt mode, press **CTRL + R**. History lists previous commands that you can select by using the arrow keys. In the following example, the auto-prompt mode history is displayed.

```
$ aws
> aws
    dynamodb list-tables
    s3 ls
```

Auto-prompt modes

Auto-prompt for the AWS CLI version 2 has 2 modes that can be configured:

- **Full mode:** Uses auto-prompt each time you attempt to run an aws command, whether you manually call it using the `--cli-auto-prompt` parameter or permanently enabled it. This includes pressing **ENTER** after both a complete command or incomplete command.
- **Partial mode:** Uses auto-prompt if a command is incomplete or cannot be run due to client-side validation errors. This mode is particular useful if you have pre-existing scripts, runbooks, or you only want to be auto-prompted for commands you are unfamiliar with rather than prompted on every command.

Configure auto-prompt

To configure auto-prompt you can use the following methods in order of precedence:

- **Command line options** enable or disable auto-prompt for a single command. Use [--cli-auto-prompt](#) to call auto-prompt and [--no-cli-auto-prompt](#) to disable auto-prompt.
- **Environment variables** use the [aws_cli_auto_prompt](#) variable.
- **Shared config files** use the [cli_auto_prompt](#) setting.

Control command output from the AWS CLI

This section describes the different ways to control the output from the AWS Command Line Interface (AWS CLI). Customizing the AWS CLI output in your terminal can improve readability, streamline scripting automation and provide easier navigation through larger data sets.

The AWS CLI supports multiple [output formats](#), including [json](#), [text](#), [yaml](#), and [table](#). Some services have server-side [pagination](#) for their data and the AWS CLI provides it's own client-side features for additional pagination options.

Lastly, the AWS CLI has both [server-side and client-side filtering](#) that you can use individually or together to filter your AWS CLI output.

Topics

- [Sensitive output](#)
- [Server-side vs client-side output options](#)
- [Set the AWS CLI output format](#)
- [Use AWS CLI pagination options](#)
- [Filter AWS CLI output](#)

Sensitive output

Some operations of the AWS CLI might return information that could be considered sensitive, including information from environment variables. The exposure of this information might represent a security risk in certain scenarios; for example, the information could be included in continuous integration and continuous deployment (CI/CD) logs. It is therefore important that you

review when you are including such output as part of your logs, and suppress the output when not needed.

For additional information about protecting sensitive data, see [the section called “Data Protection”](#).

Consider the following best practices:

- Consider programmatically retrieving your secrets from a secrets store, such as AWS Secrets Manager.
- Review the contents of your build logs to ensure they do not contain sensitive information. Consider approaches such as piping to `/dev/null` or capturing the output as a bash or PowerShell variable to suppress command outputs.

The following is a bash example for redirecting output, but not errors, to `/dev/null`:

```
$ aws s3 ls > /dev/null
```

For specifics on suppressing output for your terminal, see the user documentation of the terminal you use.

- Consider the access of your logs and scope the access appropriately for your use case.

Server-side vs client-side output options

The AWS CLI has both [server-side and client-side filtering](#) that you can use individually or together to filter your AWS CLI output. Server-side filtering is processed first and returns your output for client-side filtering. Server-side filtering is supported by the service API. Client-side filtering is supported by the AWS CLI client using the `--query` parameter.

Server-side output options are features directly supported by the AWS service API. Any data that is filtered or paged out is not sent to the client, which can speed up HTTP response times and improve bandwidth for larger data sets.

Client-side output options are features created by the AWS CLI. All data is sent to the client, then the AWS CLI filters or pages the content displayed. Client-side operations do not save on speed or bandwidth for larger datasets.

When server-side and client-side options are used together, server-side operations are completed first and then sent to the client for client-side operations. This uses the potential speed and

bandwidth savings of server-side options, while using additional AWS CLI features to get your desired output.

Set the AWS CLI output format

This topic describes the different output formats for the AWS Command Line Interface (AWS CLI). The AWS CLI supports the following output formats:

- **[json](#)** – The output is formatted as a [JSON](#) string.
- **[yaml](#)** – The output is formatted as a [YAML](#) string.
- **[yaml-stream](#)** – The output is streamed and formatted as a [YAML](#) string. Streaming allows for faster handling of large data types.
- **[text](#)** – The output is formatted as multiple lines of tab-separated string values. This can be useful to pass the output to a text processor, like `grep`, `sed`, or `awk`.
- **[table](#)** – The output is formatted as a table using the characters `+|-` to form the cell borders. It typically presents the information in a "human-friendly" format that is much easier to read than the others, but not as programmatically useful.

How to select the output format

As explained in the [configuration](#) topic, you can specify the output format in three ways:

- **Using the output option in a named profile in the config file** – The following example sets the default output format to `text`.

```
[default]
output=text
```

- **Using the `AWS_DEFAULT_OUTPUT` environment variable** – The following output sets the format to `table` for the commands in this command line session until the variable is changed or the session ends. Using this environment variable overrides any value set in the `config` file.

```
$ export AWS_DEFAULT_OUTPUT="table"
```

- **Using the `--output` option on the command line** – The following example sets the output of only this one command to `json`. Using this option on the command overrides any currently set environment variable or the value in the `config` file.

```
$ aws swf list-domains --registration-status REGISTERED --output json
```

Important

The output type you specify changes how the `--query` option operates:

- If you specify `--output text`, the output is paginated *before* the `--query` filter is applied, and the AWS CLI runs the query once on *each page* of the output. Due to this, the query includes the first matching element on each page which can result in unexpected extra output. To additionally filter the output, you can use other command line tools such as `head` or `tail`.
- If you specify `--output json`, `--output yaml`, or `--output yaml-stream` the output is completely processed as a single, native structure before the `--query` filter is applied. The AWS CLI runs the query only once against the entire structure, producing a filtered result that is then output.

JSON output format

[JSON](#) is the default output format of the AWS CLI. Most programming languages can easily decode JSON strings using built-in functions or with publicly available libraries. You can combine JSON output with the [--query option](#) in powerful ways to filter and format the AWS CLI JSON-formatted output.

For more advanced filtering that you might not be able to do with `--query`, you can consider `jq`, a command line JSON processor. You can download it and find the official tutorial at <http://stedolan.github.io/jq/>.

The following is an example of JSON output.

```
$ aws iam list-users --output json
```

```
{
  "Users": [
    {
      "Path": "/",
      "UserName": "Admin",
```

```

    "UserId": "AIDA111111111111EXAMPLE",
    "Arn": "arn:aws:iam::123456789012:user/Admin",
    "CreateDate": "2014-10-16T16:03:09+00:00",
    "PasswordLastUsed": "2016-06-03T18:37:29+00:00"
  },
  {
    "Path": "/backup/",
    "UserName": "backup-user",
    "UserId": "AIDA222222222222EXAMPLE",
    "Arn": "arn:aws:iam::123456789012:user/backup/backup-user",
    "CreateDate": "2019-09-17T19:30:40+00:00"
  },
  {
    "Path": "/",
    "UserName": "cli-user",
    "UserId": "AIDA333333333333EXAMPLE",
    "Arn": "arn:aws:iam::123456789012:user/cli-user",
    "CreateDate": "2019-09-17T19:11:39+00:00"
  }
]
}

```

YAML output format

[YAML](#) is a good choice for handling the output programmatically with services and tools that emit or consume [YAML](#)-formatted strings, such as AWS CloudFormation with its support for [YAML-formatted templates](#).

For more advanced filtering that you might not be able to do with `--query`, you can consider `yq`, a command line YAML processor. You can download `yq` in the [yq repository](#) on *GitHub*.

The following is an example of YAML output.

```
$ aws iam list-users --output yaml
```

```

Users:
- Arn: arn:aws:iam::123456789012:user/Admin
  CreateDate: '2014-10-16T16:03:09+00:00'
  PasswordLastUsed: '2016-06-03T18:37:29+00:00'
  Path: /
  UserId: AIDA111111111111EXAMPLE
  UserName: Admin

```

```
- Arn: arn:aws:iam::123456789012:user/backup/backup-user
  CreateDate: '2019-09-17T19:30:40+00:00'
  Path: /backup/
  UserId: AIDA222222222222EXAMPLE
  UserName: arq-45EFD6D1-CE56-459B-B39F-F9C1F78FBE19
- Arn: arn:aws:iam::123456789012:user/cli-user
  CreateDate: '2019-09-17T19:30:40+00:00'
  Path: /
  UserId: AIDA333333333333EXAMPLE
  UserName: cli-user
```

YAML stream output format

The `yaml-stream` format takes advantage of the [YAML](#) format while providing more responsive/faster viewing of large data sets by streaming the data to you. You can start viewing and using YAML data before the entire query downloads.

For more advanced filtering that you might not be able to do with `--query`, you can consider `yq`, a command line YAML processor. You can download `yq` in the [yq repository](#) on *GitHub*.

The following is an example of `yaml-stream` output.

```
$ aws iam list-users --output yaml-stream
```

```
- IsTruncated: false
  Users:
  - Arn: arn:aws:iam::123456789012:user/Admin
    CreateDate: '2014-10-16T16:03:09+00:00'
    PasswordLastUsed: '2016-06-03T18:37:29+00:00'
    Path: /
    UserId: AIDA111111111111EXAMPLE
    UserName: Admin
  - Arn: arn:aws:iam::123456789012:user/backup/backup-user
    CreateDate: '2019-09-17T19:30:40+00:00'
    Path: /backup/
    UserId: AIDA222222222222EXAMPLE
    UserName: arq-45EFD6D1-CE56-459B-B39F-F9C1F78FBE19
  - Arn: arn:aws:iam::123456789012:user/cli-user
    CreateDate: '2019-09-17T19:30:40+00:00'
    Path: /
    UserId: AIDA333333333333EXAMPLE
    UserName: cli-user
```

The following is an example of `yaml-stream` output in conjunction with using the `--page-size` parameter to paginate the streamed YAML content.

```
$ aws iam list-users --output yaml-stream --page-size 2
```

```
- IsTruncated: true
  Marker: ab1234cdef5ghi67jk8lmo9p/
q012rs3t445uv6789w0x1y2z/345a6b78c9d00/1efgh234ij56klmno78pqrstu90vwxyx
  Users:
    - Arn: arn:aws:iam::123456789012:user/Admin
      CreateDate: '2014-10-16T16:03:09+00:00'
      PasswordLastUsed: '2016-06-03T18:37:29+00:00'
      Path: /
      UserId: AIDA111111111111EXAMPLE
      UserName: Admin
    - Arn: arn:aws:iam::123456789012:user/backup/backup-user
      CreateDate: '2019-09-17T19:30:40+00:00'
      Path: /backup/
      UserId: AIDA222222222222EXAMPLE
      UserName: arq-45EFD6D1-CE56-459B-B39F-F9C1F78FBE19
- IsTruncated: false
  Users:
    - Arn: arn:aws:iam::123456789012:user/cli-user
      CreateDate: '2019-09-17T19:30:40+00:00'
      Path: /
      UserId: AIDA333333333333EXAMPLE
      UserName: cli-user
```

Text output format

The text format organizes the AWS CLI output into tab-delimited lines. It works well with traditional Unix text tools such as `grep`, `sed`, and `awk`, and the text processing performed by PowerShell.

The text output format follows the basic structure shown below. The columns are sorted alphabetically by the corresponding key names of the underlying JSON object.

```
IDENTIFIER sorted-column1 sorted-column2
IDENTIFIER2 sorted-column1 sorted-column2
```

The following is an example of text output. Each field is tab separated from the others, with an extra tab where there is an empty field.

```
$ aws iam list-users --output text
```

```
USERS   arn:aws:iam::123456789012:user/Admin          2014-10-16T16:03:09+00:00
2016-06-03T18:37:29+00:00 /                AIDA111111111111EXAMPLE  Admin
USERS   arn:aws:iam::123456789012:user/backup/backup-user 2019-09-17T19:30:40+00:00
/backup/ AIDA222222222222EXAMPLE backup-user
USERS   arn:aws:iam::123456789012:user/cli-user          2019-09-17T19:11:39+00:00
/                AIDA333333333333EXAMPLE cli-user
```

The fourth column is the `PasswordLastUsed` field, and is empty for the last two entries because those users never sign in to the AWS Management Console.

Important

We strongly recommend that if you specify text output, you also always use the `--query` option to ensure consistent behavior.

This is because the text format alphabetically orders output columns by the key name of the underlying JSON object returned by the AWS service, and similar resources might not have the same key names. For example, the JSON representation of a Linux-based Amazon EC2 instance might have elements that are not present in the JSON representation of a Windows-based instance, or vice versa. Also, resources might have key-value elements added or removed in future updates, altering the column ordering. This is where `--query` augments the functionality of the text output to provide you with complete control over the output format.

In the following example, the command specifies which elements to display and *defines the ordering* of the columns with the list notation `[key1, key2, ...]`. This gives you full confidence that the correct key values are always displayed in the expected column. Finally, notice how the AWS CLI outputs `None` as the value for keys that don't exist.

```
$ aws iam list-users --output text --query 'Users[*].
[UserName,Arn,CreateDate,PasswordLastUsed,UserId]'
```

```
Admin          arn:aws:iam::123456789012:user/Admin
2014-10-16T16:03:09+00:00 2016-06-03T18:37:29+00:00 AIDA111111111111EXAMPLE
```

```

backup-user  arn:aws:iam::123456789012:user/backup-user
2019-09-17T19:30:40+00:00  None  AIDA222222222222EXAMPLE
cli-user     arn:aws:iam::123456789012:user/cli-backup
2019-09-17T19:11:39+00:00  None  AIDA333333333333EXAMPLE

```

The following example shows how you can use `grep` and `awk` with the text output from the `aws ec2 describe-instances` command. The first command displays the Availability Zone, current state, and the instance ID of each instance in text output. The second command processes that output to display only the instance IDs of all running instances in the `us-west-2a` Availability Zone.

```
$ aws ec2 describe-instances --query 'Reservations[*].Instances[*].
[Placement.AvailabilityZone, State.Name, InstanceId]' --output text
```

```

us-west-2a    running i-4b41a37c
us-west-2a    stopped i-a071c394
us-west-2b    stopped i-97a217a0
us-west-2a    running i-3045b007
us-west-2a    running i-6fc67758

```

```
$ aws ec2 describe-instances --query 'Reservations[*].Instances[*].
[Placement.AvailabilityZone, State.Name, InstanceId]' --output text | grep us-west-2a |
grep running | awk '{print $3}'
```

```

i-4b41a37c
i-3045b007
i-6fc67758

```

The following example goes a step further and shows not only how to filter the output, but how to use that output to automate changing instance types for each stopped instance.

```
$ aws ec2 describe-instances --query 'Reservations[*].Instances[*].[State.Name,
InstanceId]' --output text |
> grep stopped |
> awk '{print $2}' |
> while read line;
> do aws ec2 modify-instance-attribute --instance-id $line --instance-type '{"Value":
"m1.medium"}';
```

```
> done
```

The text output can also be useful in PowerShell. Because the columns in text output are tab delimited, you can easily split the output into an array by using PowerShell's ``t` delimiter. The following command displays the value of the third column (InstanceId) if the first column (AvailabilityZone) matches the string `us-west-2a`.

```
PS C:\>aws ec2 describe-instances --query 'Reservations[*].Instances[*].
[Placement.AvailabilityZone, State.Name, InstanceId]' --output text |
%{if ($_.split("`t")[0] -match "us-west-2a") { $_.split("`t")[2]; } }
```

```
-4b41a37c
i-a071c394
i-3045b007
i-6fc67758
```

Notice that although the previous example does show how to use the `--query` parameter to parse the underlying JSON objects and pull out the desired column, PowerShell has its own ability to handle JSON, if cross-platform compatibility isn't a concern. Instead of handling the output as text, as most command shells require, PowerShell lets you use the `ConvertFrom-Json` cmdlet to produce a hierarchically structured object. You can then directly access the member you want from that object.

```
(aws ec2 describe-instances --output json | ConvertFrom-
Json).Reservations.Instances.InstanceId
```

Tip

If you output text, and filter the output to a single field using the `--query` parameter, the output is a single line of tab-separated values. To get each value onto a separate line, you can put the output field in brackets, as shown in the following examples.

Tab separated, single-line output:

```
$ aws iam list-groups-for-user --user-name susan --output text --query
"Groups[].GroupName"
```

```
HRDepartment    Developers    SpreadsheetUsers  LocalAdmins
```


You can combine the `--query` option with the `table` format to display a set of elements preselected from the raw output. Notice the output differences between dictionary and list notations: in the first example, column names are ordered alphabetically, and in the second example, unnamed columns are ordered as defined by the user. For more information about the `--query` option, see [Filter AWS CLI output](#).

```
$ aws ec2 describe-volumes --query 'Volumes[*].
{ID:VolumeId,InstanceId:Attachments[0].InstanceId,AZ:AvailabilityZone,Size:Size}' --
output table
```

```
-----
|                               DescribeVolumes                               |
+-----+-----+-----+-----+
|   AZ   |   ID   | InstanceId | Size |
+-----+-----+-----+-----+
| us-west-2a | vol-e11a5288 | i-a071c394 | 30 |
| us-west-2a | vol-2e410a47 | i-4b41a37c | 8  |
+-----+-----+-----+-----+
```

```
$ aws ec2 describe-volumes --query 'Volumes[*].
[VolumeId,Attachments[0].InstanceId,AvailabilityZone,Size]' --output table
```

```
-----
|                               DescribeVolumes                               |
+-----+-----+-----+-----+
| vol-e11a5288 | i-a071c394 | us-west-2a | 30 |
| vol-2e410a47 | i-4b41a37c | us-west-2a | 8  |
+-----+-----+-----+-----+
```

Use AWS CLI pagination options

This topic describes the different ways to paginate output from the AWS CLI.

There are primarily two ways to control pagination from the AWS CLI.

- [Using server-side pagination parameters.](#)
- [Using your default output client-side paging program.](#)

Server-side pagination parameters process first and any output is sent to client-side pagination.

Server-side pagination

For commands that can return a large list of items, the AWS Command Line Interface (AWS CLI) has multiple options to control the number of items included in the output when the AWS CLI calls a service's API to populate the list.

The options include the following:

- [How to use the `--no-paginate` parameter](#)
- [How to use the `--page-size` parameter](#)
- [How to use the `--max-items` parameter](#)
- [How to use the `--starting-token` parameter](#)

By default, the AWS CLI uses a page size determined by the individual service and retrieves all available items. For example, Amazon S3 has a default page size of 1000. If you run `aws s3api list-objects` on an Amazon S3 bucket that contains 3,500 objects, the AWS CLI automatically makes four calls to Amazon S3, handling the service-specific pagination logic for you in the background and returning all 3,500 objects in the final output.

How to use the `--no-paginate` parameter

The `--no-paginate` option disables following pagination tokens on the client side. When using a command, by default the AWS CLI automatically makes multiple calls to return all possible results to create pagination. One call for each page. Disabling pagination has the AWS CLI only call once for the first page of command results.

For example, if you run `aws s3api list-objects` on an Amazon S3 bucket that contains 3,500 objects, the AWS CLI only makes the first call to Amazon S3, returning only the first 1,000 objects in the final output.

```
$ aws s3api list-objects \  
  --bucket my-bucket \  
  --no-paginate  
{  
  "Contents": [  
  ...
```

How to use the `--page-size` parameter

If you see issues when running list commands on a large number of resources, the default page size might be too high. This can cause calls to AWS services to exceed the maximum allowed time and generate a "timed out" error. You can use the `--page-size` option to specify that the AWS CLI request a smaller number of items from each call to the AWS service. The AWS CLI still retrieves the full list, but performs a larger number of service API calls in the background and retrieves a smaller number of items with each call. This gives the individual calls a better chance of succeeding without a timeout. Changing the page size doesn't affect the output; it affects only the number of API calls that need to be made to generate the output.

```
$ aws s3api list-objects \  
  --bucket my-bucket \  
  --page-size 100  
{  
  "Contents": [  
  ...
```

How to use the `--max-items` parameter

To include fewer items at a time in the AWS CLI output, use the `--max-items` option. The AWS CLI still handles pagination with the service as described previously, but prints out only the number of items at a time that you specify.

```
$ aws s3api list-objects \  
  --bucket my-bucket \  
  --max-items 100  
{  
  "NextToken": "eyJNYXJrZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAxfQ==",  
  "Contents": [  
  ...
```

How to use the `--starting-token` parameter

If the number of items output (`--max-items`) is fewer than the total number of items returned by the underlying API calls, the output includes a `NextToken` that you can pass to a subsequent command to retrieve the next set of items. The following example shows how to use the `NextToken` value returned by the previous example, and enables you to retrieve the second 100 items.

Note

The parameter `--starting-token` cannot be null or empty. If the previous command does not return a `NextToken` value, there are no more items to return and you do not need to call the command again.

```
$ aws s3api list-objects \  
  --bucket my-bucket \  
  --max-items 100 \  
  --starting-token eyJNYXJrZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAxfQ==\  
{  
  "Contents": [  
  ...
```

The specified AWS service might not return items in the same order each time you call. If you specify different values for `--page-size` and `--max-items`, you can get unexpected results with missing or duplicated items. To prevent this, use the same number for `--page-size` and `--max-items` to sync the AWS CLI pagination with the pagination of the underlying service. You can also retrieve the whole list and perform any necessary paging operations locally.

Client-side pager

AWS CLI version 2 provides the use of a client-side pager program for output. By default, this feature returns all output through your operating system's default pager program.

In order of precedence, you can specify the output pager in the following ways:

- Using the `cli_pager` setting in the config file in the `default` or named profile.
- Using the `AWS_PAGER` environment variable.
- Using the `PAGER` environment variable.

In order of precedence, you can disable all use of an external paging program in the following ways:

- Use the `--no-cli-pager` command line option to disable the pager for a single command use.
- Set the `cli_pager` setting or `AWS_PAGER` variable to an empty string.

Client-side pager topics:

- [How to use the `cli_pager` setting](#)
- [How to use the `AWS_PAGER` environment variable](#)
- [How to use the `--no-cli-pager` option](#)
- [How to use pager flags](#)

How to use the `cli_pager` setting

You can save your frequently used configuration settings and credentials in files that are maintained by the AWS CLI. Settings in a name profile take precedence over settings in the default profile. For more information on configuration settings, see [Configuration and credential file settings](#).

The following example sets the default output pager to the `less` program.

```
[default]
cli_pager=less
```

The following example sets the default to disable the use of a pager.

```
[default]
cli_pager=
```

How to use the `AWS_PAGER` environment variable

The following example sets the default output pager to the `less` program. For more information on environment variables, see [Environment variables to configure the AWS CLI](#).

Linux and macOS

```
$ export AWS_PAGER="less"
```

Windows

```
C:\> setx AWS_PAGER "less"
```

The following example disables the use of a pager.

Linux and macOS

```
$ export AWS_PAGER=""
```

Windows

```
C:\> setx AWS_PAGER ""
```

How to use the `--no-cli-pager` option

To disable the use of a pager on a single command, use the `--no-cli-pager` option. For more information on command line options, see [Command line options](#).

```
$ aws s3api list-objects \  
  --bucket my-bucket \  
  --no-cli-pager  
{  
  "Contents": [  
  ...
```

How to use pager flags

You can specify flags to use automatically with your paging program. Flags are dependent on the paging program you use. The below examples are for the typical defaults of `less` and `more`.

Linux and macOS

If you do not specify otherwise, the pager AWS CLI version 2 uses by default is `less`. If you don't have the `LESS` environment variable set, the AWS CLI version 2 uses the `FRX` flags. You can combine flags by specifying them when setting the AWS CLI pager.

The following example uses the `S` flag. This flag then combines with the default `FRX` flags to create a final `FRXS` flag.

```
$ export AWS_PAGER="less -S"
```

If you don't want any of the `FRX` flags, you can negate them. The following example negates the `F` flag to create a final `RX` flag.

```
$ export AWS_PAGER="less -+F"
```

For more information on `less` flags see [less](#) on *manpages.org*.

Windows

If you do not specify otherwise, the pager AWS CLI version 2 uses by default is `more` with no additional flags.

The following example uses the `/c` parameter.

```
C:\> setx AWS_PAGER "more /c"
```

For more information on `more` flags see [more](#) on *Microsoft Docs*.

Filter AWS CLI output

The AWS Command Line Interface (AWS CLI) has both server-side and client-side filtering that you can use individually or together to filter your AWS CLI output. Server-side filtering is processed first and returns your output for client-side filtering.

- Server-side filtering is supported by the API, and you usually implement it with a `--filter` parameter. The service only returns matching results which can speed up HTTP response times for large data sets.
- Client-side filtering is supported by the AWS CLI client using the `--query` parameter. This parameter has capabilities the server-side filtering might not have.

Topics

- [Server-side filtering](#)
- [Client-side filtering](#)
- [Combining server-side and client-side filtering](#)
- [Additional resources](#)

Server-side filtering

Server-side filtering in the AWS CLI is provided by the AWS service API. The AWS service only returns the records in the HTTP response that match your filter, which can speed up HTTP response times for large data sets. Since server-side filtering is defined by the service API, the parameter names and functions vary between services. Some common parameter names used for filtering are:

- `--filter` such as [ses](#) and [ce](#).
- `--filters` such as [ec2](#), [autoscaling](#), and [rds](#).
- Names starting with the word `filter`, for example `--filter-expression` for the [aws dynamodb scan](#) command.

For information about whether a specific command has server-side filtering and the filtering rules, see the [AWS CLI version 2 reference guide](#).

Client-side filtering

The AWS CLI provides built-in JSON-based client-side filtering capabilities with the `--query` parameter. The `--query` parameter is a powerful tool you can use to customize the content and style of your output. The `--query` parameter takes the HTTP response that comes back from the server and filters the results before displaying them. Since the entire HTTP response is sent to the client before filtering, client-side filtering can be slower than server-side filtering for large data-sets.

Querying uses [JMESPath syntax](#) to create expressions for filtering your output. To learn JMESPath syntax, see [Tutorial](#) on the *JMESPath website*.

Important

The output type you specify changes how the `--query` option operates:

- If you specify `--output text`, the output is paginated *before* the `--query` filter is applied, and the AWS CLI runs the query once on *each page* of the output. Due to this, the query includes the first matching element on each page which can result in unexpected extra output. To additionally filter the output, you can use other command line tools such as `head` or `tail`.
- If you specify `--output json`, `--output yaml`, or `--output yaml-stream` the output is completely processed as a single, native structure before the `--query` filter is applied. The AWS CLI runs the query only once against the entire structure, producing a filtered result that is then output.

Client-side filtering topics

- [Before you start](#)

- [Identifiers](#)
- [Selecting from a list](#)
- [Filtering nested data](#)
- [Flattening results](#)
- [Filtering for specific values](#)
- [Piping expressions](#)
- [Filtering for multiple identifier values](#)
- [Adding labels to identifier values](#)
- [Functions](#)
- [Advanced --query examples](#)

Before you start

When using filter expressions used in these examples, be sure to use the correct quoting rules for your terminal shell. For more information, see [the section called “Quotes with Strings”](#).

The following JSON output shows an example of what the `--query` parameter can produce. The output describes three Amazon EBS volumes attached to separate Amazon EC2 instances.

Example output

```
$ aws ec2 describe-volumes
{
  "Volumes": [
    {
      "AvailabilityZone": "us-west-2a",
      "Attachments": [
        {
          "AttachTime": "2013-09-17T00:55:03.000Z",
          "InstanceId": "i-a071c394",
          "VolumeId": "vol-e11a5288",
          "State": "attached",
          "DeleteOnTermination": true,
          "Device": "/dev/sda1"
        }
      ],
      "VolumeType": "standard",
      "VolumeId": "vol-e11a5288",
      "State": "in-use",
```

```
"SnapshotId": "snap-f23ec1c8",
"CreateTime": "2013-09-17T00:55:03.000Z",
"Size": 30
},
{
  "AvailabilityZone": "us-west-2a",
  "Attachments": [
    {
      "AttachTime": "2013-09-18T20:26:16.000Z",
      "InstanceId": "i-4b41a37c",
      "VolumeId": "vol-2e410a47",
      "State": "attached",
      "DeleteOnTermination": true,
      "Device": "/dev/sda1"
    }
  ],
  "VolumeType": "standard",
  "VolumeId": "vol-2e410a47",
  "State": "in-use",
  "SnapshotId": "snap-708e8348",
  "CreateTime": "2013-09-18T20:26:15.000Z",
  "Size": 8
},
{
  "AvailabilityZone": "us-west-2a",
  "Attachments": [
    {
      "AttachTime": "2020-11-20T19:54:06.000Z",
      "InstanceId": "i-1jd73kv8",
      "VolumeId": "vol-a1b3c7nd",
      "State": "attached",
      "DeleteOnTermination": true,
      "Device": "/dev/sda1"
    }
  ],
  "VolumeType": "standard",
  "VolumeId": "vol-a1b3c7nd",
  "State": "in-use",
  "SnapshotId": "snap-234087fb",
  "CreateTime": "2020-11-20T19:54:05.000Z",
  "Size": 15
}
]
```

```
}
```

Identifiers

Identifier are the labels for output values. When creating filters, you use identifiers to narrow down your query results. In the following output example, all identifiers such as `Volumes`, `AvailabilityZone`, and `AttachTime` are highlighted.

```
$ aws ec2 describe-volumes
{
  "Volumes": [
    {
      "AvailabilityZone": "us-west-2a",
      "Attachments": [
        {
          "AttachTime": "2013-09-17T00:55:03.000Z",
          "InstanceId": "i-a071c394",
          "VolumeId": "vol-e11a5288",
          "State": "attached",
          "DeleteOnTermination": true,
          "Device": "/dev/sda1"
        }
      ],
      "VolumeType": "standard",
      "VolumeId": "vol-e11a5288",
      "State": "in-use",
      "SnapshotId": "snap-f23ec1c8",
      "CreateTime": "2013-09-17T00:55:03.000Z",
      "Size": 30
    },
    {
      "AvailabilityZone": "us-west-2a",
      "Attachments": [
        {
          "AttachTime": "2013-09-18T20:26:16.000Z",
          "InstanceId": "i-4b41a37c",
          "VolumeId": "vol-2e410a47",
          "State": "attached",
          "DeleteOnTermination": true,
          "Device": "/dev/sda1"
        }
      ],
      "VolumeType": "standard",
```

```

    "VolumeId": "vol-2e410a47",
    "State": "in-use",
    "SnapshotId": "snap-708e8348",
    "CreateTime": "2013-09-18T20:26:15.000Z",
    "Size": 8
  },
  {
    "AvailabilityZone": "us-west-2a",
    "Attachments": [
      {
        "AttachTime": "2020-11-20T19:54:06.000Z",
        "InstanceId": "i-1jd73kv8",
        "VolumeId": "vol-a1b3c7nd",
        "State": "attached",
        "DeleteOnTermination": true,
        "Device": "/dev/sda1"
      }
    ],
    "VolumeType": "standard",
    "VolumeId": "vol-a1b3c7nd",
    "State": "in-use",
    "SnapshotId": "snap-234087fb",
    "CreateTime": "2020-11-20T19:54:05.000Z",
    "Size": 15
  }
]
}

```

For more information, see [Identifiers](#) on the *JMESPath website*.

Selecting from a list

A list or array is an identifier that is followed by a square bracket "[" such as `Volumes` and `Attachments` in the [the section called "Before you start"](#).

Syntax

```
<listName>[ ]
```

To filter through all output from an array, you can use the wildcard notation. [Wildcard](#) expressions are expressions used to return elements using the `*` notation.

The following example queries all `Volumes` content.

```
$ aws ec2 describe-volumes \
  --query 'Volumes[*]'
[
  {
    "AvailabilityZone": "us-west-2a",
    "Attachments": [
      {
        "AttachTime": "2013-09-17T00:55:03.000Z",
        "InstanceId": "i-a071c394",
        "VolumeId": "vol-e11a5288",
        "State": "attached",
        "DeleteOnTermination": true,
        "Device": "/dev/sda1"
      }
    ],
    "VolumeType": "standard",
    "VolumeId": "vol-e11a5288",
    "State": "in-use",
    "SnapshotId": "snap-f23ec1c8",
    "CreateTime": "2013-09-17T00:55:03.000Z",
    "Size": 30
  },
  {
    "AvailabilityZone": "us-west-2a",
    "Attachments": [
      {
        "AttachTime": "2020-11-20T19:54:06.000Z",
        "InstanceId": "i-1jd73kv8",
        "VolumeId": "vol-a1b3c7nd",
        "State": "attached",
        "DeleteOnTermination": true,
        "Device": "/dev/sda1"
      }
    ],
    "VolumeType": "standard",
    "VolumeId": "vol-a1b3c7nd",
    "State": "in-use",
    "SnapshotId": "snap-234087fb",
    "CreateTime": "2020-11-20T19:54:05.000Z",
    "Size": 15
  }
]
```

To view a specific volume in the array by index, you call the array index. For example, the first item in the `Volumes` array has an index of 0, resulting in the `Volumes[0]` query. For more information about array indexes, see [index expressions](#) on the *JMESPath website*.

```
$ aws ec2 describe-volumes \
  --query 'Volumes[0]'
{
  "AvailabilityZone": "us-west-2a",
  "Attachments": [
    {
      "AttachTime": "2013-09-17T00:55:03.000Z",
      "InstanceId": "i-a071c394",
      "VolumeId": "vol-e11a5288",
      "State": "attached",
      "DeleteOnTermination": true,
      "Device": "/dev/sda1"
    }
  ],
  "VolumeType": "standard",
  "VolumeId": "vol-e11a5288",
  "State": "in-use",
  "SnapshotId": "snap-f23ec1c8",
  "CreateTime": "2013-09-17T00:55:03.000Z",
  "Size": 30
}
```

To view a specific range of volumes by index, use `slice` with the following syntax, where **start** is the starting array index, **stop** is the index where the filter stops processing, and **step** is the skip interval.

Syntax

```
<arrayName>[<start>:<stop>:<step>]
```

If any of these are omitted from the slice expression, they use the following default values:

- Start – The first index in the list, 0.
- Stop – The last index in the list.
- Step – No step skipping, where the value is 1.

To return only the first two volumes, you use a start value of 0, a stop value of 2, and a step value of 1 as shown in the following example.

```
$ aws ec2 describe-volumes \
  --query 'Volumes[0:2:1]'
[
  {
    "AvailabilityZone": "us-west-2a",
    "Attachments": [
      {
        "AttachTime": "2013-09-17T00:55:03.000Z",
        "InstanceId": "i-a071c394",
        "VolumeId": "vol-e11a5288",
        "State": "attached",
        "DeleteOnTermination": true,
        "Device": "/dev/sda1"
      }
    ],
    "VolumeType": "standard",
    "VolumeId": "vol-e11a5288",
    "State": "in-use",
    "SnapshotId": "snap-f23ec1c8",
    "CreateTime": "2013-09-17T00:55:03.000Z",
    "Size": 30
  },
  {
    "AvailabilityZone": "us-west-2a",
    "Attachments": [
      {
        "AttachTime": "2013-09-18T20:26:16.000Z",
        "InstanceId": "i-4b41a37c",
        "VolumeId": "vol-2e410a47",
        "State": "attached",
        "DeleteOnTermination": true,
        "Device": "/dev/sda1"
      }
    ],
    "VolumeType": "standard",
    "VolumeId": "vol-2e410a47",
    "State": "in-use",
    "SnapshotId": "snap-708e8348",
    "CreateTime": "2013-09-18T20:26:15.000Z",
    "Size": 8
  }
]
```



```
}  
]
```

Since this example contains default values, you can shorten the slice from `Volumes[0:2:1]` to `Volumes[:2]`.

The following example omits default values and returns every two volumes in the entire array.

```
$ aws ec2 describe-volumes \  
  --query 'Volumes[:2]'  
[  
  {  
    "AvailabilityZone": "us-west-2a",  
    "Attachments": [  
      {  
        "AttachTime": "2013-09-17T00:55:03.000Z",  
        "InstanceId": "i-a071c394",  
        "VolumeId": "vol-e11a5288",  
        "State": "attached",  
        "DeleteOnTermination": true,  
        "Device": "/dev/sda1"  
      }  
    ],  
    "VolumeType": "standard",  
    "VolumeId": "vol-e11a5288",  
    "State": "in-use",  
    "SnapshotId": "snap-f23ec1c8",  
    "CreateTime": "2013-09-17T00:55:03.000Z",  
    "Size": 30  
  },  
  {  
    "AvailabilityZone": "us-west-2a",  
    "Attachments": [  
      {  
        "AttachTime": "2020-11-20T19:54:06.000Z",  
        "InstanceId": "i-1jd73kv8",  
        "VolumeId": "vol-a1b3c7nd",  
        "State": "attached",  
        "DeleteOnTermination": true,  
        "Device": "/dev/sda1"  
      }  
    ],  
    "VolumeType": "standard",
```

```
"VolumeId": "vol-a1b3c7nd",
"State": "in-use",
"SnapshotId": "snap-234087fb",
"CreateTime": "2020-11-20T19:54:05.000Z",
"Size": 15
}
]
```

Steps can also use negative numbers to filter in the reverse order of an array as shown in the following example.

```
$ aws ec2 describe-volumes \
  --query 'Volumes[::-2]'
[
  {
    "AvailabilityZone": "us-west-2a",
    "Attachments": [
      {
        "AttachTime": "2020-11-20T19:54:06.000Z",
        "InstanceId": "i-1jd73kv8",
        "VolumeId": "vol-a1b3c7nd",
        "State": "attached",
        "DeleteOnTermination": true,
        "Device": "/dev/sda1"
      }
    ],
    "VolumeType": "standard",
    "VolumeId": "vol-a1b3c7nd",
    "State": "in-use",
    "SnapshotId": "snap-234087fb",
    "CreateTime": "2020-11-20T19:54:05.000Z",
    "Size": 15
  },
  {
    "AvailabilityZone": "us-west-2a",
    "Attachments": [
      {
        "AttachTime": "2013-09-17T00:55:03.000Z",
        "InstanceId": "i-a071c394",
        "VolumeId": "vol-e11a5288",
        "State": "attached",
        "DeleteOnTermination": true,
        "Device": "/dev/sda1"
      }
    ]
  }
]
```

```

    }
  ],
  "VolumeType": "standard",
  "VolumeId": "vol-e11a5288",
  "State": "in-use",
  "SnapshotId": "snap-f23ec1c8",
  "CreateTime": "2013-09-17T00:55:03.000Z",
  "Size": 30
}
]

```

For more information, see [Slices](#) on the *JMESPath website*.

Filtering nested data

To narrow the filtering of the `Volumes[*]` for nested values, you use subexpressions by appending a period and your filter criteria.

Syntax

```
<expression>.<expression>
```

The following example shows all Attachments information for all volumes.

```

$ aws ec2 describe-volumes \
  --query 'Volumes[*].Attachments'
[
  [
    {
      "AttachTime": "2013-09-17T00:55:03.000Z",
      "InstanceId": "i-a071c394",
      "VolumeId": "vol-e11a5288",
      "State": "attached",
      "DeleteOnTermination": true,
      "Device": "/dev/sda1"
    }
  ],
  [
    {
      "AttachTime": "2013-09-18T20:26:16.000Z",
      "InstanceId": "i-4b41a37c",
      "VolumeId": "vol-2e410a47",
      "State": "attached",

```

```

    "DeleteOnTermination": true,
    "Device": "/dev/sda1"
  }
],
[
  {
    "AttachTime": "2020-11-20T19:54:06.000Z",
    "InstanceId": "i-1jd73kv8",
    "VolumeId": "vol-a1b3c7nd",
    "State": "attached",
    "DeleteOnTermination": true,
    "Device": "/dev/sda1"
  }
]
]
```

To filter further into the nested values, append the expression for each nested identifier. The following example lists the State for all Volumes.

```

$ aws ec2 describe-volumes \
  --query 'Volumes[*].Attachments[*].State'
[
  [
    "attached"
  ],
  [
    "attached"
  ],
  [
    "attached"
  ]
]
```

Flattening results

For more information, see [SubExpressions](#) on the *JMESPath website*.

You can flatten the results for `Volumes[*].Attachments[*].State` by removing the wildcard notation resulting in the `Volumes[*].Attachments[].State` query. Flattening often is useful to improve the readability of results.

```

$ aws ec2 describe-volumes \
```

```
--query 'Volumes[*].Attachments[].State'
[
  "attached",
  "attached",
  "attached"
]
```

For more information, see [Flatten](#) on the *JMESPath website*.

Filtering for specific values

To filter for specific values in a list, you use a filter expression as shown in the following syntax.

Syntax

```
? <expression> <comparator> <expression>]
```

Expression comparators include ==, !=, <, <=, >, and >= . The following example filters for the VolumeIds for all Volumes in an AttachedState.

```
$ aws ec2 describe-volumes \
  --query 'Volumes[*].Attachments[?State==`attached`].VolumeId'
[
  [
    "vol-e11a5288"
  ],
  [
    "vol-2e410a47"
  ],
  [
    "vol-a1b3c7nd"
  ]
]
```

This can then be flattened resulting in the following example.

```
$ aws ec2 describe-volumes \
  --query 'Volumes[*].Attachments[?State==`attached`].VolumeId[]'
[
  "vol-e11a5288",
  "vol-2e410a47",
  "vol-a1b3c7nd"
]
```

```
]
```

The following example filters for the VolumeIds of all Volumes that have a size less than 20.

```
$ aws ec2 describe-volumes \  
  --query 'Volumes[?Size < `20`].VolumeId'  
[  
  "vol-2e410a47",  
  "vol-a1b3c7nd"  
]
```

For more information, see [Filter Expressions](#) on the *JMESPath website*.

Piping expressions

You can pipe results of a filter to a new list, and then filter the result with another expression using the following syntax:

Syntax

```
<expression> | <expression>]
```

The following example takes the filter results of the `Volumes[*].Attachments[].InstanceId` expression and outputs the first result in the array.

```
$ aws ec2 describe-volumes \  
  --query 'Volumes[*].Attachments[].InstanceId | [0]'  
"i-a071c394"
```

This example does this by first creating the array from the following expression.

```
$ aws ec2 describe-volumes \  
  --query 'Volumes[*].Attachments[].InstanceId'  
"i-a071c394",  
"i-4b41a37c",  
"i-1jd73kv8"
```

And then returns the first element in that array.

```
"i-a071c394"
```

For more information, see [Pipe Expressions](#) on the *JMESPath website*.

Filtering for multiple identifier values

To filter for multiple identifiers, you use a multiselect list by using the following syntax:

Syntax

```
<listName>[].[<expression>, <expression>]
```

In the following example, `VolumeId` and `VolumeType` are filtered in the `Volumes` list resulting in the following expression.

```
$ aws ec2 describe-volumes \  
  --query 'Volumes[][VolumeId, VolumeType]'  
[  
  [  
    "vol-e11a5288",  
    "standard"  
  ],  
  [  
    "vol-2e410a47",  
    "standard"  
  ],  
  [  
    "vol-a1b3c7nd",  
    "standard"  
  ]  
]
```

To add nested data to the list, you add another multiselect list. The following example expands on the previous example by also filtering for `InstanceId` and `State` in the nested `Attachments` list. This results in the following expression.

```
$ aws ec2 describe-volumes \  
  --query 'Volumes[][VolumeId, VolumeType, Attachments[][InstanceId, State]]'  
[  
  [  
    "vol-e11a5288",  
    "standard",  
    [  
      [  

```

```

        "i-a071c394",
        "attached"
    ]
  ],
  [
    "vol-2e410a47",
    "standard",
    [
      [
        "i-4b41a37c",
        "attached"
      ]
    ]
  ],
  [
    "vol-a1b3c7nd",
    "standard",
    [
      [
        "i-1jd73kv8",
        "attached"
      ]
    ]
  ]
]

```

To be more readable, flatten out the expression as shown in the following example.

```

$ aws ec2 describe-volumes \
  --query 'Volumes[][VolumeId, VolumeType, Attachments[][InstanceId, State]][]'
[
  "vol-e11a5288",
  "standard",
  [
    "i-a071c394",
    "attached"
  ],
  "vol-2e410a47",
  "standard",
  [
    "i-4b41a37c",
    "attached"
  ]
]

```



```

],
"vol-a1b3c7nd",
"standard",
[
  "i-1jd73kv8",
  "attached"
]
]

```

For more information, see [Multiselect list](#) on the *JMESPath website*.

Adding labels to identifier values

To make this output easier to read, use a multiselect hash with the following syntax.

Syntax

```
<listName>[].{<label>: <expression>, <label>: <expression>}
```

Your identifier label does not need to be the same as the name of the identifier. The following example uses the label `VolumeType` for the `VolumeType` values.

```

$ aws ec2 describe-volumes \
  --query 'Volumes[].{VolumeType: VolumeType}'
[
  {
    "VolumeType": "standard",
  },
  {
    "VolumeType": "standard",
  },
  {
    "VolumeType": "standard",
  }
]

```

For simplicity, the following example keeps the identifier names for each label and displays the `VolumeId`, `VolumeType`, `InstanceId`, and `State` for all volumes:

```

$ aws ec2 describe-volumes \
  --query 'Volumes[].{VolumeId: VolumeId, VolumeType: VolumeType, InstanceId:
    Attachments[0].InstanceId, State: Attachments[0].State}'

```

```
[
  {
    "VolumeId": "vol-e11a5288",
    "VolumeType": "standard",
    "InstanceId": "i-a071c394",
    "State": "attached"
  },
  {
    "VolumeId": "vol-2e410a47",
    "VolumeType": "standard",
    "InstanceId": "i-4b41a37c",
    "State": "attached"
  },
  {
    "VolumeId": "vol-a1b3c7nd",
    "VolumeType": "standard",
    "InstanceId": "i-1jd73kv8",
    "State": "attached"
  }
]
```

For more information, see [Multiselect hash](#) on the *JMESPath website*.

Functions

The JMESPath syntax contains many functions that you can use for your queries. For information on JMESPath functions, see [Built-in Functions](#) on the *JMESPath website*.

To demonstrate how you can incorporate a function into your queries, the following example uses the `sort_by` function. The `sort_by` function sorts an array using an expression as the sort key using the following syntax:

Syntax

```
sort_by(<listName>, <sort expression>)[].<expression>
```

The following example uses the previous [multiselect hash example](#) and sorts the output by `VolumeId`.

```
$ aws ec2 describe-volumes \
  --query 'sort_by(Volumes, &VolumeId)[].{VolumeId: VolumeId, VolumeType: VolumeType,
  InstanceId: Attachments[0].InstanceId, State: Attachments[0].State}'
```

```
[
  {
    "VolumeId": "vol-2e410a47",
    "VolumeType": "standard",
    "InstanceId": "i-4b41a37c",
    "State": "attached"
  },
  {
    "VolumeId": "vol-a1b3c7nd",
    "VolumeType": "standard",
    "InstanceId": "i-1jd73kv8",
    "State": "attached"
  },
  {
    "VolumeId": "vol-e11a5288",
    "VolumeType": "standard",
    "InstanceId": "i-a071c394",
    "State": "attached"
  }
]
```

For more information, see [sort_by](#) on the *JMESPath website*.

Advanced --query examples

To extract information from a specific item

The following example uses the `--query` parameter to find a specific item in a list and then extracts information from that item. The example lists all of the `AvailabilityZones` associated with the specified service endpoint. It extracts the item from the `ServiceDetails` list that has the specified `ServiceName`, then outputs the `AvailabilityZones` field from that selected item.

```
$ aws --region us-east-1 ec2 describe-vpc-endpoint-services \
  --query 'ServiceDetails[?ServiceName==`com.amazonaws.us-
east-1.ecs`].AvailabilityZones'
[
  [
    "us-east-1a",
    "us-east-1b",
    "us-east-1c",
    "us-east-1d",
    "us-east-1e",
    "us-east-1f"
  ]
]
```

```
]
]
```

To show snapshots after the specified creation date

The following example shows how to list all of your snapshots that were created after a specified date, including only a few of the available fields in the output.

```
$ aws ec2 describe-snapshots --owner self \
  --output json \
  --query 'Snapshots[?StartTime>=`2018-02-07`].
{Id:SnapshotId,VID:VolumeId,Size:VolumeSize}'
[
  {
    "id": "snap-0effb42b7a1b2c3d4",
    "vid": "vol-0be9bb0bf12345678",
    "Size": 8
  }
]
```

To show the most recent AMIs

The following example lists the five most recent Amazon Machine Images (AMIs) that you created, sorted from most recent to oldest.

```
$ aws ec2 describe-images \
  --owners self \
  --query 'reverse(sort_by(Images,&CreationDate))[:5].{id:ImageId,date:CreationDate}'
[
  {
    "id": "ami-0a1b2c3d4e5f60001",
    "date": "2018-11-28T17:16:38.000Z"
  },
  {
    "id": "ami-0a1b2c3d4e5f60002",
    "date": "2018-09-15T13:51:22.000Z"
  },
  {
    "id": "ami-0a1b2c3d4e5f60003",
    "date": "2018-08-19T10:22:45.000Z"
  },
  {
    "id": "ami-0a1b2c3d4e5f60004",
```

```
    "date": "2018-05-03T12:04:02.000Z"
  },
  {
    "id": "ami-0a1b2c3d4e5f60005",
    "date": "2017-12-13T17:16:38.000Z"
  }
]
```

To show unhealthy Auto Scaling instances

The following example shows only the InstanceId for any unhealthy instances in the specified Auto Scaling group.

```
$ aws autoscaling describe-auto-scaling-groups \
  --auto-scaling-group-name My-AutoScaling-Group-Name \
  --output text \
  --query 'AutoScalingGroups[*].Instances[?HealthStatus==`Unhealthy`].InstanceId'
```

To include volumes with the specified tag

The following example describes all instances with a test tag. As long as there is another tag beside test attached to the volume, the volume is still returned in the results.

The below expression to return all tags with the test tag in an array. Any tags that are not the test tag contain a null value.

```
$ aws ec2 describe-volumes \
  --query 'Volumes[*].Tags[?Value == `test`]'
```

To exclude volumes with the specified tag

The following example describes all instances without a test tag. Using a simple ?Value != `test` expression does not work for excluding a volume as volumes can have multiple tags. As long as there is another tag beside test attached to the volume, the volume is still returned in the results.

To exclude all volumes with the test tag, start with the below expression to return all tags with the test tag in an array. Any tags that are not the test tag contain a null value.

```
$ aws ec2 describe-volumes \
  --query 'Volumes[*].Tags[?Value == `test`]'
```

Then filter out all the positive test results using the `not_null` function.

```
$ aws ec2 describe-volumes \  
  --query 'Volumes[!not_null(Tags[?Value == `test`].Value)]'
```

Pipe the results to flatten out the results resulting in the following query.

```
$ aws ec2 describe-volumes \  
  --query 'Volumes[!not_null(Tags[?Value == `test`].Value)] | []'
```

Combining server-side and client-side filtering

You can use server-side and client-side filtering together. Server-side filtering is completed first, which sends the data to the client that the `--query` parameter then filters. If you're using large data sets, using server-side filtering first can lower the amount of data sent to the client for each AWS CLI call, while still keeping the powerful customization that client-side filtering provides.

The following example lists Amazon EC2 volumes using both server-side and client-side filtering. The service filters a list of all attached volumes in the `us-west-2a` Availability Zone. The `--query` parameter further limits the output to only those volumes with a `Size` value that is larger than 50, and shows only the specified fields with user-defined names.

```
$ aws ec2 describe-volumes \  
  --filters "Name=availability-zone,Values=us-west-2a" "Name=status,Values=attached" \  
  \  
  --query 'Volumes[?Size > `50`].{Id:VolumeId,Size:Size,Type:VolumeType}' \  
  [ \  
    { \  
      "Id": "vol-0be9bb0bf12345678", \  
      "Size": 80, \  
      "VolumeType": "gp2" \  
    } \  
  ]
```

The following example retrieves a list of images that meet several criteria. It then uses the `--query` parameter to sort the output by `CreationDate`, selecting only the most recent. Finally, it displays the `ImageId` of that one image.

```
$ aws ec2 describe-images \  
  --owners amazon \  
  \  
  --query 'Images[?CreationDate >=`2017-01-01`].ImageId'
```

```
--filters "Name=name,Values=amzn*gp2" "Name=virtualization-type,Values=hvm"
"Name=root-device-type,Values=ebs" \
--query "sort_by(Images, &CreationDate)[-1].ImageId" \
--output text
ami-00ced3122871a4921
```

The following example displays the number of available volumes that are more than 1000 IOPS by using `length` to count how many are in a list.

```
$ aws ec2 describe-volumes \
  --filters "Name=status,Values=available" \
  --query 'length(Volumes[?Iops > `1000`])'
3
```

Additional resources

AWS CLI autoprompt

When beginning to use filter expressions, you can use the auto-prompt feature in the AWS CLI version 2. The auto-prompt feature provides a preview when you press the **F5** key. For more information, see [the section called "Auto-prompt"](#).

JMESPath Terminal

JMESPath Terminal is an interactive terminal command to experiment with JMESPath expressions that are used for client-side filtering. Using the `jpترم` command, the terminal shows immediate query results as you're typing. You can directly pipe AWS CLI output to the terminal, enabling advanced querying experimentation.

The following example pipes `aws ec2 describe-volumes` output directly to JMESPath Terminal.

```
$ aws ec2 describe-volumes | jpترم
```

For more information on JMESPath Terminal and installation instructions, see [JMESPath Terminal](#) on *GitHub*.

jq utility

The `jq` utility provides you a way to transform your output on the client-side to an output format you desire. For more information on `jq` and installation instructions, see [jq](#) on *GitHub*.

Return codes from the AWS CLI

The return code is usually a hidden code sent after running a AWS Command Line Interface (AWS CLI) command which describes the status of the command. You can use the `echo` command to display the code sent from the last AWS CLI command and use these codes to determine if a command was successful or if it failed, and why a command might have an error. In addition to the return codes, you can view more details about a failure by running your commands with the `--debug` switch. This switch produces a detailed report of the steps the AWS CLI uses to process the command, and what the result of each step was.

To determine the return code of an AWS CLI command, run one of the following commands immediately after running the CLI command.

Linux and macOS

```
$ echo $?  
0
```

Windows PowerShell

```
PS> echo $lastexitcode  
0
```

Windows Command Prompt

```
C:\> echo %errorlevel%  
0
```

The following are the return code values that can be returned at the end of running an AWS Command Line Interface (AWS CLI) command.

Code	Meaning
0	The service responded with an HTTP response status code of 200 indicating that there were no errors generated by the AWS CLI and AWS service the request was sent to.
1	One or more Amazon S3 transfer operations failed. <i>Limited to S3 commands.</i>

Code	Meaning
2	<p>The meaning of this return code depends on the command:</p> <ul style="list-style-type: none">• <i>Applicable to all AWS CLI commands</i> – the command entered couldn't be parsed. Parsing failures can be caused by, but aren't limited to, missing required subcommands or arguments, or using unknown commands or arguments.• <i>Limited to S3 commands</i> – One or more files marked for transfer were skipped during the transfer process. However, all other files marked for transfer were successfully transferred. Files that are skipped during the transfer process include: files that don't exist; files that are character special devices, block special device, FIFO queues, or sockets; and files that the user doesn't have read permissions to.
130	<p>The command was interrupted by a SIGINT. This is the signal sent by you to cancel a command with <code>Ctrl+C</code>.</p>
252	<p>Command syntax was invalid, an unknown parameter was provided, or a parameter value was incorrect and prevented the command from running.</p>
253	<p>The system environment or configuration was invalid. While the command provided might be syntactically valid, missing configuration or credentials prevented the command from running.</p>
254	<p>The command successfully parsed and a request made to the specified service but the service returned an error. This will generally indicate incorrect API usage or other service specific issues.</p>
255	<p>The command failed. There were errors generated by the AWS CLI or by the AWS service to which the request was sent.</p>

Interactive commands using the AWS CLI wizards

The AWS Command Line Interface (AWS CLI) provides the ability to use a wizard for some commands. To contribute or view the full list of available AWS CLI wizards, see the [AWS CLI wizards folder](#) on GitHub.

How it works

Similar to the AWS console, the AWS CLI has a UI wizard that guides you through managing your AWS resources. To use the wizard, you call the `wizard` subcommand and the wizard name after the service name in a command. The command structure is as follows:

Syntax:

```
$ aws <command> wizard <wizardName>
```

The following example is calling the wizard to create a new dynamodb table.

```
$ aws dynamodb wizard new-table
```

`aws configure` is the only wizard that does not have a wizard name. When running the wizard, run the `aws configure wizard` command as the following example demonstrates:

```
$ aws configure wizard
```

After calling a wizard, a form in the shell is displayed. For each parameter, you are either provided a list of options to select from or prompted to enter in a string. To select from a list, use your up and down arrow keys and press **ENTER**. To view details on an option, press the right arrow key. When you've finished filling out a parameter, press **ENTER**.

```
$ aws configure wizard
What would you like to configure
> Static Credentials
  Assume Role
  Process Provider
  Additional CLI configuration
Enter the name of the profile:
Enter your Access Key Id:
Enter your Secret Access Key:
```

To edit previous prompts, use **SHIFT + TAB**. For some wizards, after filling in all prompts, you can preview an AWS CloudFormation template or the AWS CLI command filled with your information. This preview mode is useful to learn the AWS CLI, service APIs, and creating templates for scripts.

Press **ENTER** after previewing or the last prompt to run the final command.

```
$ aws configure wizard
What would you like to configure
Enter the name of the profile: testWizard
Enter your Access Key Id: AB1C2D3EF4GH5I678J90K
Enter your Secret Access Key: ab1c2def34gh5i67j8k90l1mnop2qr3s45tu678v90
<ENTER>
```

Create and use AWS CLI command shortcuts called aliases

Aliases are shortcuts you can create in the AWS Command Line Interface (AWS CLI) to shorten commands or scripts that you frequently use. You create aliases in the `aliases` file located in your configuration folder.

Topics

- [Prerequisites](#)
- [Step 1: Creating the alias file](#)
- [Step 2: Creating an alias](#)
- [Step 3: Calling an alias](#)
- [Alias repository examples](#)
- [Resources](#)

Prerequisites

To use alias commands, you need to complete the following:

- Install and configure the AWS CLI. For more information, see [the section called “Install/Update”](#) and [Authentication and access credentials](#).
- Use a minimum AWS CLI version of 1.11.24 or 2.0.0.
- (Optional) To use AWS CLI alias bash scripts, you must use a bash-compatible terminal.

Step 1: Creating the alias file

To create the `aliases` file, you can use your file navigation and a text editor, or use your preferred terminal by using the step-by-step procedure. To quickly create your alias file, use the following command block.

Linux and macOS

```
$ mkdir -p ~/.aws/cli
$ echo '[toplevel]' > ~/.aws/cli/alias
```

Windows

```
C:\> md %USERPROFILE%\.aws\cli
C:\> echo [toplevel] > %USERPROFILE%\.aws\cli\alias
```

To create the alias file

1. Create a folder named `cli` in your AWS CLI configuration folder. By default the configuration folder is `~/.aws/` on Linux or macOS and `%USERPROFILE%\.aws\` on Windows. You can create this through your file navigation or by using the following command.

Linux and macOS

```
$ mkdir -p ~/.aws/cli
```

Windows

```
C:\> md %USERPROFILE%\.aws\cli
```

The resulting `cli` folder default path is `~/.aws/cli/` on Linux or macOS and `%USERPROFILE%\.aws\cli` on Windows.

2. In the `cli` folder, create a text file named `alias` with no extension and add `[toplevel]` to the first line. You can create this file through your preferred text editor or use the following command.

Linux and macOS

```
$ echo '[toplevel]' > ~/.aws/cli/alias
```

Windows

```
C:\> echo [toplevel] > %USERPROFILE%\.aws\cli\alias
```

Step 2: Creating an alias

You can create an alias using basic commands or bash scripting.

Creating a basic command alias

You can create your alias by adding a command using the following syntax in the `alias` file you created in the previous step.

Syntax

```
aliasname = command [--options]
```

The *aliasname* is what you call your alias. The *command* is the command you want to call, which can include other aliases. You can include options or parameters in your alias, or add them when calling your alias.

The following example creates an alias named `aws whoami` using the [aws sts get-caller-identity](#) command. Since this alias calls an existing AWS CLI command, you can write the command without the `aws` prefix.

```
whoami = sts get-caller-identity
```

The following example takes the previous `whoami` example and adds the `Account` filter and text output options.

```
whoami2 = sts get-caller-identity --query Account --output text
```

Creating a sub-command alias

Note

The sub-command alias feature requires a minimum AWS CLI version of 1.11.24 or 2.0.0

You can create an alias for sub-commands by adding a command using the following syntax in the `alias` file you created in the previous step.

Syntax

```
[command commandGroup]  
aliasname = command [--options]
```

The *commandGroup* is the command namespace, e.g. The command `aws ec2 describe-regions` is under the `ec2` command group. The *aliasname* is what you call your alias. The *command* is the command you want to call, which can include other aliases. You can include options or parameters in your alias, or add them when calling your alias.

The following example creates an alias named `aws ec2 regions` using the [aws ec2 describe-regions](#) command. Since this alias calls an existing AWS CLI command under the `ec2` command namespace, you can write the command without the `aws ec2` prefix.

```
[command ec2]  
regions = describe-regions --query Regions[].RegionName
```

To create aliases from commands outside of the command namespace, prefix the full command with an exclamation mark. The following example creates an alias named `aws ec2 instance-profiles` using the [aws iam list-instance-profiles](#) command.

```
[command ec2]  
instance-profiles = !aws iam list-instance-profiles
```

Note

Aliases only use existing command namespaces and you cannot create new ones. e.g. You can't create an alias with the `[command johnsmith]` section as the `johnsmith` command namespace does not already exist.

Creating a bash scripting alias

Warning

To use AWS CLI alias bash scripts, you must use a bash-compatible terminal

You can create an alias using bash scripts for more advanced processes using the following syntax.

Syntax

```
aliasname =  
    !f() {  
        script content  
    }; f
```

The *aliasname* is what you call your alias and *script content* is the script you want to run when you call the alias.

The following example uses `opendns` to output your current IP address. Since you can use aliases in other aliases, the following `myip` alias is useful to allow or revoke access for your IP address from within other aliases.

```
myip =  
    !f() {  
        dig +short myip.opendns.com @resolver1.opendns.com  
    }; f
```

The following script example calls the previous `aws myip` alias to authorize your IP address for an Amazon EC2 security group ingress.

```
authorize-my-ip =  
    !f() {  
        ip=$(aws myip)  
        aws ec2 authorize-security-group-ingress --group-id ${1} --cidr $ip/32 --protocol  
        tcp --port 22  
    }; f
```

When you call aliases that use bash scripting, the variables are always passed in the order that you entered them. In bash scripting, the variable names are not taken into consideration, only the order they appear. In the following `textalert` alias example, the variable for the `--message` option is first and `--phone-number` option is second.

```
textalert =  
    !f() {  
        aws sns publish --message "${1}" --phone-number ${2}  
    }; f
```

Step 3: Calling an alias

To run the alias you created in your `alias` file use the following syntax. You can add additional options when you call your alias.

Syntax

```
$ aws aliasname
```

The following example uses the `aws whoami` command alias.

```
$ aws  
whoami  
{  
  "UserId": "A12BCD34E5FGHI6JKLM",  
  "Account": "1234567890987",  
  "Arn": "arn:aws:iam::1234567890987:user/userName"  
}
```

The following example uses the `aws whoami` alias with additional options to only return the Account number in text output.

```
$ aws whoami --query Account --output  
text  
1234567890987
```

The following example uses the `aws ec2 regions` [sub-command alias](#).

```
$ aws ec2  
regions  
[  
  "ap-south-1",  
  "eu-north-1",  
  "eu-west-3",  
  "eu-west-2",  
  ...
```

Calling an alias using bash scripting variables

When you call aliases that use bash scripting, variables are passed in the order they are entered. In bash scripting, the name of the variables are not taken into consideration, only the order they

appear. For example, in the following `textalert` alias, the variable for the option `--message` is first and `--phone-number` is second.

```
textalert =
  !f() {
    aws sns publish --message "${1}" --phone-number ${2}
  }; f
```

When you call the `textalert` alias, you need to pass variables in the same order as they are run in the alias. In the following example we use the variables `$message` and `$phone`. The `$message` variable is passed as `${1}` for the `--message` option and the `$phone` variable is passed as `${2}` for the `--phone-number` option. This results in successfully calling the `textalert` alias to send a message.

```
$ aws textalert $message
$phone
{
  "MessageId": "1ab2cd3e4-fg56-7h89-i01j-2k1mn34567"
}
```

In the following example, the order is switched when calling the alias to `$phone` and `$message`. The `$phone` variable is passed as `${1}` for the `--message` option and the `$message` variable is passed as `${2}` for the `--phone-number` option. Since the variables are out of order, the alias passes the variables incorrectly. This causes an error because the contents of `$message` do not match the phone number formatting requirements for the `--phone-number` option.

```
$ aws textalert $phone
$message
usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:

aws help
aws <command> help
aws <command> <subcommand> help

Unknown options: text
```

Alias repository examples

The [AWS CLI alias repository](#) on *GitHub* contains AWS CLI alias examples created by the AWS CLI developer team and community. You can use the entire `alias` file example or take individual aliases for your own use.

Warning

Running the commands in this section deletes your existing `alias` file. To avoid overwriting your existing alias file, change your download location.

To use aliases from the repository

1. Install Git. For installation instructions, see [Getting Started - Installing Git](#) in the *Git Documentation*.
2. Install the `jp` command. The `jp` command is used in the `tostring` alias. For installation instructions, see the [JMESPath \(jp\) README.md](#) on *GitHub*.
3. Install the `jq` command. The `jq` command is used in the `tostring-with-jq` alias. For installation instructions, see the [JSON processor \(jq\)](#) on *GitHub*.
4. Download the `alias` file by doing one of the following:
 - Run the following commands that downloads from the repository and copies the `alias` file to your configuration folder.

Linux and macOS

```
$ git clone https://github.com/aws-labs/awscli-aliases.git
$ mkdir -p ~/.aws/cli
$ cp awscli-aliases/alias ~/.aws/cli/alias
```

Windows

```
C:\> git clone https://github.com/aws-labs/awscli-aliases.git
C:\> md %USERPROFILE%\aws\cli
C:\> copy awscli-aliases\alias %USERPROFILE%\aws\cli
```

- Download directly from the repository and save to the `cli` folder in your AWS CLI configuration folder. By default the configuration folder is `~/ .aws/` on Linux or macOS and `%USERPROFILE%\ .aws\` on Windows.
5. To verify the aliases are working, run the following alias.

```
$ aws whoami
```

This displays the same response as the `aws sts get-caller-identity` command:

```
{
  "Account": "012345678901",
  "UserId": "AIUAINBADX2VEG2TC6HD6",
  "Arn": "arn:aws:iam::012345678901:user/myuser"
}
```

Resources

- The [AWS CLI alias repository](#) on *GitHub* contains AWS CLI alias examples created by the AWS CLI developer team and the contribution of the AWS CLI community.
- The alias feature announcement from [AWS re:Invent 2016: The Effective AWS CLI User](#) on *YouTube*.
- [aws sts get-caller-identity](#)
- [aws ec2 describe-instances](#)
- [aws sns publish](#)

Code examples

This chapter provides a collection of examples that show you how to use the AWS Command Line Interface (AWS CLI) with AWS services.

The AWS CLI has the following types of examples in this guide:

- **[Guided command examples](#)** - Guided command examples for the AWS CLI User Guide on how to use the AWS CLI with some AWS services. These are often more detailed examples than the examples from the [AWS CLI version 2 reference guide](#).
- **[AWS CLI command examples](#)** - Open source command examples that are also available in the [AWS CLI version 2 reference guide](#). Command examples are hosted in the [AWS CLI](#) repository on *GitHub*.
- **[AWS CLI using Bash scripting code examples](#)** - Open source bash scripting examples. Bash scripting examples are hosted in the [AWS Code Examples Repository](#) on *GitHub*.

Example feedback

Can't find what you need? Request a command example by using the **Provide feedback** link at the bottom of this page or on the relevant command page in the [AWS CLI version 2 reference guide](#).

Want to contribute? Contribute AWS CLI command examples in the [AWS Code Examples Repository](#) on *GitHub*. For more information on contributing, see [AWS CLI code example contribution quick steps](#) on *GitHub pages*.

Guided AWS CLI command examples

This section provides examples that show how to use the AWS Command Line Interface (AWS CLI) to access various AWS services.

Note

For a complete reference of all the available commands for each service, see the [AWS CLI version 2 reference guide](#), or use the built-in command line help. For more information, see [Get help with the AWS CLI](#).

Services

- [Use Amazon DynamoDB with the AWS CLI](#)
- [Use Amazon EC2 with the AWS CLI](#)
- [Use Amazon S3 Glacier with the AWS CLI](#)
- [Use AWS Identity and Access Management from the AWS CLI](#)
- [Use Amazon S3 with the AWS CLI](#)
- [Use Amazon SNS with the AWS CLI](#)

Use Amazon DynamoDB with the AWS CLI

An introduction to Amazon DynamoDB

[What is Amazon DynamoDB?](#)

The AWS Command Line Interface (AWS CLI) provides support for all of the AWS database services, including Amazon DynamoDB. You can use the AWS CLI for impromptu operations, such as creating a table. You can also use it to embed DynamoDB operations within utility scripts.

For more information about using the AWS CLI with DynamoDB, see [dynamodb](#) in the *AWS CLI Command Reference*.

To list the AWS CLI commands for DynamoDB, use the following command.

```
$ aws dynamodb help
```

Topics

- [Prerequisites](#)
- [Creating and using DynamoDB tables](#)
- [Using DynamoDB Local](#)
- [Resources](#)

Prerequisites

To run the dynamodb commands, you need to:

- Install and configure the AWS CLI. For more information, see [the section called “Install/Update”](#) and [Authentication and access credentials](#).

Creating and using DynamoDB tables

The command line format consists of an DynamoDB command name, followed by the parameters for that command. The AWS CLI supports the CLI [shorthand syntax](#) for the parameter values, and full JSON.

The following example creates a table named MusicCollection.

```
$ aws dynamodb create-table \  
  --table-name MusicCollection \  
  --attribute-definitions AttributeName=Artist,AttributeType=S  
  AttributeName=SongTitle,AttributeType=S \  
  --key-schema AttributeName=Artist,KeyType=HASH  
  AttributeName=SongTitle,KeyType=RANGE \  
  --provisioned-throughput ReadCapacityUnits=1,WriteCapacityUnits=1
```

You can add new lines to the table with commands similar to those shown in the following example. These examples use a combination of shorthand syntax and JSON.

```
$ aws dynamodb put-item \  
  --table-name MusicCollection \  
  --item '{  
    "Artist": {"S": "No One You Know"},  
    "SongTitle": {"S": "Call Me Today"} ,  
    "AlbumTitle": {"S": "Somewhat Famous"}  
  }' \  
  --return-consumed-capacity TOTAL  
{  
  "ConsumedCapacity": {  
    "CapacityUnits": 1.0,  
    "TableName": "MusicCollection"  
  }  
}
```

```
$ aws dynamodb put-item \  
  --table-name MusicCollection \  
  --item '{
```

```

    "Artist": {"S": "Acme Band"},
    "SongTitle": {"S": "Happy Day"} ,
    "AlbumTitle": {"S": "Songs About Life"}
  }' \
--return-consumed-capacity TOTAL
{
  "ConsumedCapacity": {
    "CapacityUnits": 1.0,
    "TableName": "MusicCollection"
  }
}

```

It can be difficult to compose valid JSON in a single-line command. To make this easier, the AWS CLI can read JSON files. For example, consider the following JSON snippet, which is stored in a file named `expression-attributes.json`.

```

{
  ":v1": {"S": "No One You Know"},
  ":v2": {"S": "Call Me Today"}
}

```

You can use that file to issue a query request using the AWS CLI. In the following example, the content of the `expression-attributes.json` file is used as the value for the `--expression-attribute-values` parameter.

```

$ aws dynamodb query --table-name MusicCollection \
  --key-condition-expression "Artist = :v1 AND SongTitle = :v2" \
  --expression-attribute-values file://expression-attributes.json
{
  "Count": 1,
  "Items": [
    {
      "AlbumTitle": {
        "S": "Somewhat Famous"
      },
      "SongTitle": {
        "S": "Call Me Today"
      },
      "Artist": {
        "S": "No One You Know"
      }
    }
  ]
}

```

```
  ],  
  "ScannedCount": 1,  
  "ConsumedCapacity": null  
}
```

Using DynamoDB Local

In addition to DynamoDB, you can use the AWS CLI with DynamoDB Local. DynamoDB Local is a small client-side database and server that mimics the DynamoDB service. DynamoDB Local enables you to write applications that use the DynamoDB API, without manipulating any tables or data in the DynamoDB web service. Instead, all of the API actions are rerouted to a local database. This lets you save on provisioned throughput, data storage, and data transfer fees.

For more information about DynamoDB Local and how to use it with the AWS CLI, see the following sections of the [Amazon DynamoDB Developer Guide](#):

- [DynamoDB Local](#)
- [Using the AWS CLI with DynamoDB Local](#)

Resources

AWS CLI reference:

- [aws dynamodb](#)
- [aws dynamodb create-table](#)
- [aws dynamodb put-item](#)
- [aws dynamodb query](#)

Service reference:

- [DynamoDB Local](#) in the Amazon DynamoDB Developer Guide
- [Using the AWS CLI with DynamoDB Local](#) in the Amazon DynamoDB Developer Guide

Use Amazon EC2 with the AWS CLI

An introduction to Amazon Elastic Compute Cloud

[Introduction to Amazon EC2 - Elastic Cloud Server and Hosting with AWS](#)

You can access the features of Amazon Elastic Compute Cloud (Amazon EC2) using the AWS Command Line Interface (AWS CLI). To list the AWS CLI commands for Amazon EC2, use the following command.

```
aws ec2 help
```

Before you run any commands, set your default credentials. For more information, see [Configure the AWS CLI](#).

This topic shows short-form examples of AWS CLI commands that perform common tasks for Amazon EC2.

For long-form examples of AWS CLI commands, see [AWS CLI code examples repository](#) on *GitHub*.

Topics

- [Create, display, and delete Amazon EC2 key pairs](#)
- [Create, configure, and delete security groups for Amazon EC2](#)
- [Launch, list, and terminate Amazon EC2 instances](#)
- [Change an Amazon EC2 instance type with a bash script](#)

Create, display, and delete Amazon EC2 key pairs

You can use the AWS Command Line Interface (AWS CLI) to create, display, and delete your key pairs for Amazon Elastic Compute Cloud (Amazon EC2). You use key pairs to connect to an Amazon EC2 instance.

You must provide the key pair to Amazon EC2 when you create the instance, and then use that key pair to authenticate when you connect to the instance.

Note

For additional command examples, see the [AWS CLI reference guide](#).

Topics

- [Prerequisites](#)
- [Create a key pair](#)
- [Display your key pair](#)
- [Delete your key pair](#)
- [References](#)

Prerequisites

To run the `ec2` commands, you need to:

- Install and configure the AWS CLI. For more information, see [the section called “Install/Update”](#) and [Authentication and access credentials](#).
- Set your IAM permissions to allow for Amazon EC2 access. For more information about IAM permissions for Amazon EC2, see [IAM policies for Amazon EC2](#) in the *Amazon EC2 User Guide*.

Create a key pair

To create a key pair, use the `aws ec2 create-key-pair` command with the `--query` option, and the `--output text` option to pipe your private key directly into a file.

```
$ aws ec2 create-key-pair --key-name MyKeyPair --query 'KeyMaterial' --output text  
> MyKeyPair.pem
```

For PowerShell, the `> file` redirection defaults to UTF-8 encoding, which cannot be used with some SSH clients. So, you must convert the output by piping it to the `out-file` command and explicitly set the encoding to `ascii`.

```
PS C:\>aws ec2 create-key-pair --key-name MyKeyPair --query 'KeyMaterial' --output text  
| out-file -encoding ascii -filepath MyKeyPair.pem
```

The resulting `MyKeyPair.pem` file looks similar to the following.

```
-----BEGIN RSA PRIVATE KEY-----
EXAMPLEKEYKCAQEAY7WZhaDsra1W3mRlQtvhwyORRX8gnxgDAfRt/gx42kWXsT4rXE/b5CpSgie/
vBoU7jLxx92pNHoFnByP+Dc21eyyz6CvjTmWA0JwfWiW5/akH7i05dSrvC7dQkW2duV5QuUdE0QW
Z/aNxMniGQE6XAgfwlnXVBwrerrQo+ZWQeqiUwwMkuEbLeJFLhMCvYURpUMSC1oehm449i1x9X1F
G50TCFe0zf18dqqCP6GzbPaIjIU19xX/az0R9V+tpU0zEL+wmXnZt3/nHPQ5xvD20JH67km6SuPW
oPzev/D8V+x4+bHthfSjR9Y7DvQFjfbVwHXigBdtZcU2/wei8D/HYwIDAQABAoIBAGZ1kaEvntrqu
/uler7vgIn5m7lN5LKw4hJLAIW6tUT/fzvtcCHK0SkbQCQXuriHmQ2MQyJX/0kn2NfjLV/ufGxbL1
mb5qwMGUnEpJaZD6QSSs3kICLwUUYUiGfc0uisbmJoap/GTLU0W5Mfcv36PaBUNy5p53V6G7hXb2
bahyWyJNfjLe4M86yd2YK3V2CmK+X/B0sShnJ36+hjrXPPWmV3N9zEmCdJjA+K15DYmhm/tJWSD9
81oGk9TopEp7CkIfatEATyyZiVqoRq6k64iuM9JkA30zdXzMqexXVJ1TLZVEH0E7bh1Y9d801ozR
oQs/FiZNAx2iijCWyv01pjE73+kCgYEA9mZtyhkHkFDpwrSM1APaL8oNAbbjwEy7Z5Mqfq1+lIp1
YkriL0DbLXlvRAH+yHPRit2hH0jtUNZh4Axv+cpG09qbUI3+43eEy24B7G/Uh+GTfbjsXs0xQx/x
p9otyVvc7hsQ5TA5PZb+mvkJ50BEKzet9XcKw0NBXYELGhnEPe7cCgYEA06Vgov6YHleHui9kHuws
ayav0elc5zkxjF9nfHFJRry21R1trw2Vdpn+9g481URrpzWV0Eihvm+xTtmaZ1Sp//1kq75XDwnU
WA8gkn603QE3fq2yN98BURsAKdJfJ5RL1HvGQvTe10HLYYXpJnEkHv+Unl2ajLivWUt5pbBrKbUC
gYBjb0+0Zk0sCcpZ29sbzjYjpIddErySIyRX5gV2uNQwAjLdp9PfN295yQ+BxMBXiIycWVQiw0bH
oMo7yykABY70zd5wQewBQ4AdS1WSX4nGDtsiFxiI5sKuAAe0CbTosy1s8w8fxoJ5Tz1sdoxNeGs
Arq6Wv/G16zQuAE9zK9vVwKBgF+09VI/1wJBirsDGz9whVwFFPrTkJNvJZzYt69qezx1sjgFKshy
WBhd4xHZtmCqpBP1AymEjr/T01bxyARMXmNIOWIANNXMGB4KGSy11mzSVAoQ+fqR+cJ3d0dyP1j
jjb0Ed/NY8fr1NDxAVHE8BSkdsx2f6ELEyBKJSRr9snRAoGAMrTwYneXzvTskF/S5Fyu0i0egLda
NWUH38v/nDCgEpIXD5Hn3qAEcju1Ijmbw1vtW+nY2jVhv7UGd8MjwUTNGItbdb6nsYqM2asinF3qS
VRkAKKKYeGjKpUfVTrW0YFjXkfcR/V+QFL50ndHAKJXjW7a4ejJLncTzmZSpYzwApc=
-----END RSA PRIVATE KEY-----
```

Your private key isn't stored in AWS and can be retrieved **only** when it's created. You can't recover it later. Instead, if you lose the private key, you must create a new key pair.

If you're connecting to your instance from a Linux computer, we recommend that you use the following command to set the permissions of your private key file so that only you can read it.

```
$ chmod 400 MyKeyPair.pem
```

Display your key pair

A "fingerprint" is generated from your key pair, and you can use it to verify that the private key that you have on your local machine matches the public key that's stored in AWS.

The fingerprint is an SHA1 hash taken from a DER-encoded copy of the private key. This value is captured when the key pair is created, and is stored in AWS with the public key. You can view the

fingerprint in the Amazon EC2 console or by running the AWS CLI command [aws ec2 describe-key-pairs](#).

The following example displays the fingerprint for MyKeyPair.

```
$ aws ec2 describe-key-pairs --key-name MyKeyPair
{
  "KeyPairs": [
    {
      "KeyName": "MyKeyPair",
      "KeyFingerprint":
        "1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f"
    }
  ]
}
```

For more information about keys and fingerprints, see [Amazon EC2 Key Pairs](#) in the *Amazon EC2 User Guide*.

Delete your key pair

To delete a key pair, run the [aws ec2 delete-key-pair](#) command, substituting *MyKeyPair* with the name of the pair to delete.

```
$ aws ec2 delete-key-pair --key-name MyKeyPair
```

References

AWS CLI reference:

- [aws ec2](#)
- [aws ec2 create-key-pair](#)
- [aws ec2 delete-key-pair](#)
- [aws ec2 describe-key-pairs](#)

Other reference:

- [Amazon Elastic Compute Cloud Documentation](#)
- To view and contribute to AWS SDK and AWS CLI code examples, see the [AWS Code Examples Repository](#) on *GitHub*.

Create, configure, and delete security groups for Amazon EC2

You can create a security group for your Amazon Elastic Compute Cloud (Amazon EC2) instances that essentially operates as a firewall, with rules that determine what network traffic can enter and leave.

Use the AWS Command Line Interface (AWS CLI) to create a security group, add rules to existing security groups, and delete security groups.

Note

For additional command examples, see the [AWS CLI reference guide](#).

Topics

- [Prerequisites](#)
- [Create a security group](#)
- [Add rules to your security group](#)
- [Delete your security group](#)
- [References](#)

Prerequisites

To run the `ec2` commands, you need to:

- Install and configure the AWS CLI. For more information, see [the section called “Install/Update”](#) and [Authentication and access credentials](#).
- Set your IAM permissions to allow for Amazon EC2 access. For more information about IAM permissions for Amazon EC2, see [IAM policies for Amazon EC2](#) in the *Amazon EC2 User Guide*.

Create a security group

You can create security groups associated with virtual private clouds (VPCs) .

The following [aws ec2 create-security-group](#) example shows how to create a security group for a specified VPC.

```
$ aws ec2 create-security-group --group-name my-sg --description "My security group" --  
vpc-id vpc-1a2b3c4d  
{  
  "GroupId": "sg-903004f8"  
}
```

To view the initial information for a security group, run the [aws ec2 describe-security-groups](#) command. You can reference an EC2-VPC security group only by its `vpc-id`, not its name.

```
$ aws ec2 describe-security-groups --group-ids sg-903004f8  
{  
  "SecurityGroups": [  
    {  
      "IpPermissionsEgress": [  
        {  
          "IpProtocol": "-1",  
          "IpRanges": [  
            {  
              "CidrIp": "0.0.0.0/0"  
            }  
          ],  
          "UserIdGroupPairs": []  
        }  
      ],  
      "Description": "My security group"  
      "IpPermissions": [],  
      "GroupName": "my-sg",  
      "VpcId": "vpc-1a2b3c4d",  
      "OwnerId": "123456789012",  
      "GroupId": "sg-903004f8"  
    }  
  ]  
}
```

Add rules to your security group

When you run an Amazon EC2 instance, you must enable rules in the security group to allow incoming network traffic for your means of connecting to the image.

For example, if you're launching a Windows instance, you typically add a rule to allow inbound traffic on TCP port 3389 to support Remote Desktop Protocol (RDP). If you're launching a

Linux instance, you typically add a rule to allow inbound traffic on TCP port 22 to support SSH connections.

Use the [aws ec2 authorize-security-group-ingress](#) command to add a rule to your security group. A required parameter of this command is the public IP address of your computer, or the network (in the form of an address range) that your computer is attached to, in [CIDR](#) notation.

Note

We provide the following service, <https://checkip.amazonaws.com/>, to enable you to determine your public IP address. To find other services that can help you identify your IP address, use your browser to search for "what is my IP address". If you connect through an ISP or from behind your firewall using a dynamic IP address (through a NAT gateway from a private network), your address can change periodically. In that case, you must find out the range of IP addresses used by client computers.

The following example shows how to add a rule for RDP (TCP port 3389) to an EC2-VPC security group with the ID `sg-903004f8` using your IP address.

To start, find your IP address.

```
$ curl https://checkip.amazonaws.com
x.x.x.x
```

You can then add the IP address to your security group by running the [aws ec2 authorize-security-group-ingress](#) command.

```
$ aws ec2 authorize-security-group-ingress --group-id sg-903004f8 --protocol tcp --port 3389 --cidr x.x.x.x/x
```

The following command adds another rule to enable SSH to instances in the same security group.

```
$ aws ec2 authorize-security-group-ingress --group-id sg-903004f8 --protocol tcp --port 22 --cidr x.x.x.x/x
```

To view the changes to the security group, run the [aws ec2 describe-security-groups](#) command.

```
$ aws ec2 describe-security-groups --group-ids sg-903004f8
```

```
{
  "SecurityGroups": [
    {
      "IpPermissionsEgress": [
        {
          "IpProtocol": "-1",
          "IpRanges": [
            {
              "CidrIp": "0.0.0.0/0"
            }
          ],
          "UserIdGroupPairs": []
        }
      ],
      "Description": "My security group"
      "IpPermissions": [
        {
          "ToPort": 22,
          "IpProtocol": "tcp",
          "IpRanges": [
            {
              "CidrIp": "x.x.x.x/x"
            }
          ],
          "UserIdGroupPairs": [],
          "FromPort": 22
        }
      ],
      "GroupName": "my-sg",
      "OwnerId": "123456789012",
      "GroupId": "sg-903004f8"
    }
  ]
}
```

Delete your security group

To delete a security group, run the [aws ec2 delete-security-group](#) command.

Note

You can't delete a security group if it's currently attached to an environment.

The following command example deletes an EC2-VPC security group.

```
$ aws ec2 delete-security-group --group-id sg-903004f8
```

References

AWS CLI reference:

- [aws ec2](#)
- [aws ec2 authorize-security-group-ingress](#)
- [aws ec2 create-security-group](#)
- [aws ec2 delete-security-group](#)
- [aws ec2 describe-security-groups](#)

Other reference:

- [Amazon Elastic Compute Cloud Documentation](#)
- To view and contribute to AWS SDK and AWS CLI code examples, see the [AWS Code Examples Repository](#) on *GitHub*.

Launch, list, and terminate Amazon EC2 instances

You can use the AWS Command Line Interface (AWS CLI) to launch, list, and terminate Amazon Elastic Compute Cloud (Amazon EC2) instances. If you launch an instance that isn't within the AWS Free Tier, you are billed after you launch the instance and charged for the time that the instance is running, even if it remains idle.

Note

For additional command examples, see the [AWS CLI reference guide](#).

Topics

- [Prerequisites](#)
- [Launch your instance](#)
- [Add a block device to your instance](#)

- [Add a tag to your instance](#)
- [Connect to your instance](#)
- [List your instances](#)
- [Terminate your instance](#)
- [References](#)

Prerequisites

To run the `ec2` commands in this topic, you need to:

- Install and configure the AWS CLI. For more information, see [the section called “Install/Update”](#) and [Authentication and access credentials](#).
- Set your IAM permissions to allow for Amazon EC2 access. For more information about IAM permissions for Amazon EC2, see [IAM policies for Amazon EC2](#) in the *Amazon EC2 User Guide*.
- Create a [key pair](#) and a [security group](#).
- Select an Amazon Machine Image (AMI) and note the AMI ID. For more information, see [Finding a Suitable AMI](#) in the *Amazon EC2 User Guide*.

Launch your instance

To launch an Amazon EC2 instance using the AMI you selected, use the `aws ec2 run-instances` command. You can launch the instance into a virtual private cloud (VPC).

Initially, your instance appears in the pending state, but changes to the running state after a few minutes.

The following example shows how to launch a `t2.micro` instance in the specified subnet of a VPC. Replace the *italicized* parameter values with your own.

```
$ aws ec2 run-instances --image-id ami-xxxxxxx --count 1 --instance-type t2.micro --
key-name MyKeyPair --security-group-ids sg-903004f8 --subnet-id subnet-6e7f829e
{
  "OwnerId": "123456789012",
  "ReservationId": "r-5875ca20",
  "Groups": [
    {
      "GroupName": "my-sg",
      "GroupId": "sg-903004f8"
```

```
    }
  ],
  "Instances": [
    {
      "Monitoring": {
        "State": "disabled"
      },
      "PublicDnsName": null,
      "Platform": "windows",
      "State": {
        "Code": 0,
        "Name": "pending"
      },
      "EbsOptimized": false,
      "LaunchTime": "2013-07-19T02:42:39.000Z",
      "PrivateIpAddress": "10.0.1.114",
      "ProductCodes": [],
      "VpcId": "vpc-1a2b3c4d",
      "InstanceId": "i-5203422c",
      "ImageId": "ami-173d747e",
      "PrivateDnsName": "ip-10-0-1-114.ec2.internal",
      "KeyName": "MyKeyPair",
      "SecurityGroups": [
        {
          "GroupName": "my-sg",
          "GroupId": "sg-903004f8"
        }
      ],
      "ClientToken": null,
      "SubnetId": "subnet-6e7f829e",
      "InstanceType": "t2.micro",
      "NetworkInterfaces": [
        {
          "Status": "in-use",
          "SourceDestCheck": true,
          "VpcId": "vpc-1a2b3c4d",
          "Description": "Primary network interface",
          "NetworkInterfaceId": "eni-a7edb1c9",
          "PrivateIpAddresses": [
            {
              "PrivateDnsName": "ip-10-0-1-114.ec2.internal",
              "Primary": true,
              "PrivateIpAddress": "10.0.1.114"
            }
          ]
        }
      ]
    }
  ]
}
```

```
    ],
    "PrivateDnsName": "ip-10-0-1-114.ec2.internal",
    "Attachment": {
      "Status": "attached",
      "DeviceIndex": 0,
      "DeleteOnTermination": true,
      "AttachmentId": "eni-attach-52193138",
      "AttachTime": "2013-07-19T02:42:39.000Z"
    },
    "Groups": [
      {
        "GroupName": "my-sg",
        "GroupId": "sg-903004f8"
      }
    ],
    "SubnetId": "subnet-6e7f829e",
    "OwnerId": "123456789012",
    "PrivateIpAddress": "10.0.1.114"
  }
],
"SourceDestCheck": true,
"Placement": {
  "Tenancy": "default",
  "GroupName": null,
  "AvailabilityZone": "us-west-2b"
},
"Hypervisor": "xen",
"BlockDeviceMappings": [
  {
    "DeviceName": "/dev/sda1",
    "Ebs": {
      "Status": "attached",
      "DeleteOnTermination": true,
      "VolumeId": "vol-877166c8",
      "AttachTime": "2013-07-19T02:42:39.000Z"
    }
  }
],
"Architecture": "x86_64",
"StateReason": {
  "Message": "pending",
  "Code": "pending"
},
"RootDeviceName": "/dev/sda1",
```

```

        "VirtualizationType": "hvm",
        "RootDeviceType": "ebs",
        "Tags": [
            {
                "Value": "MyInstance",
                "Key": "Name"
            }
        ],
        "AmiLaunchIndex": 0
    }
]
}

```

Add a block device to your instance

Each instance that you launch has an associated root device volume. You can use block device mapping to specify additional Amazon Elastic Block Store (Amazon EBS) volumes or instance store volumes to attach to an instance when it's launched.

To add a block device to your instance, specify the `--block-device-mappings` option when you use `run-instances`.

The following example parameter provisions a standard Amazon EBS volume that is 20 GB in size, and maps it to your instance using the identifier `/dev/sdf`.

```
--block-device-mappings "[{\"DeviceName\": \"/dev/sdf\", \"Ebs\": {\"VolumeSize\": 20, \"DeleteOnTermination\": false}}]"
```

The following example adds an Amazon EBS volume, mapped to `/dev/sdf`, based on an existing snapshot. A snapshot represents an image that is loaded onto the volume for you. When you specify a snapshot, you don't have to specify a volume size; it will be large enough to hold your image. However, if you do specify a size, it must be greater than or equal to the size of the snapshot.

```
--block-device-mappings "[{\"DeviceName\": \"/dev/sdf\", \"Ebs\": {\"SnapshotId\": \"snap-a1b2c3d4\"}}]"
```

The following example adds two volumes to your instance. The number of volumes available to your instance depends on its instance type.

```
--block-device-mappings "[{\"DeviceName\":\"/dev/sdf\",\"VirtualName\":\"ephemeral0\"},  
{\"DeviceName\":\"/dev/sdg\",\"VirtualName\":\"ephemeral1\"}]"
```

The following example creates the mapping (/dev/sdj), but doesn't provision a volume for the instance.

```
--block-device-mappings "[{\"DeviceName\":\"/dev/sdj\",\"NoDevice\":\"\"}]"
```

For more information, see [Block Device Mapping](#) in the *Amazon EC2 User Guide*.

Add a tag to your instance

A tag is a label that you assign to an AWS resource. It enables you to add metadata to your resources that you can use for a variety of purposes. For more information, see [Tagging Your Resources](#) in the *Amazon EC2 User Guide*.

The following example shows how to add a tag with the key name "Name" and the value "MyInstance" to the specified instance, by using the [aws ec2 create-tags](#) command.

```
$ aws ec2 create-tags --resources i-5203422c --tags Key=Name,Value=MyInstance
```

Connect to your instance

When your instance is running, you can connect to it and use it just as you'd use a computer sitting in front of you. For more information, see [Connect to Your Amazon EC2 Instance](#) in the *Amazon EC2 User Guide*.

List your instances

You can use the AWS CLI to list your instances and view information about them. You can list all your instances, or filter the results based on the instances that you're interested in.

The following examples show how to use the [aws ec2 describe-instances](#) command.

The following command lists all your instances.

```
$ aws ec2 describe-instances
```

The following command filters the list to only your `t2.micro` instances and outputs only the `InstanceId` values for each match.

```
$ aws ec2 describe-instances --filters "Name=instance-type,Values=t2.micro" --query
"Reservations[].Instances[].InstanceId"
[
  "i-05e998023d9c69f9a"
]
```

The following command lists any of your instances that have the tag Name=MyInstance.

```
$ aws ec2 describe-instances --filters "Name=tag:Name,Values=MyInstance"
```

The following command lists your instances that were launched using any of the following AMIs: ami-x0123456, ami-y0123456, and ami-z0123456.

```
$ aws ec2 describe-instances --filters "Name=image-id,Values=ami-x0123456,ami-
y0123456,ami-z0123456"
```

Terminate your instance

Terminating an instance deletes it. You can't reconnect to an instance after you've terminated it.

As soon as the state of the instance changes to shutting-down or terminated, you stop incurring charges for that instance. If you want to reconnect to an instance later, use [stop-instances](#) instead of `terminate-instances`. For more information, see [Terminate Your Instance](#) in the *Amazon EC2 User Guide*.

To delete an instance, you use the command [aws ec2 terminate-instances](#) to delete it.

```
$ aws ec2 terminate-instances --instance-ids i-5203422c
{
  "TerminatingInstances": [
    {
      "InstanceId": "i-5203422c",
      "CurrentState": {
        "Code": 32,
        "Name": "shutting-down"
      },
      "PreviousState": {
        "Code": 16,
        "Name": "running"
      }
    }
  ]
}
```

```
    }  
  ]  
}
```

References

AWS CLI reference:

- [aws ec2](#)
- [aws ec2 create-tags](#)
- [aws ec2 describe-instances](#)
- [aws ec2 run-instances](#)
- [aws ec2 terminate-instances](#)

Other reference:

- [Amazon Elastic Compute Cloud Documentation](#)
- To view and contribute to AWS SDK and AWS CLI code examples, see the [AWS Code Examples Repository](#) on *GitHub*.

Change an Amazon EC2 instance type with a bash script

This bash scripting example for Amazon EC2 changes the instance type for an Amazon EC2 instance using the AWS Command Line Interface (AWS CLI). It stops the instance if it's running, changes the instance type, and then, if requested, restarts the instance. Shell scripts are programs designed to run in a command line interface.

Note

For additional command examples, see the [AWS CLI reference guide](#).

Topics

- [Before you start](#)
- [About this example](#)
- [Parameters](#)

- [Files](#)
- [References](#)

Before you start

Before you can run any of the below examples, the following things need to be completed.

- Install and configure the AWS CLI. For more information, see [the section called “Install/Update” and Authentication and access credentials](#).
- The profile that you use must have permissions that allow the AWS operations performed by the examples.
- A running Amazon EC2 instance in the account for which you have permission to stop and modify. If you run the test script, it launches an instance for you, tests changing the type, and then terminates the instance.
- As an AWS best practice, grant this code least privilege, or only the permissions required to perform a task. For more information, see [Grant Least Privilege](#) in the *AWS Identity and Access Management (IAM) User Guide*.
- This code has not been tested in all AWS Regions. Some AWS services are available only in specific Regions. For more information, see [Service Endpoints and Quotas](#) in the *AWS General Reference Guide*.
- Running this code can result in charges to your AWS account. It is your responsibility to ensure that any resources created by this script are removed when you are done with them.

About this example

This example is written as a function in the shell script file `change_ec2_instance_type.sh` that you can `source` from another script or from the command line. Each script file contains comments describing each of the functions. Once the function is in memory, you can invoke it from the command line. For example, the following commands change the type of the specified instance to `t2.nano`:

```
$ source ./change_ec2_instance_type.sh
$ ./change_ec2_instance_type -i *instance-id* -t new-type
```

For the full example and downloadable script files, see [Change Amazon EC2 Instance Type](#) in the *AWS Code Examples Repository on GitHub*.

Parameters

-i - (*string*) Specifies the instance ID to modify.

-t - (*string*) Specifies the Amazon EC2 instance type to switch to.

-r - (*switch*) By default, this is unset. If **-r** is set, restarts the instance after the type switch.

-f - (*switch*) By default, the script prompts the user to confirm shutting down the instance before making the switch. If **-f** is set, the function doesn't prompt the user before shutting down the instance to make the type switch

-v - (*switch*) By default, the script operates silently and displays output only in the event of an error. If **-v** is set, the function displays status throughout its operation.

Files

`change_ec2_instance_type.sh`

The main script file contains the `change_ec2_instance_type()` function that performs the following tasks:

- Verifies that the specified Amazon EC2 instance exists.
- Unless **-f** is selected, warns the user before stopping the instance.
- Changes the instance type
- If you set **-r**, restarts the instance and confirms that the instance is running

View the code for [change_ec2_instance_type.sh](#) on *GitHub*.

`test_change_ec2_instance_type.sh`

The file `test_change_ec2_instance_type.sh` script tests the various code paths for the `change_ec2_instance_type` function. If all steps in the test script work correctly, the test script removes all resources that it created.

You can run the test script with the following parameters:

- **-v** - (*switch*) The each test shows a pass/failure status as they run. By default, the tests runs silently and the output includes only the final overall pass/failure status.
- **-i** - (*switch*) The script pauses after each test to enable you to browse the intermediate results of each step. Enables you to examine the current status of the instance using the Amazon EC2 console. The script proceeds to the next step after you press *ENTER* at the prompt.

View the code for [test_change_ec2_instance_type.sh](#) on *GitHub*.

awsdocs_general.sh

The script file `awsdocs_general.sh` holds general purpose functions used across advanced examples for the AWS CLI.

View the code for [awsdocs_general.sh](#) on *GitHub*.

References

AWS CLI reference:

- [aws ec2](#)
- [aws ec2 describe-instances](#)
- [aws ec2 modify-instance-attribute](#)
- [aws ec2 start-instances](#)
- [aws ec2 stop-instances](#)
- [aws ec2 wait instance-running](#)
- [aws ec2 wait instance-stopped](#)

Other reference:

- [Amazon Elastic Compute Cloud Documentation](#)
- To view and contribute to AWS SDK and AWS CLI code examples, see the [AWS Code Examples Repository](#) on *GitHub*.

Use Amazon S3 Glacier with the AWS CLI

An introduction to Amazon S3 Glacier

[Introduction to Amazon S3 Glacier](#)

This topic shows examples of AWS CLI commands that perform common tasks for S3 Glacier. The examples demonstrate how to use the AWS CLI to upload a large file to S3 Glacier by splitting it into smaller parts and uploading them from the command line.

You can access Amazon S3 Glacier features using the AWS Command Line Interface (AWS CLI). To list the AWS CLI commands for S3 Glacier, use the following command.

```
aws glacier help
```

Note

For command reference and additional examples, see [aws glacier](#) in the *AWS CLI Command Reference*.

Topics

- [Prerequisites](#)
- [Create an Amazon S3 Glacier vault](#)
- [Prepare a file for uploading](#)
- [Initiate a multipart upload and upload files](#)
- [Complete the upload](#)
- [Resources](#)

Prerequisites

To run the `glacier` commands, you need to:

- Install and configure the AWS CLI. For more information, see [the section called “Install/Update”](#) and [Authentication and access credentials](#).
- This tutorial uses several command line tools that typically come preinstalled on Unix-like operating systems, including Linux and macOS. Windows users can use the same tools by installing [Cygwin](#) and running the commands from the Cygwin terminal. We note Windows native commands and utilities that perform the same functions where available.

Create an Amazon S3 Glacier vault

Create a vault with the [create-vault](#) command.

```
$ aws glacier create-vault --account-id - --vault-name myvault
{
  "location": "/123456789012/vaults/myvault"
}
```

Note

All S3 Glacier commands require an account ID parameter. Use the hyphen character (`--account-id -`) to use the current account.

Prepare a file for uploading

Create a file for the test upload. The following commands create a file named *largefile* that contains exactly 3 MiB of random data.

Linux or macOS

```
$ dd if=/dev/urandom of=largefile bs=3145728 count=1
1+0 records in
1+0 records out
3145728 bytes (3.1 MB) copied, 0.205813 s, 15.3 MB/s
```

`dd` is a utility that copies a number of bytes from an input file to an output file. The previous example uses the system device file `/dev/urandom` as a source of random data. `fsutil` performs a similar function in Windows.

Windows

```
C:\> fsutil file createnew largefile 3145728
File C:\temp\largefile is created
```

Next, split the file into 1 MiB (1,048,576 byte) chunks.

```
$ split -b 1048576 --verbose largefile chunk
```

```
creating file `chunkaa`  
creating file `chunkab`  
creating file `chunkac`
```

Note

[HJ-Split](#) is a free file splitter for Windows and many other platforms.

Initiate a multipart upload and upload files

Create a multipart upload in Amazon S3 Glacier by using the [initiate-multipart-upload](#) command.

```
$ aws glacier initiate-multipart-upload --account-id - --archive-description "multipart  
upload test" --part-size 1048576 --vault-name myvault  
{  
  "uploadId": "19gaRezEXAMPLES6Ry5YYdqthHOC_kGRCT03L9yetr220UmPtBYKk-  
0ssZtLqyFu7sY1_1R7vgFuJV6NtcV5zpsJ",  
  "location": "/123456789012/vaults/myvault/multipart-  
uploads/19gaRezEXAMPLES6Ry5YYdqthHOC_kGRCT03L9yetr220UmPtBYKk-  
0ssZtLqyFu7sY1_1R7vgFuJV6NtcV5zpsJ"  
}
```

S3 Glacier requires the size of each part in bytes (1 MiB in this example), your vault name, and an account ID to configure the multipart upload. The AWS CLI outputs an upload ID when the operation is complete. Save the upload ID to a shell variable for later use.

Linux or macOS

```
$ UPLOADID="19gaRezEXAMPLES6Ry5YYdqthHOC_kGRCT03L9yetr220UmPtBYKk-  
0ssZtLqyFu7sY1_1R7vgFuJV6NtcV5zpsJ"
```

Windows

```
C:\> set UPLOADID="19gaRezEXAMPLES6Ry5YYdqthHOC_kGRCT03L9yetr220UmPtBYKk-  
0ssZtLqyFu7sY1_1R7vgFuJV6NtcV5zpsJ"
```

Next, use the [upload-multipart-part](#) command to upload each of the three parts.

```
$ aws glacier upload-multipart-part --upload-id $UPLOADID --body chunkaa --range 'bytes
0-1048575/*' --account-id - --vault-name myvault
{
  "checksum": "e1f2a7cd6e047fa606fe2f0280350f69b9f8cfa602097a9a026360a7edc1f553"
}
$ aws glacier upload-multipart-part --upload-id $UPLOADID --body chunkab --range 'bytes
1048576-2097151/*' --account-id - --vault-name myvault
{
  "checksum": "e1f2a7cd6e047fa606fe2f0280350f69b9f8cfa602097a9a026360a7edc1f553"
}
$ aws glacier upload-multipart-part --upload-id $UPLOADID --body chunkac --range 'bytes
2097152-3145727/*' --account-id - --vault-name myvault
{
  "checksum": "e1f2a7cd6e047fa606fe2f0280350f69b9f8cfa602097a9a026360a7edc1f553"
}
```

Note

The previous example uses the dollar sign (\$) to reference the contents of the UPLOADID shell variable on Linux. On the Windows command line, use a percent sign (%) on either side of the variable name (for example, %UPLOADID%).

You must specify the byte range of each part when you upload it so that S3 Glacier can reassemble it in the correct order. Each piece is 1,048,576 bytes, so the first piece occupies bytes 0-1048575, the second 1048576-2097151, and the third 2097152-3145727.

Complete the upload

Amazon S3 Glacier requires a tree hash of the original file to confirm that all of the uploaded pieces reached AWS intact.

To calculate a tree hash, you must split the file into 1 MiB parts and calculate a binary SHA-256 hash of each piece. Then you split the list of hashes into pairs, combine the two binary hashes in each pair, and take hashes of the results. Repeat this process until there is only one hash left. If there is an odd number of hashes at any level, promote it to the next level without modifying it.

The key to calculating a tree hash correctly when using command line utilities is to store each hash in binary format and convert to hexadecimal only at the last step. Combining or hashing the hexadecimal version of any hash in the tree will cause an incorrect result.

Note

Windows users can use the `type` command in place of `cat`. OpenSSL is available for Windows at [OpenSSL.org](https://www.openssl.org).

To calculate a tree hash

1. If you haven't already, split the original file into 1 MiB parts.

```
$ split --bytes=1048576 --verbose largefile chunk
creating file `chunkaa'
creating file `chunkab'
creating file `chunkac'
```

2. Calculate and store the binary SHA-256 hash of each chunk.

```
$ openssl dgst -sha256 -binary chunkaa > hash1
$ openssl dgst -sha256 -binary chunkab > hash2
$ openssl dgst -sha256 -binary chunkac > hash3
```

3. Combine the first two hashes and take the binary hash of the result.

```
$ cat hash1 hash2 > hash12
$ openssl dgst -sha256 -binary hash12 > hash12hash
```

4. Combine the parent hash of chunks aa and ab with the hash of chunk ac and hash the result, this time outputting hexadecimal. Store the result in a shell variable.

```
$ cat hash12hash hash3 > hash123
$ openssl dgst -sha256 hash123
SHA256(hash123)= 9628195fcdcbbbe76cdde932d4646fa7de5f219fb39823836d81f0cc0e18aa67
$ TREEHASH=9628195fcdcbbbe76cdde932d4646fa7de5f219fb39823836d81f0cc0e18aa67
```

Finally, complete the upload with the [complete-multipart-upload](#) command. This command takes the original file's size in bytes, the final tree hash value in hexadecimal, and your account ID and vault name.

```
$ aws glacier complete-multipart-upload --checksum $TREEHASH --archive-size 3145728 --
upload-id $UPLOADID --account-id - --vault-name myvault
```



```
{
  "archiveId": "d3AbWhE0YE1m6f_fI1jPG82F8xzbMEEZmrALLGAA0NJAzo5QdP-
N83MKqd96Unspoa5H51ItWX-sK8-QS0ZhwsyGiu9-R-
kWUyS1dSBlmgPPWkEbeFfqDSav053rU7FvVLHfRc6hg",
  "checksum": "9628195fcdbcbbe76cdde932d4646fa7de5f219fb39823836d81f0cc0e18aa67",
  "location": "/123456789012/vaults/myvault/archives/
d3AbWhE0YE1m6f_fI1jPG82F8xzbMEEZmrALLGAA0NJAzo5QdP-N83MKqd96Unspoa5H51ItWX-sK8-
QS0ZhwsyGiu9-R-kWUyS1dSBlmgPPWkEbeFfqDSav053rU7FvVLHfRc6hg"
}
```

You can also check the status of the vault using the [describe-vault](#) command.

```
$ aws glacier describe-vault --account-id - --vault-name myvault
{
  "SizeInBytes": 3178496,
  "VaultARN": "arn:aws:glacier:us-west-2:123456789012:vaults/myvault",
  "LastInventoryDate": "2018-12-07T00:26:19.028Z",
  "NumberOfArchives": 1,
  "CreationDate": "2018-12-06T21:23:45.708Z",
  "VaultName": "myvault"
}
```

Note

Vault status is updated about once per day. See [Working with Vaults](#) for more information.

Now it's safe to remove the chunk and hash files that you created.

```
$ rm chunk* hash*
```

For more information on multipart uploads, see [Uploading Large Archives in Parts](#) and [Computing Checksums](#) in the *Amazon S3 Glacier Developer Guide*.

Resources

AWS CLI reference:

- [aws glacier](#)
- [aws glacier complete-multipart-upload](#)

- [aws glacier create-vault](#)
- [aws glacier describe-vault](#)
- [aws glacier initiate-multipart-upload](#)

Service reference:

- [Amazon S3 Glacier Developer Guide](#)
- [Uploading Large Archives in Parts](#) in the *Amazon S3 Glacier Developer Guide*
- [Computing Checksums](#) in the *Amazon S3 Glacier Developer Guide*
- [Working with Vaults](#) in the *Amazon S3 Glacier Developer Guide*

Use AWS Identity and Access Management from the AWS CLI

An introduction to AWS Identity and Access Management

[Introduction to AWS Identity and Access Management](#)

You can access the features of AWS Identity and Access Management (IAM) using the AWS Command Line Interface (AWS CLI). To list the AWS CLI commands for IAM, use the following command.

```
aws iam help
```

This topic shows examples of AWS CLI commands that perform common tasks for IAM.

Before you run any commands, set your default credentials. For more information, see [Configure the AWS CLI](#).

For more information on the IAM service, see the [AWS Identity and Access Management User Guide](#).

Topics

- [Create IAM users and groups](#)
- [Attach an IAM managed policy to a user](#)
- [Set an initial password for an IAM user](#)

- [Create an access key for an IAM user](#)

Create IAM users and groups

This topic describes how to use AWS Command Line Interface (AWS CLI) commands to create an AWS Identity and Access Management (IAM) group and a new user, and then add the user to the group. For more information on the IAM service, see the [AWS Identity and Access Management User Guide](#).

Before you run any commands, set your default credentials. For more information, see [Configure the AWS CLI](#).

To create a group and add a new user to it

1. Use the [create-group](#) command to create the group.

```
$ aws iam create-group --group-name MyIamGroup
{
  "Group": {
    "GroupName": "MyIamGroup",
    "CreateDate": "2018-12-14T03:03:52.834Z",
    "GroupId": "AGPAJNUJ2W4IJVEXAMPLE",
    "Arn": "arn:aws:iam::123456789012:group/MyIamGroup",
    "Path": "/"
  }
}
```

2. Use the [create-user](#) command to create the user.

```
$ aws iam create-user --user-name MyUser
{
  "User": {
    "UserName": "MyUser",
    "Path": "/",
    "CreateDate": "2018-12-14T03:13:02.581Z",
    "UserId": "AIDAJY2PE5XUZ4EXAMPLE",
    "Arn": "arn:aws:iam::123456789012:user/MyUser"
  }
}
```

3. Use the [add-user-to-group](#) command to add the user to the group.

```
$ aws iam add-user-to-group --user-name MyUser --group-name MyIamGroup
```

4. To verify that the MyIamGroup group contains the MyUser, use the [get-group](#) command.

```
$ aws iam get-group --group-name MyIamGroup
{
  "Group": {
    "GroupName": "MyIamGroup",
    "CreateDate": "2018-12-14T03:03:52Z",
    "GroupId": "AGPAJNUJ2W4IJVEXAMPLE",
    "Arn": "arn:aws:iam::123456789012:group/MyIamGroup",
    "Path": "/"
  },
  "Users": [
    {
      "UserName": "MyUser",
      "Path": "/",
      "CreateDate": "2018-12-14T03:13:02Z",
      "UserId": "AIDAJY2PE5XUZ4EXAMPLE",
      "Arn": "arn:aws:iam::123456789012:user/MyUser"
    }
  ],
  "IsTruncated": "false"
}
```

Attach an IAM managed policy to a user

This topic describes how to use AWS Command Line Interface (AWS CLI) commands to attach an AWS Identity and Access Management (IAM) policy to a user. The policy in this example provides the user with "Power User Access". For more information on the IAM service, see the [AWS Identity and Access Management User Guide](#).

Before you run any commands, set your default credentials. For more information, see [Configure the AWS CLI](#).

To attach an IAM managed policy to a user

1. Determine the Amazon Resource Name (ARN) of the policy to attach. The following command uses `list-policies` to find the ARN of the policy with the name `PowerUserAccess`. It then stores that ARN in an environment variable.

```
$ export POLICYARN=$(aws iam list-policies --query 'Policies[?
PolicyName==`PowerUserAccess`].{ARN:Arn}' --output text) ~
$ echo $POLICYARN
arn:aws:iam::aws:policy/PowerUserAccess
```

2. To attach the policy, use the [attach-user-policy](#) command, and reference the environment variable that holds the policy ARN.

```
$ aws iam attach-user-policy --user-name MyUser --policy-arn $POLICYARN
```

3. Verify that the policy is attached to the user by running the [list-attached-user-policies](#) command.

```
$ aws iam list-attached-user-policies --user-name MyUser
{
  "AttachedPolicies": [
    {
      "PolicyName": "PowerUserAccess",
      "PolicyArn": "arn:aws:iam::aws:policy/PowerUserAccess"
    }
  ]
}
```

For more information, see [Access Management Resources](#). This topic provides links to an overview of permissions and policies, and links to examples of policies for accessing Amazon S3, Amazon EC2, and other services.

Set an initial password for an IAM user

This topic describes how to use AWS Command Line Interface (AWS CLI) commands to set an initial password for an AWS Identity and Access Management (IAM) user. For more information on the IAM service, see the [AWS Identity and Access Management User Guide](#).

Before you run any commands, set your default credentials. For more information, see [Configure the AWS CLI](#).

The following command uses [create-login-profile](#) to set an initial password on the specified user. When the user signs in for the first time, the user is required to change the password to something that only the user knows.

```
$ aws iam create-login-profile --user-name MyUser --password My!User1Login8P@ssword --password-reset-required
{
  "LoginProfile": {
    "UserName": "MyUser",
    "CreateDate": "2018-12-14T17:27:18Z",
    "PasswordResetRequired": true
  }
}
```

You can use the `update-login-profile` command to *change* the password for a user.

```
$ aws iam update-login-profile --user-name MyUser --password My!User1ADifferentP@ssword
```

Create an access key for an IAM user

This topic describes how to use AWS Command Line Interface (AWS CLI) commands to create a set of access keys for an AWS Identity and Access Management (IAM) user. For more information on the IAM service, see the [AWS Identity and Access Management User Guide](#).

Before you run any commands, set your default credentials. For more information, see [Configure the AWS CLI](#).

You can use the [create-access-key](#) command to create an access key for a user. An access key is a set of security credentials that consists of an access key ID and a secret key.

A user can create only two access keys at one time. If you try to create a third set, the command returns a `LimitExceeded` error.

```
$ aws iam create-access-key --user-name MyUser
{
  "AccessKey": {
    "UserName": "MyUser",
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "Status": "Active",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "CreateDate": "2018-12-14T17:34:16Z"
  }
}
```

Use the [delete-access-key](#) command to delete an access key for a user. Specify which access key to delete by using the access key ID.

```
$ aws iam delete-access-key --user-name MyUser --access-key-id AKIAIOSFODNN7EXAMPLE
```

Use Amazon S3 with the AWS CLI

An introduction to Amazon Simple Storage Service (Amazon S3)

[Introduction to Amazon Simple Storage Service \(Amazon S3 - Cloud Storage on AWS\)](#)

You can access the features of Amazon Simple Storage Service (Amazon S3) using the AWS Command Line Interface (AWS CLI). The AWS CLI provides two tiers of commands for accessing Amazon S3:

- **s3** – High-level commands that simplify performing common tasks, such as creating, manipulating, and deleting objects and buckets.
- **s3api** – Exposes direct access to all Amazon S3 API operations which enables you to carry out advanced operations.

Topics in this guide:

- [Use high-level \(s3\) commands with the AWS CLI](#)
- [Use API-Level \(s3api\) commands with the AWS CLI](#)
- [Amazon S3 bucket lifecycle operations scripting example](#)

Use high-level (s3) commands with the AWS CLI

This topic describes some of the commands you can use to manage Amazon S3 buckets and objects using the [aws s3](#) commands in the AWS CLI. For commands not covered in this topic and additional command examples, see the [aws s3](#) commands in the *AWS CLI Reference*.

The high-level `aws s3` commands simplify managing Amazon S3 objects. These commands enable you to manage the contents of Amazon S3 within itself and with local directories.

Topics

- [Prerequisites](#)
- [Before you start](#)
- [Create a bucket](#)
- [List buckets and objects](#)
- [Delete buckets](#)
- [Delete objects](#)
- [Move objects](#)
- [Copy objects](#)
- [Sync objects](#)
- [Frequently used options for s3 commands](#)
- [Resources](#)

Prerequisites

To run the s3 commands, you need to:

- Install and configure the AWS CLI. For more information, see [the section called “Install/Update” and Authentication and access credentials](#).
- The profile that you use must have permissions that allow the AWS operations performed by the examples.
- Understand these Amazon S3 terms:
 - **Bucket** – A top-level Amazon S3 folder.
 - **Prefix** – An Amazon S3 folder in a bucket.
 - **Object** – Any item that's hosted in an Amazon S3 bucket.

Before you start

This section describes a few things to note before you use `aws s3` commands.

Large object uploads

When you use `aws s3` commands to upload large objects to an Amazon S3 bucket, the AWS CLI automatically performs a multipart upload. You can't resume a failed upload when using these `aws s3` commands.

If the multipart upload fails due to a timeout, or if you manually canceled in the AWS CLI, the AWS CLI stops the upload and cleans up any files that were created. This process can take several minutes.

If the multipart upload or cleanup process is canceled by a kill command or system failure, the created files remain in the Amazon S3 bucket. To clean up the multipart upload, use the [s3api abort-multipart-upload](#) command.

File properties and tags in multipart copies

When you use the AWS CLI version 1 version of commands in the `aws s3` namespace to copy a file from one Amazon S3 bucket location to another Amazon S3 bucket location, and that operation uses [multipart copy](#), no file properties from the source object are copied to the destination object.

By default, the AWS CLI version 2 commands in the `s3` namespace that perform multipart copies transfers all tags and the following set of properties from the source to the destination copy: `content-type`, `content-language`, `content-encoding`, `content-disposition`, `cache-control`, `expires`, and `metadata`.

This can result in additional AWS API calls to the Amazon S3 endpoint that would not have been made if you used AWS CLI version 1. These can include: `HeadObject`, `GetObjectTagging`, and `PutObjectTagging`.

If you need to change this default behavior in AWS CLI version 2 commands, use the `--copy-props` parameter to specify one of the following options:

- **default** – The default value. Specifies that the copy includes all tags attached to the source object and the properties encompassed by the `--metadata-directive` parameter used for non-multipart copies: `content-type`, `content-language`, `content-encoding`, `content-disposition`, `cache-control`, `expires`, and `metadata`.
- **metadata-directive** – Specifies that the copy includes only the properties that are encompassed by the `--metadata-directive` parameter used for non-multipart copies. It doesn't copy any tags.
- **none** – Specifies that the copy includes none of the properties from the source object.

Create a bucket

Use the [s3 mb](#) command to make a bucket. Bucket names must be **globally** unique (unique across all of Amazon S3) and should be DNS compliant.

Bucket names can contain lowercase letters, numbers, hyphens, and periods. Bucket names can start and end only with a letter or number, and cannot contain a period next to a hyphen or another period.

Syntax

```
$ aws s3 mb <target> [--options]
```

s3 mb examples

The following example creates the `s3://bucket-name` bucket.

```
$ aws s3 mb s3://bucket-name
```

List buckets and objects

To list your buckets, folders, or objects, use the [s3 ls](#) command. Using the command without a target or options lists all buckets.

Syntax

```
$ aws s3 ls <target> [--options]
```

For a few common options to use with this command, and examples, see [Frequently used options for s3 commands](#). For a complete list of available options, see [s3 ls](#) in the *AWS CLI Command Reference*.

s3 ls examples

The following example lists all of your Amazon S3 buckets.

```
$ aws s3 ls
2018-12-11 17:08:50 my-bucket
2018-12-14 14:55:44 my-bucket2
```

The following command lists all objects and prefixes in a bucket. In this example output, the prefix `example/` has one file named `MyFile1.txt`.

```
$ aws s3 ls s3://bucket-name
                PRE example/
```

```
2018-12-04 19:05:48          3 MyFile1.txt
```

You can filter the output to a specific prefix by including it in the command. The following command lists the objects in *bucket-name/example/* (that is, objects in *bucket-name* filtered by the prefix *example/*).

```
$ aws s3 ls s3://bucket-name/example/  
2018-12-06 18:59:32          3 MyFile1.txt
```

Delete buckets

To delete a bucket, use the [s3 rb](#) command.

Syntax

```
$ aws s3 rb <target> [--options]
```

s3 rb examples

The following example removes the `s3://bucket-name` bucket.

```
$ aws s3 rb s3://bucket-name
```

By default, the bucket must be empty for the operation to succeed. To remove a bucket that's not empty, you need to include the `--force` option. If you're using a versioned bucket that contains previously deleted—but retained—objects, this command does *not* allow you to remove the bucket. You must first remove all of the content.

The following example deletes all objects and prefixes in the bucket, and then deletes the bucket.

```
$ aws s3 rb s3://bucket-name --force
```

Delete objects

To delete objects in a bucket or your local directory, use the [s3 rm](#) command.

Syntax

```
$ aws s3 rm <target> [--options]
```

For a few common options to use with this command, and examples, see [Frequently used options for s3 commands](#). For a complete list of options, see [s3 rm](#) in the *AWS CLI Command Reference*.

s3 rm examples

The following example deletes `filename.txt` from `s3://bucket-name/example`.

```
$ aws s3 rm s3://bucket-name/example/filename.txt
```

The following example deletes all objects from `s3://bucket-name/example` using the `--recursive` option.

```
$ aws s3 rm s3://bucket-name/example --recursive
```

Move objects

Use the [s3 mv](#) command to move objects from a bucket or a local directory. The `s3 mv` command copies the source object or file to the specified destination and then deletes the source object or file.

Syntax

```
$ aws s3 mv <source> <target> [--options]
```

For a few common options to use with this command, and examples, see [Frequently used options for s3 commands](#). For a complete list of available options, see [s3 mv](#) in the *AWS CLI Command Reference*.

Warning

If you are using any type of access point ARNs or access point aliases in your Amazon S3 source or destination URIs, you must take extra care that your source and destination Amazon S3 URIs resolve to different underlying buckets. If the source and destination buckets are the same, the source file or object can be moved onto itself, which can result in accidental deletion of your source file or object. To verify that the source and destination buckets are not the same, use the `--validate-same-s3-paths` parameter, or set the environment variable [AWS_CLI_S3_MV_VALIDATE_SAME_S3_PATHS](#) to `true`.

s3 mv examples

The following example moves all objects from `s3://bucket-name/example` to `s3://my-bucket/`.

```
$ aws s3 mv s3://bucket-name/example s3://my-bucket/
```

The following example moves a local file from your current working directory to the Amazon S3 bucket with the `s3 mv` command.

```
$ aws s3 mv filename.txt s3://bucket-name
```

The following example moves a file from your Amazon S3 bucket to your current working directory, where `./` specifies your current working directory.

```
$ aws s3 mv s3://bucket-name/filename.txt ./
```

Copy objects

Use the [s3 cp](#) command to copy objects from a bucket or a local directory.

Syntax

```
$ aws s3 cp <source> <target> [--options]
```

You can use the dash parameter for file streaming to standard input (`stdin`) or standard output (`stdout`).

Warning

If you're using PowerShell, the shell might alter the encoding of a CRLF or add a CRLF to piped input or output, or redirected output.

The `s3 cp` command uses the following syntax to upload a file stream from `stdin` to a specified bucket.

Syntax

```
$ aws s3 cp - <target> [--options]
```

The `s3 cp` command uses the following syntax to download an Amazon S3 file stream for stdout.

Syntax

```
$ aws s3 cp <target> [--options] -
```

For a few common options to use with this command, and examples, see [Frequently used options for s3 commands](#). For the complete list of options, see [s3 cp](#) in the *AWS CLI Command Reference*.

s3 cp examples

The following example copies all objects from `s3://bucket-name/example` to `s3://my-bucket/`.

```
$ aws s3 cp s3://bucket-name/example s3://my-bucket/
```

The following example copies a local file from your current working directory to the Amazon S3 bucket with the `s3 cp` command.

```
$ aws s3 cp filename.txt s3://bucket-name
```

The following example copies a file from your Amazon S3 bucket to your current working directory, where `./` specifies your current working directory.

```
$ aws s3 cp s3://bucket-name/filename.txt ./
```

The following example uses `echo` to stream the text "hello world" to the `s3://bucket-name/filename.txt` file.

```
$ echo "hello world" | aws s3 cp - s3://bucket-name/filename.txt
```

The following example streams the `s3://bucket-name/filename.txt` file to stdout and prints the contents to the console.

```
$ aws s3 cp s3://bucket-name/filename.txt -  
hello world
```

The following example streams the contents of `s3://bucket-name/pre` to stdout, uses the `bzip2` command to compress the files, and uploads the new compressed file named `key.bz2` to `s3://bucket-name`.

```
$ aws s3 cp s3://bucket-name/pre - | bzip2 --best | aws s3 cp - s3://bucket-name/key.bz2
```

Sync objects

The `s3 sync` command synchronizes the contents of a bucket and a directory, or the contents of two buckets. Typically, `s3 sync` copies missing or outdated files or objects between the source and target. However, you can also supply the `--delete` option to remove files or objects from the target that are not present in the source.

Syntax

```
$ aws s3 sync <source> <target> [--options]
```

For a few common options to use with this command, and examples, see [Frequently used options for s3 commands](#). For a complete list of options, see `s3 sync` in the *AWS CLI Command Reference*.

s3 sync examples

The following example synchronizes the contents of an Amazon S3 prefix named `path` in the bucket named `my-bucket` with the current working directory.

`s3 sync` updates any files that have a size or modified time that are different from files with the same name at the destination. The output displays specific operations performed during the sync. Notice that the operation recursively synchronizes the subdirectory `MySubdirectory` and its contents with `s3://my-bucket/path/MySubdirectory`.

```
$ aws s3 sync . s3://my-bucket/path
upload: MySubdirectory\MyFile3.txt to s3://my-bucket/path/MySubdirectory/MyFile3.txt
upload: MyFile2.txt to s3://my-bucket/path/MyFile2.txt
upload: MyFile1.txt to s3://my-bucket/path/MyFile1.txt
```

The following example, which extends the previous one, shows how to use the `--delete` option.

```
// Delete local file
```

```
$ rm ./MyFile1.txt

// Attempt sync without --delete option - nothing happens
$ aws s3 sync . s3://my-bucket/path

// Sync with deletion - object is deleted from bucket
$ aws s3 sync . s3://my-bucket/path --delete
delete: s3://my-bucket/path/MyFile1.txt

// Delete object from bucket
$ aws s3 rm s3://my-bucket/path/MySubdirectory/MyFile3.txt
delete: s3://my-bucket/path/MySubdirectory/MyFile3.txt

// Sync with deletion - local file is deleted
$ aws s3 sync s3://my-bucket/path . --delete
delete: MySubdirectory\MyFile3.txt

// Sync with Infrequent Access storage class
$ aws s3 sync . s3://my-bucket/path --storage-class STANDARD_IA
```

When using the `--delete` option, the `--exclude` and `--include` options can filter files or objects to delete during an `s3 sync` operation. In this case, the parameter string must specify files to exclude from, or include for, deletion in the context of the target directory or bucket. The following shows an example.

```
Assume local directory and s3://my-bucket/path currently in sync and each contains 3
files:
MyFile1.txt
MyFile2.rtf
MyFile88.txt
...

// Sync with delete, excluding files that match a pattern. MyFile88.txt is deleted,
while remote MyFile1.txt is not.
$ aws s3 sync . s3://my-bucket/path --delete --exclude "path/MyFile?.txt"
delete: s3://my-bucket/path/MyFile88.txt
...

// Sync with delete, excluding MyFile2.rtf - local file is NOT deleted
$ aws s3 sync s3://my-bucket/path . --delete --exclude "./MyFile2.rtf"
download: s3://my-bucket/path/MyFile1.txt to MyFile1.txt
...
```



```
// Sync with delete, local copy of MyFile2.rtf is deleted
$ aws s3 sync s3://my-bucket/path . --delete
delete: MyFile2.rtf
```

Frequently used options for s3 commands

The following options are frequently used for the commands described in this topic. For a complete list of options you can use on a command, see the specific command in the [AWS CLI version 2 reference guide](#).

acl

`s3 sync` and `s3 cp` can use the `--acl` option. This enables you to set the access permissions for files copied to Amazon S3. The `--acl` option accepts `private`, `public-read`, and `public-read-write` values. For more information, see [Canned ACL](#) in the *Amazon Simple Storage Service User Guide*.

```
$ aws s3 sync . s3://my-bucket/path --acl public-read
```

exclude

When you use the `s3 cp`, `s3 mv`, `s3 sync`, or `s3 rm` command, you can filter the results by using the `--exclude` or `--include` option. The `--exclude` option sets rules to only exclude objects from the command, and the options apply in the order specified. This is shown in the following example.

```
Local directory contains 3 files:
MyFile1.txt
MyFile2.rtf
MyFile88.txt

// Exclude all .txt files, resulting in only MyFile2.rtf being copied
$ aws s3 cp . s3://my-bucket/path --exclude "*.txt"

// Exclude all .txt files but include all files with the "MyFile*.txt" format,
resulting in, MyFile1.txt, MyFile2.rtf, MyFile88.txt being copied
$ aws s3 cp . s3://my-bucket/path --exclude "*.txt" --include "MyFile*.txt"

// Exclude all .txt files, but include all files with the "MyFile*.txt" format,
but exclude all files with the "MyFile?.txt" format resulting in, MyFile2.rtf and
MyFile88.txt being copied
```

```
$ aws s3 cp . s3://my-bucket/path --exclude "*.txt" --include "MyFile*.txt" --
exclude "MyFile?.txt"
```

include

When you use the `s3 cp`, `s3 mv`, `s3 sync`, or `s3 rm` command, you can filter the results using the `--exclude` or `--include` option. The `--include` option sets rules to only include objects specified for the command, and the options apply in the order specified. This is shown in the following example.

```
Local directory contains 3 files:
MyFile1.txt
MyFile2.rtf
MyFile88.txt

// Include all .txt files, resulting in MyFile1.txt and MyFile88.txt being copied
$ aws s3 cp . s3://my-bucket/path --include "*.txt"

// Include all .txt files but exclude all files with the "MyFile*.txt" format,
resulting in no files being copied
$ aws s3 cp . s3://my-bucket/path --include "*.txt" --exclude "MyFile*.txt"

// Include all .txt files, but exclude all files with the "MyFile*.txt" format, but
include all files with the "MyFile?.txt" format resulting in MyFile1.txt being
copied

$ aws s3 cp . s3://my-bucket/path --include "*.txt" --exclude "MyFile*.txt" --
include "MyFile?.txt"
```

grant

The `s3 cp`, `s3 mv`, and `s3 sync` commands include a `--grants` option that you can use to grant permissions on the object to specified users or groups. Set the `--grants` option to a list of permissions using the following syntax. Replace `Permission`, `Grantee_Type`, and `Grantee_ID` with your own values.

Syntax

```
--grants Permission=Grantee_Type=Grantee_ID
         [Permission=Grantee_Type=Grantee_ID ...]
```

Each value contains the following elements:

- *Permission* – Specifies the granted permissions. Can be set to `read`, `readacl`, `writeacl`, or `full`.
- *Grantee_Type* – Specifies how to identify the grantee. Can be set to `uri`, `emailaddress`, or `id`.
- *Grantee_ID* – Specifies the grantee based on *Grantee_Type*.
 - `uri` – The group's URI. For more information, see [Who is a grantee?](#)
 - `emailaddress` – The account's email address.
 - `id` – The account's canonical ID.

For more information about Amazon S3 access control, see [Access control](#).

The following example copies an object into a bucket. It grants read permissions on the object to everyone, and full permissions (`read`, `readacl`, and `writeacl`) to the account associated with `user@example.com`.

```
$ aws s3 cp file.txt s3://my-bucket/ --grants read=uri=http://acs.amazonaws.com/groups/global/AllUsers full=emailaddress=user@example.com
```

You can also specify a nondefault storage class (`REDUCED_REDUNDANCY` or `STANDARD_IA`) for objects that you upload to Amazon S3. To do this, use the `--storage-class` option.

```
$ aws s3 cp file.txt s3://my-bucket/ --storage-class REDUCED_REDUNDANCY
```

recursive

When you use this option, the command is performed on all files or objects under the specified directory or prefix. The following example deletes `s3://my-bucket/path` and all of its contents.

```
$ aws s3 rm s3://my-bucket/path --recursive
```

Resources

AWS CLI reference:

- [aws s3](#)

- [aws s3 cp](#)
- [aws s3 mb](#)
- [aws s3 mv](#)
- [aws s3 ls](#)
- [aws s3 rb](#)
- [aws s3 rm](#)
- [aws s3 sync](#)

Service reference:

- [Working with Amazon S3 buckets](#) in the *Amazon Simple Storage Service User Guide*
- [Working with Amazon S3 objects](#) in the *Amazon Simple Storage Service User Guide*
- [Listing keys hierarchically using a prefix and delimiter](#) in the *Amazon Simple Storage Service User Guide*
- [Abort multipart uploads to an S3 bucket using the AWS SDK for .NET \(low-level\)](#) in the *Amazon Simple Storage Service User Guide*

Use API-Level (s3api) commands with the AWS CLI

The API-level commands (contained in the `s3api` command set) provide direct access to the Amazon Simple Storage Service (Amazon S3) APIs, and enable some operations that are not exposed in the high-level `s3` commands. These commands are the equivalent of the other AWS services that provide API-level access to the services' functionality. For more information on the `s3` commands, see [Use high-level \(s3\) commands with the AWS CLI](#)

This topic provides examples that demonstrate how to use the lower-level commands that map to the Amazon S3 APIs. In addition, you can find examples for each S3 API command in the `s3api` section of the [AWS CLI version 2 reference guide](#).

Topics

- [Prerequisites](#)
- [Apply a custom ACL](#)
- [Configure a logging policy](#)
- [Resources](#)

Prerequisites

To run the `s3api` commands, you need to:

- Install and configure the AWS CLI. For more information, see [the section called “Install/Update”](#) and [Authentication and access credentials](#).
- The profile that you use must have permissions that allow the AWS operations performed by the examples.
- Understand these Amazon S3 terms:
 - **Bucket** – A top-level Amazon S3 folder.
 - **Prefix** – An Amazon S3 folder in a bucket.
 - **Object** – Any item that's hosted in an Amazon S3 bucket.

Apply a custom ACL

With high-level commands, you can use the `--acl` option to apply predefined access control lists (ACLs) to Amazon S3 objects. But you can't use that command to set bucket-wide ACLs. However, you can do this by using the [put-bucket-acl](#) API-level command.

The following example shows how to grant full control to two AWS users (`user1@example.com` and `user2@example.com`) and read permission to everyone. The identifier for "everyone" comes from a special URI that you pass as a parameter.

```
$ aws s3api put-bucket-acl --bucket MyBucket --grant-full-control  
'emailaddress="user1@example.com",emailaddress="user2@example.com"' --grant-read  
'uri="http://acs.amazonaws.com/groups/global/AllUsers"'
```

For details about how to construct the ACLs, see [PUT Bucket acl](#) in the *Amazon Simple Storage Service API Reference*. The `s3api` ACL commands in the CLI, such as `put-bucket-acl`, use the same [shorthand argument notation](#).

Configure a logging policy

The API command `put-bucket-logging` configures a bucket logging policy.

In the following example, the AWS user `user@example.com` is granted full control over the log files, and all users have read access to them. Notice that the `put-bucket-acl` command is also

required to grant the Amazon S3 log delivery system (specified by a URI) the permissions needed to read and write the logs to the bucket.

```
$ aws s3api put-bucket-acl --bucket MyBucket --grant-read-acp 'URI="http://acs.amazonaws.com/groups/s3/LogDelivery"' --grant-write 'URI="http://acs.amazonaws.com/groups/s3/LogDelivery"'
$ aws s3api put-bucket-logging --bucket MyBucket --bucket-logging-status file://logging.json
```

The `logging.json` file in the previous command has the following content.

```
{
  "LoggingEnabled": {
    "TargetBucket": "MyBucket",
    "TargetPrefix": "MyBucketLogs/",
    "TargetGrants": [
      {
        "Grantee": {
          "Type": "AmazonCustomerByEmail",
          "EmailAddress": "user@example.com"
        },
        "Permission": "FULL_CONTROL"
      },
      {
        "Grantee": {
          "Type": "Group",
          "URI": "http://acs.amazonaws.com/groups/global/AllUsers"
        },
        "Permission": "READ"
      }
    ]
  }
}
```

Resources

AWS CLI reference:

- [aws s3api](#)
- [aws s3api put-bucket-acl](#)
- [aws s3api put-bucket-logging](#)

Service reference:

- [Working with Amazon S3 buckets](#) in the *Amazon Simple Storage Service User Guide*
- [Working with Amazon S3 objects](#) in the *Amazon Simple Storage Service User Guide*
- [Listing keys hierarchically using a prefix and delimiter](#) in the *Amazon Simple Storage Service User Guide*
- [Abort multipart uploads to an S3 bucket using the AWS SDK for .NET \(low-level\)](#) in the *Amazon Simple Storage Service User Guide*

Amazon S3 bucket lifecycle operations scripting example

This topic uses a bash scripting example for Amazon S3 bucket lifecycle operations using the AWS Command Line Interface (AWS CLI). This scripting example uses the [aws s3api](#) set of commands. Shell scripts are programs designed to run in a command line interface.

Topics

- [Before you start](#)
- [About this example](#)
- [Files](#)
- [References](#)

Before you start

Before you can run any of the below examples, the following things need to be completed.

- Install and configure the AWS CLI. For more information, see [the section called “Install/Update”](#) and [Authentication and access credentials](#).
- The profile that you use must have permissions that allow the AWS operations performed by the examples.
- As an AWS best practice, grant this code least privilege, or only the permissions required to perform a task. For more information, see [Grant Least Privilege](#) in the *IAM User Guide*.
- This code has not been tested in all AWS Regions. Some AWS services are available only in specific Regions. For more information, see [Service Endpoints and Quotas](#) in the *AWS General Reference Guide*.

- Running this code can result in charges to your AWS account. It is your responsibility to ensure that any resources created by this script are removed when you are done with them.

The Amazon S3 service uses the following terms:

- **Bucket** — A top level Amazon S3 folder.
- **Prefix** — An Amazon S3 folder in a bucket.
- **Object** — Any item hosted in an Amazon S3 bucket.

About this example

This example demonstrates how to interact with some of the basic Amazon S3 operations using a set of functions in shell script files. The functions are located in the shell script file named `bucket-operations.sh`. You can call these functions in another file. Each script file contains comments describing each of the functions.

To see the intermediate results of each step, run the script with a `-i` parameter. You can view the current status of the bucket or its contents using the Amazon S3 console. The script only proceeds to the next step when you press **enter** at the prompt.

For the full example and downloadable script files, see [Amazon S3 Bucket Lifecycle Operations](#) in the *AWS Code Examples Repository on GitHub*.

Files

The example contains the following files:

bucket-operations.sh

This main script file can be sourced from another file. It includes functions that perform the following tasks:

- Creating a bucket and verifying that it exists
- Copying a file from the local computer to a bucket
- Copying a file from one bucket location to a different bucket location
- Listing the contents of a bucket
- Deleting a file from a bucket
- Deleting a bucket

View the code for [bucket-operations.sh](#) on *GitHub*.

test-bucket-operations.sh

The shell script file `test-bucket-operations.sh` demonstrates how to call the functions by sourcing the `bucket-operations.sh` file and calling each of the functions. After calling functions, the test script removes all resources that it created.

View the code for [test-bucket-operations.sh](#) on *GitHub*.

awsdocs-general.sh

The script file `awsdocs-general.sh` holds general purpose functions used across advanced code examples for the AWS CLI.

View the code for [awsdocs-general.sh](#) on *GitHub*.

References

AWS CLI reference:

- [aws s3api](#)
- [aws s3api create-bucket](#)
- [aws s3api copy-object](#)
- [aws s3api delete-bucket](#)
- [aws s3api delete-object](#)
- [aws s3api head-bucket](#)
- [aws s3api list-objects](#)
- [aws s3api put-object](#)

Other reference:

- [Working with Amazon S3 buckets](#) in the *Amazon Simple Storage Service User Guide*
- [Working with Amazon S3 objects](#) in the *Amazon Simple Storage Service User Guide*
- To view and contribute to AWS SDK and AWS CLI code examples, see the [AWS Code Examples Repository](#) on *GitHub*.

Use Amazon SNS with the AWS CLI

You can access the features of Amazon Simple Notification Service (Amazon SNS) using the AWS Command Line Interface (AWS CLI). To list the AWS CLI commands for Amazon SNS, use the following command.

```
aws sns help
```

Before you run any commands, set your default credentials. For more information, see [Configure the AWS CLI](#).

This topic shows examples of AWS CLI commands that perform common tasks for Amazon SNS.

Topics

- [Create a topic](#)
- [Subscribe to a topic](#)
- [Publish to a topic](#)
- [Unsubscribe from a topic](#)
- [Delete a topic](#)

Create a topic

To create a topic, use the [sns create-topic](#) command and specify the name to assign to the topic.

```
$ aws sns create-topic --name my-topic
{
  "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic"
}
```

Make a note of the response's `TopicArn`, which you use later to publish a message.

Subscribe to a topic

To subscribe to a topic, use the [sns subscribe](#) command.

The following example specifies the email protocol and an email address for the notification-endpoint.

```
$ aws sns subscribe --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic --
protocol email --notification-endpoint saanvi@example.com
{
  "SubscriptionArn": "pending confirmation"
}
```

AWS immediately sends a confirmation message by email to the address you specified in the subscribe command. The email message has the following text.

```
You have chosen to subscribe to the topic:
arn:aws:sns:us-west-2:123456789012:my-topic
To confirm this subscription, click or visit the following link (If this was in error
no action is necessary):
Confirm subscription
```

After the recipient clicks the **Confirm subscription** link, the recipient's browser displays a notification message with information similar to the following.

```
Subscription confirmed!

You have subscribed saanvi@example.com to the topic:my-topic.

Your subscription's id is:
arn:aws:sns:us-west-2:123456789012:my-topic:1328f057-de93-4c15-512e-8bb22EXAMPLE

If it was not your intention to subscribe, click here to unsubscribe.
```

Publish to a topic

To send a message to all subscribers of a topic, use the [sns publish](#) command.

The following example sends the message "Hello World!" to all subscribers of the specified topic.

```
$ aws sns publish --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic --
message "Hello World!"
{
  "MessageId": "4e41661d-5eec-5ddf-8dab-2c867EXAMPLE"
}
```

In this example, AWS sends an email message with the text "Hello World!" to saanvi@example.com.

Unsubscribe from a topic

To unsubscribe from a topic and stop receiving messages published to that topic, use the [sns unsubscribe](#) command and specify the ARN of the topic you want to unsubscribe from.

```
$ aws sns unsubscribe --subscription-arn arn:aws:sns:us-west-2:123456789012:my-topic:1328f057-de93-4c15-512e-8bb22EXAMPLE
```

To verify that you successfully unsubscribed, use the [sns list-subscriptions](#) command to confirm that the ARN no longer appears in the list.

```
$ aws sns list-subscriptions
```

Delete a topic

To delete a topic, run the [sns delete-topic](#) command.

```
$ aws sns delete-topic --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic
```

To verify that AWS successfully deleted the topic, use the [sns list-topics](#) command to confirm that the topic no longer appears in the list.

```
$ aws sns list-topics
```

AWS CLI command examples

The code examples in this topic show you how to use the AWS Command Line Interface with AWS.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Cross-service examples are sample applications that work across multiple AWS services.

Examples

- [Actions and scenarios using AWS CLI](#)

Actions and scenarios using AWS CLI

The following code examples show how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS services.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Services

- [ACM examples using AWS CLI](#)
- [API Gateway examples using AWS CLI](#)
- [API Gateway HTTP and WebSocket API examples using AWS CLI](#)
- [API Gateway Management API examples using AWS CLI](#)
- [App Mesh examples using AWS CLI](#)
- [App Runner examples using AWS CLI](#)
- [AWS AppConfig examples using AWS CLI](#)
- [Application Auto Scaling examples using AWS CLI](#)
- [Application Discovery Service examples using AWS CLI](#)
- [AppRegistry examples using AWS CLI](#)
- [Athena examples using AWS CLI](#)
- [Auto Scaling examples using AWS CLI](#)
- [Auto Scaling Plans examples using AWS CLI](#)
- [AWS Backup examples using AWS CLI](#)
- [AWS Batch examples using AWS CLI](#)
- [AWS Budgets examples using AWS CLI](#)
- [Amazon Chime examples using AWS CLI](#)
- [Cloud Control API examples using AWS CLI](#)

- [AWS Cloud Map examples using AWS CLI](#)
- [AWS Cloud9 examples using AWS CLI](#)
- [AWS CloudFormation examples using AWS CLI](#)
- [CloudFront examples using AWS CLI](#)
- [Amazon CloudSearch examples using AWS CLI](#)
- [CloudTrail examples using AWS CLI](#)
- [CloudWatch examples using AWS CLI](#)
- [CloudWatch Logs examples using AWS CLI](#)
- [CloudWatch Network Monitoring examples using AWS CLI](#)
- [CodeArtifact examples using AWS CLI](#)
- [CodeBuild examples using AWS CLI](#)
- [CodeCommit examples using AWS CLI](#)
- [CodeDeploy examples using AWS CLI](#)
- [CodeGuru Reviewer examples using AWS CLI](#)
- [CodePipeline examples using AWS CLI](#)
- [AWS CodeStar examples using AWS CLI](#)
- [AWS CodeStar Notifications examples using AWS CLI](#)
- [CodeConnections examples using AWS CLI](#)
- [Amazon Cognito Identity examples using AWS CLI](#)
- [Amazon Cognito Identity Provider examples using AWS CLI](#)
- [Amazon Comprehend examples using AWS CLI](#)
- [Amazon Comprehend Medical examples using AWS CLI](#)
- [AWS Config examples using AWS CLI](#)
- [Amazon Connect examples using AWS CLI](#)
- [AWS Cost and Usage Report examples using AWS CLI](#)
- [Cost Explorer Service examples using AWS CLI](#)
- [Firehose examples using AWS CLI](#)
- [Amazon Data Lifecycle Manager examples using AWS CLI](#)
- [AWS Data Pipeline examples using AWS CLI](#)
- [DataSync examples using AWS CLI](#)

- [DAX examples using AWS CLI](#)
- [Detective examples using AWS CLI](#)
- [Device Farm examples using AWS CLI](#)
- [AWS Direct Connect examples using AWS CLI](#)
- [AWS Directory Service examples using AWS CLI](#)
- [AWS DMS examples using AWS CLI](#)
- [Amazon DocumentDB examples using AWS CLI](#)
- [DynamoDB examples using AWS CLI](#)
- [DynamoDB Streams examples using AWS CLI](#)
- [Amazon EC2 examples using AWS CLI](#)
- [Amazon EC2 Instance Connect examples using AWS CLI](#)
- [Amazon ECR examples using AWS CLI](#)
- [Amazon ECS examples using AWS CLI](#)
- [Amazon EFS examples using AWS CLI](#)
- [Amazon EKS examples using AWS CLI](#)
- [Elastic Beanstalk examples using AWS CLI](#)
- [Elastic Load Balancing - Version 1 examples using AWS CLI](#)
- [Elastic Load Balancing - Version 2 examples using AWS CLI](#)
- [Elastic Transcoder examples using AWS CLI](#)
- [ElastiCache examples using AWS CLI](#)
- [MediaStore examples using AWS CLI](#)
- [Amazon EMR examples using AWS CLI](#)
- [Amazon EMR on EKS examples using AWS CLI](#)
- [EventBridge examples using AWS CLI](#)
- [Firewall Manager examples using AWS CLI](#)
- [AWS FIS examples using AWS CLI](#)
- [Amazon GameLift examples using AWS CLI](#)
- [Global Accelerator examples using AWS CLI](#)
- [AWS Glue examples using AWS CLI](#)
- [GuardDuty examples using AWS CLI](#)

- [AWS Health examples using AWS CLI](#)
- [HealthImaging examples using AWS CLI](#)
- [HealthLake examples using AWS CLI](#)
- [HealthOmics examples using AWS CLI](#)
- [IAM examples using AWS CLI](#)
- [IAM Access Analyzer examples using AWS CLI](#)
- [Image Builder examples using AWS CLI](#)
- [Incident Manager examples using AWS CLI](#)
- [Incident Manager Contacts examples using AWS CLI](#)
- [Amazon Inspector examples using AWS CLI](#)
- [AWS IoT examples using AWS CLI](#)
- [AWS IoT 1-Click Devices examples using AWS CLI](#)
- [AWS IoT 1-Click Projects examples using AWS CLI](#)
- [AWS IoT Analytics examples using AWS CLI](#)
- [Device Advisor examples using AWS CLI](#)
- [AWS IoT data examples using AWS CLI](#)
- [AWS IoT Events examples using AWS CLI](#)
- [AWS IoT Events-Data examples using AWS CLI](#)
- [AWS IoT Greengrass examples using AWS CLI](#)
- [AWS IoT Greengrass V2 examples using AWS CLI](#)
- [AWS IoT Jobs SDK release examples using AWS CLI](#)
- [AWS IoT SiteWise examples using AWS CLI](#)
- [AWS IoT Things Graph examples using AWS CLI](#)
- [AWS IoT Wireless examples using AWS CLI](#)
- [Amazon IVS examples using AWS CLI](#)
- [Amazon IVS Chat examples using AWS CLI](#)
- [Amazon IVS Real-Time Streaming examples using AWS CLI](#)
- [Amazon Kendra examples using AWS CLI](#)
- [Kinesis examples using AWS CLI](#)
- [AWS KMS examples using AWS CLI](#)

- [Lake Formation examples using AWS CLI](#)
- [Lambda examples using AWS CLI](#)
- [License Manager examples using AWS CLI](#)
- [Lightsail examples using AWS CLI](#)
- [Macie examples using AWS CLI](#)
- [Amazon Managed Grafana examples using AWS CLI](#)
- [MediaConnect examples using AWS CLI](#)
- [MediaConvert examples using AWS CLI](#)
- [MediaLive examples using AWS CLI](#)
- [MediaPackage examples using AWS CLI](#)
- [MediaPackage VOD examples using AWS CLI](#)
- [MediaStore Data Plane examples using AWS CLI](#)
- [MediaTailor examples using AWS CLI](#)
- [MemoryDB examples using AWS CLI](#)
- [Amazon MSK examples using AWS CLI](#)
- [Network Manager examples using AWS CLI](#)
- [Nimble Studio examples using AWS CLI](#)
- [OpenSearch Service examples using AWS CLI](#)
- [AWS OpsWorks examples using AWS CLI](#)
- [AWS OpsWorks CM examples using AWS CLI](#)
- [Organizations examples using AWS CLI](#)
- [AWS Outposts examples using AWS CLI](#)
- [AWS Payment Cryptography examples using AWS CLI](#)
- [AWS Payment Cryptography Data Plane examples using AWS CLI](#)
- [Amazon Pinpoint examples using AWS CLI](#)
- [Amazon Polly examples using AWS CLI](#)
- [AWS Price List examples using AWS CLI](#)
- [AWS Private CA examples using AWS CLI](#)
- [AWS Proton examples using AWS CLI](#)
- [QLDB examples using AWS CLI](#)

- [Amazon RDS examples using AWS CLI](#)
- [Amazon RDS Data Service examples using AWS CLI](#)
- [Amazon RDS Performance Insights examples using AWS CLI](#)
- [Amazon Redshift examples using AWS CLI](#)
- [Amazon Rekognition examples using AWS CLI](#)
- [AWS RAM examples using AWS CLI](#)
- [Resource Explorer examples using AWS CLI](#)
- [Resource Groups examples using AWS CLI](#)
- [Resource Groups Tagging API examples using AWS CLI](#)
- [AWS RoboMaker examples using AWS CLI](#)
- [Route 53 examples using AWS CLI](#)
- [Route 53 domain registration examples using AWS CLI](#)
- [Route 53 Resolver examples using AWS CLI](#)
- [Amazon S3 examples using AWS CLI](#)
- [Amazon S3 Control examples using AWS CLI](#)
- [S3 Glacier examples using AWS CLI](#)
- [Secrets Manager examples using AWS CLI](#)
- [Security Hub examples using AWS CLI](#)
- [AWS Serverless Application Repository examples using AWS CLI](#)
- [Service Catalog examples using AWS CLI](#)
- [Service Quotas examples using AWS CLI](#)
- [Amazon SES examples using AWS CLI](#)
- [Shield examples using AWS CLI](#)
- [Signer examples using AWS CLI](#)
- [Snowball examples using AWS CLI](#)
- [Amazon SNS examples using AWS CLI](#)
- [Amazon SQS examples using AWS CLI](#)
- [Storage Gateway examples using AWS CLI](#)
- [AWS STS examples using AWS CLI](#)
- [AWS Support examples using AWS CLI](#)

- [Amazon SWF examples using AWS CLI](#)
- [Systems Manager examples using AWS CLI](#)
- [Amazon Textract examples using AWS CLI](#)
- [Amazon Transcribe examples using AWS CLI](#)
- [Amazon Translate examples using AWS CLI](#)
- [Trusted Advisor examples using AWS CLI](#)
- [Verified Permissions examples using AWS CLI](#)
- [VPC Lattice examples using AWS CLI](#)
- [AWS WAF Classic examples using AWS CLI](#)
- [AWS WAF Classic Regional examples using AWS CLI](#)
- [AWS WAFV2 examples using AWS CLI](#)
- [Amazon WorkDocs examples using AWS CLI](#)
- [Amazon WorkMail examples using AWS CLI](#)
- [Amazon WorkMail Message Flow examples using AWS CLI](#)
- [WorkSpaces examples using AWS CLI](#)
- [X-Ray examples using AWS CLI](#)

ACM examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with ACM.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-tags-to-certificate

The following code example shows how to use `add-tags-to-certificate`.

AWS CLI

To add tags to an existing ACM Certificate

The following `add-tags-to-certificate` command adds two tags to the specified certificate. Use a space to separate multiple tags:

```
aws acm add-tags-to-certificate --certificate-arn
arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012 --tags
Key=Admin,Value=Alice Key=Purpose,Value=Website
```

- For API details, see [AddTagsToCertificate](#) in *AWS CLI Command Reference*.

delete-certificate

The following code example shows how to use `delete-certificate`.

AWS CLI

To delete an ACM certificate from your account

The following `delete-certificate` command deletes the certificate with the specified ARN:

```
aws acm delete-certificate --certificate-arn
arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012
```

- For API details, see [DeleteCertificate](#) in *AWS CLI Command Reference*.

describe-certificate

The following code example shows how to use `describe-certificate`.

AWS CLI

To retrieve the fields contained in an ACM certificate

The following describe-certificate command retrieves all of the fields for the certificate with the specified ARN:

```
aws acm describe-certificate --certificate-arn
arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012
```

Output similar to the following is displayed:

```
{
  "Certificate": {
    "CertificateArn":
"arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012",
    "CreatedAt": 1446835267.0,
    "DomainName": "www.example.com",
    "DomainValidationOptions": [
      {
        "DomainName": "www.example.com",
        "ValidationDomain": "www.example.com",
        "ValidationEmails": [
          "hostmaster@example.com",
          "admin@example.com",
          "owner@example.com.whoisprivacyservice.org",
          "tech@example.com.whoisprivacyservice.org",
          "admin@example.com.whoisprivacyservice.org",
          "postmaster@example.com",
          "webmaster@example.com",
          "administrator@example.com"
        ]
      }
    ],
  },
  {
    "DomainName": "www.example.net",
    "ValidationDomain": "www.example.net",
    "ValidationEmails": [
      "postmaster@example.net",
      "admin@example.net",
      "owner@example.net.whoisprivacyservice.org",
      "tech@example.net.whoisprivacyservice.org",
      "admin@example.net.whoisprivacyservice.org",
      "hostmaster@example.net",
      "administrator@example.net",
      "webmaster@example.net"
    ]
  }
}
```

```

    ],
    "InUseBy": [],
    "IssuedAt": 1446835815.0,
    "Issuer": "Amazon",
    "KeyAlgorithm": "RSA-2048",
    "NotAfter": 1478433600.0,
    "NotBefore": 1446768000.0,
    "Serial": "0f:ac:b0:a3:8d:ea:65:52:2d:7d:01:3a:39:36:db:d6",
    "SignatureAlgorithm": "SHA256WITHRSA",
    "Status": "ISSUED",
    "Subject": "CN=www.example.com",
    "SubjectAlternativeNames": [
      "www.example.com",
      "www.example.net"
    ]
  }
}

```

- For API details, see [DescribeCertificate](#) in *AWS CLI Command Reference*.

export-certificate

The following code example shows how to use `export-certificate`.

AWS CLI

To export a private certificate issued by a private CA.

The following `export-certificate` command exports a private certificate, certificate chain, and private key to your display:

```

aws acm export-certificate --certificate-arn
arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012 --
passphrase file://path-to-passphrase-file

```

To export the certificate, chain, and private key to a local file, use the following command:

```

aws acm export-certificate --certificate-arn
arn:aws:acm:region:sccount:certificate/12345678-1234-1234-1234-123456789012 --
passphrase file://path-to-passphrase-file > c:\temp\export.txt

```

- For API details, see [ExportCertificate](#) in *AWS CLI Command Reference*.

get-certificate

The following code example shows how to use `get-certificate`.

AWS CLI

To retrieve an ACM certificate

The following `get-certificate` command retrieves the certificate for the specified ARN and the certificate chain:

```
aws acm get-certificate --certificate-arn
arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012
```

Output similar to the following is displayed:

```
{
  "Certificate": "-----BEGIN CERTIFICATE-----
MIICiTCCAFICCCQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBA5TC0lBTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWVxHmAd
BgkqhkiG9w0BCQEWEG5vb251QGftYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYD
VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBA5TC0lBTSBDb25z
b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWVxHmAdBgkqhkiG9w0BCQEWEG5vb251QGft
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLygVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb30hjZnczvQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVvxYUntneD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
FFbjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjStB
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----",
  "CertificateChain": "-----BEGIN CERTIFICATE-----
MIICiTCCAFICCCQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBA5TC0lBTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWVxHmAd
BgkqhkiG9w0BCQEWEG5vb251QGftYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYD
VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBA5TC0lBTSBDb25z
b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWVxHmAdBgkqhkiG9w0BCQEWEG5vb251QGft
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
```

```

21uUSfwfEvySwTC2XADZ4nB+BLyGVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb30hjZnczvQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUHVvXyUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjStB
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----",
"-----BEGIN CERTIFICATE-----
MIICiTCcAfICCQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwDTZWf0dGx1MQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAwTC01BTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWVxHmZAd
BgkqhkiG9w0BCQEWEG5vb25lQGftYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
VQQHEwDTZWf0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAwTC01BTSBDb25z
b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWVxHmZAdBgkqhkiG9w0BCQEWEG5vb25lQGft
YXpvbi5jb20wZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySwTC2XADZ4nB+BLyGVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb30hjZnczvQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUHVvXyUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjStB
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----",
"-----BEGIN CERTIFICATE-----
MIICiTCcAfICCQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwDTZWf0dGx1MQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAwTC01BTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWVxHmZAd
BgkqhkiG9w0BCQEWEG5vb25lQGftYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
VQQHEwDTZWf0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAwTC01BTSBDb25z
b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWVxHmZAdBgkqhkiG9w0BCQEWEG5vb25lQGft
YXpvbi5jb20wZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySwTC2XADZ4nB+BLyGVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb30hjZnczvQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUHVvXyUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjStB
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----"
}

```

- For API details, see [GetCertificate](#) in *AWS CLI Command Reference*.

import-certificate

The following code example shows how to use `import-certificate`.

AWS CLI

To import a certificate into ACM.

The following `import-certificate` command imports a certificate into ACM. Replace the file names with your own:

```
aws acm import-certificate --certificate file://Certificate.pem --certificate-chain
file://CertificateChain.pem --private-key file://PrivateKey.pem
```

- For API details, see [ImportCertificate](#) in *AWS CLI Command Reference*.

list-certificates

The following code example shows how to use `list-certificates`.

AWS CLI

To list the ACM certificates for an AWS account

The following `list-certificates` command lists the ARNs of the certificates in your account:

```
aws acm list-certificates
```

The preceding command produces output similar to the following:

```
{
  "CertificateSummaryList": [
    {
      "CertificateArn":
"arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012",
      "DomainName": "www.example.com"
    },
    {
      "CertificateArn": "arn:aws:acm:region:account:certificate/aaaaaaaa-bbbb-
cccc-dddd-eeeeeeeeeeee",

```

```

        "DomainName": "www.example.net"
    }
]
}

```

You can decide how many certificates you want to display each time you call `list-certificates`. For example, if you have four certificates and you want to display no more than two at a time, set the `max-items` argument to 2 as in the following example:

```
aws acm list-certificates --max-items 2
```

Two certificate ARNs and a `NextToken` value will be displayed:

```

"CertificateSummaryList": [
  {
    "CertificateArn": "arn:aws:acm:region:account: \
      certificate/12345678-1234-1234-1234-123456789012",
    "DomainName": "www.example.com"
  },
  {
    "CertificateArn": "arn:aws:acm:region:account: \
      certificate/aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
    "DomainName": "www.example.net"
  }
],
"NextToken": "9f4d9f69-275a-41fe-b58e-2b837bd9ba48"

```

To display the next two certificates in your account, set this `NextToken` value in your next call:

```
aws acm list-certificates --max-items 2 --next-token 9f4d9f69-275a-41fe-
b58e-2b837bd9ba48
```

You can filter your output by using the `certificate-statuses` argument. The following command displays certificates that have a `PENDING_VALIDATION` status:

```
aws acm list-certificates --certificate-statuses PENDING_VALIDATION
```

You can also filter your output by using the `includes` argument. The following command displays certificates filtered on the following properties. The certificates to be displayed:

- Specify that the RSA algorithm and a 2048 bit key are used to generate key pairs.
- Contain a Key Usage extension that specifies that the certificates can be used to create digital signatures.
- Contain an Extended Key Usage extension that specifies that the certificates can be used for code signing.

```
aws acm list-certificates --max-items 10 --includes
  extendedKeyUsage=CODE_SIGNING,keyUsage=DIGITAL_SIGNATURE,keyTypes=RSA_2048
```

- For API details, see [ListCertificates](#) in *AWS CLI Command Reference*.

list-tags-for-certificate

The following code example shows how to use `list-tags-for-certificate`.

AWS CLI

To list the tags applied to an ACM Certificate

The following `list-tags-for-certificate` command lists the tags applied to a certificate in your account:

```
aws acm list-tags-for-certificate --certificate-arn
  arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012
```

The preceding command produces output similar to the following:

```
{
  "Tags": [
    {
      "Value": "Website",
      "Key": "Purpose"
    },
    {
      "Value": "Alice",
      "Key": "Admin"
    }
  ]
}
```

- For API details, see [ListTagsForCertificate](#) in *AWS CLI Command Reference*.

remove-tags-from-certificate

The following code example shows how to use `remove-tags-from-certificate`.

AWS CLI

To remove a tag from an ACM Certificate

The following `remove-tags-from-certificate` command removes two tags from the specified certificate. Use a space to separate multiple tags:

```
aws acm remove-tags-from-certificate --certificate-arn
arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012 --tags
Key=Admin,Value=Alice Key=Purpose,Value=Website
```

- For API details, see [RemoveTagsFromCertificate](#) in *AWS CLI Command Reference*.

request-certificate

The following code example shows how to use `request-certificate`.

AWS CLI

To request a new ACM certificate

The following `request-certificate` command requests a new certificate for the `www.example.com` domain using DNS validation:

```
aws acm request-certificate --domain-name www.example.com --validation-method DNS
```

You can enter an idempotency token to distinguish between calls to `request-certificate`:

```
aws acm request-certificate --domain-name www.example.com --validation-method DNS --
idempotency-token 91adc45q
```

You can enter one or more subject alternative names to request a certificate that will protect more than one apex domain:

```
aws acm request-certificate --domain-name example.com --validation-method DNS --
idempotency-token 91adc45q --subject-alternative-names www.example.net
```

You can enter an alternative name that can also be used to reach your website:

```
aws acm request-certificate --domain-name example.com --validation-method DNS --  
idempotency-token 91adc45q --subject-alternative-names www.example.com
```

You can use an asterisk (*) as a wildcard to create a certificate for several subdomains in the same domain:

```
aws acm request-certificate --domain-name example.com --validation-method DNS --  
idempotency-token 91adc45q --subject-alternative-names *.example.com
```

You can also enter multiple alternative names:

```
aws acm request-certificate --domain-name example.com --validation-method DNS --  
subject-alternative-names b.example.com c.example.com d.example.com
```

If you are using email for validation, you can enter domain validation options to specify the domain to which the validation email will be sent:

```
aws acm request-certificate --domain-name example.com --validation-method  
EMAIL --subject-alternative-names www.example.com --domain-validation-options  
DomainName=example.com,ValidationDomain=example.com
```

The following command opts out of certificate transparency logging when you request a new certificate:

```
aws acm request-certificate --domain-name www.example.com --validation-method DNS --  
options CertificateTransparencyLoggingPreference=DISABLED --idempotency-token 184627
```

- For API details, see [RequestCertificate](#) in *AWS CLI Command Reference*.

resend-validation-email

The following code example shows how to use `resend-validation-email`.

AWS CLI

To resend validation email for your ACM certificate request

The following `resend-validation-email` command tells the Amazon certificate authority to send validation email to the appropriate addresses:

```
aws acm resend-validation-email --certificate-arn
arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012 --
domain www.example.com --validation-domain example.com
```

- For API details, see [ResendValidationEmail](#) in *AWS CLI Command Reference*.

update-certificate-options

The following code example shows how to use `update-certificate-options`.

AWS CLI

To update the certificate options

The following `update-certificate-options` command opts out of certificate transparency logging:

```
aws acm update-certificate-options --certificate-arn
arn:aws:acm:region:account:certificate/12345678-1234-1234-1234-123456789012 --
options CertificateTransparencyLoggingPreference=DISABLED
```

- For API details, see [UpdateCertificateOptions](#) in *AWS CLI Command Reference*.

API Gateway examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with API Gateway.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-api-key

The following code example shows how to use `create-api-key`.

AWS CLI

To create an API key that is enabled for an existing API and Stage

Command:

```
aws apigateway create-api-key --name 'Dev API Key' --description 'Used for
development' --enabled --stage-keys restApiId='a1b2c3d4e5',stageName='dev'
```

- For API details, see [CreateApiKey](#) in *AWS CLI Command Reference*.

create-authorizer

The following code example shows how to use `create-authorizer`.

AWS CLI

Example 1: To create a token-based API Gateway Custom Authorizer for the API

The following `create-authorizer` example creates a token-based authorizer.

```
aws apigateway create-authorizer \
  --rest-api-id 1234123412 \
  --name 'First-Token-Custom-Authorizer' \
  --type TOKEN \
  --authorizer-uri 'arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-west-2:123412341234:function:customAuthFunction/invocations' \
  --identity-source 'method.request.header.Authorization' \
  --authorizer-result-ttl-in-seconds 300
```

Output:

```
{
  "authType": "custom",
  "name": "First-Token-Custom-Authorizer",
  "authorizerUri": "arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/arn:aws:lambda:us-west-2:123412341234:function:customAuthFunction/invocations",
  "authorizerResultTtlInSeconds": 300,
  "identitySource": "method.request.header.Authorization",
  "type": "TOKEN",
  "id": "z40xj0"
}
```

Example 2: To create a Cognito User Pools based API Gateway Custom Authorizer for the API

The following `create-authorizer` example creates a Cognito User Pools based API Gateway Custom Authorizer.

```
aws apigateway create-authorizer \
  --rest-api-id 1234123412 \
  --name 'First_Cognito_Custom_Authorizer' \
  --type COGNITO_USER_POOLS \
  --provider-arns 'arn:aws:cognito-idp:us-east-1:123412341234:userpool/us-east-1_aWcZeQbuD' \
  --identity-source 'method.request.header.Authorization'
```

Output:

```
{
  "authType": "cognito_user_pools",
  "identitySource": "method.request.header.Authorization",
  "name": "First_Cognito_Custom_Authorizer",
  "providerARNs": [
    "arn:aws:cognito-idp:us-east-1:342398297714:userpool/us-east-1_qWbZzQhzE"
  ],
  "type": "COGNITO_USER_POOLS",
  "id": "5yid1t"
}
```

Example 3: To create a request-based API Gateway Custom Authorizer for the API

The following `create-authorizer` example creates a request-based authorizer.

```
aws apigateway create-authorizer \
```



```
--rest-api-id 1234123412 \  
--name 'First_Request_Custom_Authorizer' \  
--type REQUEST \  
--authorizer-uri 'arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/  
arn:aws:lambda:us-west-2:123412341234:function:customAuthFunction/invocations' \  
--identity-source 'method.request.header.Authorization,context.accountId' \  
--authorizer-result-ttl-in-seconds 300
```

Output:

```
{  
  "id": "z40xj0",  
  "name": "First_Request_Custom_Authorizer",  
  "type": "REQUEST",  
  "authType": "custom",  
  "authorizerUri": "arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/  
arn:aws:lambda:us-west-2:123412341234:function:customAuthFunction/invocations",  
  "identitySource": "method.request.header.Authorization,context.accountId",  
  "authorizerResultTtlInSeconds": 300  
}
```

- For API details, see [CreateAuthorizer](#) in *AWS CLI Command Reference*.

create-base-path-mapping

The following code example shows how to use create-base-path-mapping.

AWS CLI

To create the base path mapping for a custom domain name

Command:

```
aws apigateway create-base-path-mapping --domain-name subdomain.domain.tld --rest-  
api-id 1234123412 --stage prod --base-path v1
```

- For API details, see [CreateBasePathMapping](#) in *AWS CLI Command Reference*.

create-deployment

The following code example shows how to use create-deployment.

AWS CLI

To deploy the configured resources for an API to a new Stage

Command:

```
aws apigateway create-deployment --rest-api-id 1234123412 --stage-name dev --stage-description 'Development Stage' --description 'First deployment to the dev stage'
```

To deploy the configured resources for an API to an existing stage

Command:

```
aws apigateway create-deployment --rest-api-id 1234123412 --stage-name dev --description 'Second deployment to the dev stage'
```

To deploy the configured resources for an API to an existing stage with Stage Variables

```
aws apigateway create-deployment --rest-api-id 1234123412 --stage-name dev --description 'Third deployment to the dev stage' --variables key='value',otherKey='otherValue'
```

- For API details, see [CreateDeployment](#) in *AWS CLI Command Reference*.

create-domain-name

The following code example shows how to use create-domain-name.

AWS CLI

To create the custom domain name

Command:

```
aws apigateway create-domain-name --domain-name 'my.domain.tld' --certificate-name 'my.domain.tld cert' --certificate-arn 'arn:aws:acm:us-east-1:012345678910:certificate/fb1b9770-a305-495d-aefb-27e5e101ff3'
```

- For API details, see [CreateDomainName](#) in *AWS CLI Command Reference*.

create-model

The following code example shows how to use create-model.

AWS CLI

To create a model for an API

Command:

```
aws apigateway create-model --rest-api-id 1234123412 --name 'firstModel' --description 'The First Model' --content-type 'application/json' --schema '{ "$schema": "http://json-schema.org/draft-04/schema#", "title": "firstModel", "type": "object", "properties": { "firstProperty" : { "type": "object", "properties": { "key": { "type": "string" } } } } }'
```

Output:

```
{
  "contentType": "application/json",
  "description": "The First Model",
  "name": "firstModel",
  "id": "2rzg01",
  "schema": "{ \"$schema\": \"http://json-schema.org/draft-04/schema#\", \"title\": \"firstModel\", \"type\": \"object\", \"properties\": { \"firstProperty\": { \"type\": \"object\", \"properties\": { \"key\": { \"type\": \"string\" } } } } }"
```

- For API details, see [CreateModel](#) in *AWS CLI Command Reference*.

create-resource

The following code example shows how to use `create-resource`.

AWS CLI

To create a resource in an API

Command:

```
aws apigateway create-resource --rest-api-id 1234123412 --parent-id a1b2c3 --path-part 'new-resource'
```

- For API details, see [CreateResource](#) in *AWS CLI Command Reference*.

create-rest-api

The following code example shows how to use create-rest-api.

AWS CLI

To create an API

Command:

```
aws apigateway create-rest-api --name 'My First API' --description 'This is my first API'
```

To create a duplicate API from an existing API

Command:

```
aws apigateway create-rest-api --name 'Copy of My First API' --description 'This is a copy of my first API' --clone-from 1234123412
```

- For API details, see [CreateRestApi](#) in *AWS CLI Command Reference*.

create-stage

The following code example shows how to use create-stage.

AWS CLI

To create a stage in an API which will contain an existing deployment

Command:

```
aws apigateway create-stage --rest-api-id 1234123412 --stage-name 'dev' --description 'Development stage' --deployment-id a1b2c3
```

To create a stage in an API which will contain an existing deployment and custom Stage Variables

Command:

```
aws apigateway create-stage --rest-api-id 1234123412 --stage-name 'dev'  
--description 'Development stage' --deployment-id a1b2c3 --variables  
key='value',otherKey='otherValue'
```

- For API details, see [CreateStage](#) in *AWS CLI Command Reference*.

create-usage-plan-key

The following code example shows how to use create-usage-plan-key.

AWS CLI

Associate an existing API key with a Usage Plan

Command:

```
aws apigateway create-usage-plan-key --usage-plan-id a1b2c3 --key-type "API_KEY" --  
key-id 4vq3yryqm5
```

- For API details, see [CreateUsagePlanKey](#) in *AWS CLI Command Reference*.

create-usage-plan

The following code example shows how to use create-usage-plan.

AWS CLI

To create a usage plan with throttle and quota limits that resets at the beginning of the month

Command:

```
aws apigateway create-usage-plan --name "New Usage Plan" --description "A new usage  
plan" --throttle burstLimit=10,rateLimit=5 --quota limit=500,offset=0,period=MONTH
```

- For API details, see [CreateUsagePlan](#) in *AWS CLI Command Reference*.

delete-api-key

The following code example shows how to use delete-api-key.

AWS CLI

To delete an API key

Command:

```
aws apigateway delete-api-key --api-key 8bk1k8b11k3sB38D9B310enyWT8c09B301kq0b1k
```

- For API details, see [DeleteApiKey](#) in *AWS CLI Command Reference*.

delete-authorizer

The following code example shows how to use delete-authorizer.

AWS CLI

To delete a Custom Authorizer in an API

Command:

```
aws apigateway delete-authorizer --rest-api-id 1234123412 --authorizer-id 7gkfbo
```

- For API details, see [DeleteAuthorizer](#) in *AWS CLI Command Reference*.

delete-base-path-mapping

The following code example shows how to use delete-base-path-mapping.

AWS CLI

To delete a base path mapping for a custom domain name

Command:

```
aws apigateway delete-base-path-mapping --domain-name 'api.domain.tld' --base-path 'dev'
```

- For API details, see [DeleteBasePathMapping](#) in *AWS CLI Command Reference*.

delete-client-certificate

The following code example shows how to use delete-client-certificate.

AWS CLI

To delete a client certificate

Command:

```
aws apigateway delete-client-certificate --client-certificate-id a1b2c3
```

- For API details, see [DeleteClientCertificate](#) in *AWS CLI Command Reference*.

delete-deployment

The following code example shows how to use delete-deployment.

AWS CLI

To delete a deployment in an API

Command:

```
aws apigateway delete-deployment --rest-api-id 1234123412 --deployment-id a1b2c3
```

- For API details, see [DeleteDeployment](#) in *AWS CLI Command Reference*.

delete-domain-name

The following code example shows how to use delete-domain-name.

AWS CLI

To delete a custom domain name

Command:

```
aws apigateway delete-domain-name --domain-name 'api.domain.tld'
```

- For API details, see [DeleteDomainName](#) in *AWS CLI Command Reference*.

delete-integration-response

The following code example shows how to use delete-integration-response.

AWS CLI

To delete an integration response for a given resource, method, and status code in an API

Command:

```
aws apigateway delete-integration-response --rest-api-id 1234123412 --resource-id
a1b2c3 --http-method GET --status-code 200
```

- For API details, see [DeleteIntegrationResponse](#) in *AWS CLI Command Reference*.

delete-integration

The following code example shows how to use `delete-integration`.

AWS CLI

To delete an integration for a given resource and method in an API

Command:

```
aws apigateway delete-integration --rest-api-id 1234123412 --resource-id a1b2c3 --
http-method GET
```

- For API details, see [DeleteIntegration](#) in *AWS CLI Command Reference*.

delete-method-response

The following code example shows how to use `delete-method-response`.

AWS CLI

To delete a method response for the given resource, method, and status code in an API

Command:

```
aws apigateway delete-method-response --rest-api-id 1234123412 --resource-id a1b2c3
--http-method GET --status-code 200
```

- For API details, see [DeleteMethodResponse](#) in *AWS CLI Command Reference*.

delete-method

The following code example shows how to use delete-method.

AWS CLI

To delete a method for the given resource in an API

Command:

```
aws apigateway delete-method --rest-api-id 1234123412 --resource-id a1b2c3 --http-method GET
```

- For API details, see [DeleteMethod](#) in *AWS CLI Command Reference*.

delete-model

The following code example shows how to use delete-model.

AWS CLI

To delete a model in the given API

Command:

```
aws apigateway delete-model --rest-api-id 1234123412 --model-name 'customModel'
```

- For API details, see [DeleteModel](#) in *AWS CLI Command Reference*.

delete-resource

The following code example shows how to use delete-resource.

AWS CLI

To delete a resource in an API

Command:

```
aws apigateway delete-resource --rest-api-id 1234123412 --resource-id a1b2c3
```

- For API details, see [DeleteResource](#) in *AWS CLI Command Reference*.

delete-rest-api

The following code example shows how to use `delete-rest-api`.

AWS CLI

To delete an API

Command:

```
aws apigateway delete-rest-api --rest-api-id 1234123412
```

- For API details, see [DeleteRestApi](#) in *AWS CLI Command Reference*.

delete-stage

The following code example shows how to use `delete-stage`.

AWS CLI

To delete a stage in an API

Command:

```
aws apigateway delete-stage --rest-api-id 1234123412 --stage-name 'dev'
```

- For API details, see [DeleteStage](#) in *AWS CLI Command Reference*.

delete-usage-plan-key

The following code example shows how to use `delete-usage-plan-key`.

AWS CLI

To remove an API key from a Usage Plan

Command:

```
aws apigateway delete-usage-plan-key --usage-plan-id a1b2c3 --key-id  
1NbjQzMReAkeEQPNAW8r3dXsU2rDD7fc7f2Sipnu
```

- For API details, see [DeleteUsagePlanKey](#) in *AWS CLI Command Reference*.

delete-usage-plan

The following code example shows how to use `delete-usage-plan`.

AWS CLI

To delete a Usage Plan

Command:

```
aws apigateway delete-usage-plan --usage-plan-id a1b2c3
```

- For API details, see [DeleteUsagePlan](#) in *AWS CLI Command Reference*.

flush-stage-authorizers-cache

The following code example shows how to use `flush-stage-authorizers-cache`.

AWS CLI

To flush all authorizer cache entries on a stage

Command:

```
aws apigateway flush-stage-authorizers-cache --rest-api-id 1234123412 --stage-name dev
```

- For API details, see [FlushStageAuthorizersCache](#) in *AWS CLI Command Reference*.

flush-stage-cache

The following code example shows how to use `flush-stage-cache`.

AWS CLI

To flush the cache for an API's stage

Command:

```
aws apigateway flush-stage-cache --rest-api-id 1234123412 --stage-name dev
```

- For API details, see [FlushStageCache](#) in *AWS CLI Command Reference*.

generate-client-certificate

The following code example shows how to use `generate-client-certificate`.

AWS CLI

To create a Client-Side SSL Certificate

Command:

```
aws apigateway generate-client-certificate --description 'My First Client Certificate'
```

- For API details, see [GenerateClientCertificate](#) in *AWS CLI Command Reference*.

get-account

The following code example shows how to use `get-account`.

AWS CLI

To get API Gateway account settings

Command:

```
aws apigateway get-account
```

Output:

```
{
  "cloudwatchRoleArn": "arn:aws:iam::123412341234:role/APIGatewayToCloudWatchLogsRole",
  "throttleSettings": {
    "rateLimit": 500.0,
    "burstLimit": 1000
  }
}
```

```
}
```

- For API details, see [GetAccount](#) in *AWS CLI Command Reference*.

get-api-key

The following code example shows how to use `get-api-key`.

AWS CLI

To get the information about a specific API key

Command:

```
aws apigateway get-api-key --api-key 8bk1k8b11k3sB38D9B310enyWT8c09B301kq0blk
```

Output:

```
{
  "description": "My first key",
  "enabled": true,
  "stageKeys": [
    "a1b2c3d4e5/dev",
    "e5d4c3b2a1/dev"
  ],
  "lastUpdatedDate": 1456184515,
  "createdDate": 1456184452,
  "id": "8bk1k8b11k3sB38D9B310enyWT8c09B301kq0blk",
  "name": "My key"
}
```

- For API details, see [GetApiKey](#) in *AWS CLI Command Reference*.

get-api-keys

The following code example shows how to use `get-api-keys`.

AWS CLI

To get the list of API keys

Command:

```
aws apigateway get-api-keys
```

Output:

```
{
  "items": [
    {
      "description": "My first key",
      "enabled": true,
      "stageKeys": [
        "a1b2c3d4e5/dev",
        "e5d4c3b2a1/dev"
      ],
      "lastUpdatedDate": 1456184515,
      "createdDate": 1456184452,
      "id": "8bk1k8b11k3sB38D9B310enyWT8c09B301kq0b1k",
      "name": "My key"
    }
  ]
}
```

- For API details, see [GetApiKeys](#) in *AWS CLI Command Reference*.

get-authorizer

The following code example shows how to use `get-authorizer`.

AWS CLI**To get the API Gateway per-API Authorizer settings****Command:**

```
aws apigateway get-authorizer --rest-api-id 1234123412 --authorizer-id gfi4n3
```

Output:

```
{
```

```
"authorizerResultTtlInSeconds": 300,  
"name": "MyAuthorizer",  
"type": "TOKEN",  
"identitySource": "method.request.header.Authorization",  
"authorizerUri": "arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/  
arn:aws:lambda:us-west-2:123412341234:function:authorizer_function/invocations",  
"id": "gfi4n3"  
}
```

- For API details, see [GetAuthorizer](#) in *AWS CLI Command Reference*.

get-authorizers

The following code example shows how to use `get-authorizers`.

AWS CLI

To get the list of authorizers for a REST API

Command:

```
aws apigateway get-authorizers --rest-api-id 1234123412
```

Output:

```
{  
  "items": [  
    {  
      "name": "MyAuthorizer",  
      "authorizerUri": "arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/  
functions/arn:aws:lambda:us-west-2:123412341234:function:My_Authorizer_Function/  
invocations",  
      "authorizerResultTtlInSeconds": 300,  
      "identitySource": "method.request.header.Authorization",  
      "type": "TOKEN",  
      "id": "gfi4n3"  
    }  
  ]  
}
```

- For API details, see [GetAuthorizers](#) in *AWS CLI Command Reference*.

get-base-path-mapping

The following code example shows how to use `get-base-path-mapping`.

AWS CLI

To get the base path mapping for a custom domain name

Command:

```
aws apigateway get-base-path-mapping --domain-name subdomain.domain.tld --base-path v1
```

Output:

```
{
  "basePath": "v1",
  "restApiId": "1234w4321e",
  "stage": "api"
}
```

- For API details, see [GetBasePathMapping](#) in *AWS CLI Command Reference*.

get-base-path-mappings

The following code example shows how to use `get-base-path-mappings`.

AWS CLI

To get the base path mappings for a custom domain name

Command:

```
aws apigateway get-base-path-mappings --domain-name subdomain.domain.tld
```

Output:

```
{
  "items": [
    {
```



```
    "basePath": "(none)",
    "restApiId": "1234w4321e",
    "stage": "dev"
  },
  {
    "basePath": "v1",
    "restApiId": "1234w4321e",
    "stage": "api"
  }
]
```

- For API details, see [GetBasePathMappings](#) in *AWS CLI Command Reference*.

get-client-certificate

The following code example shows how to use `get-client-certificate`.

AWS CLI

To get a client certificate

Command:

```
aws apigateway get-client-certificate --client-certificate-id a1b2c3
```

- For API details, see [GetClientCertificate](#) in *AWS CLI Command Reference*.

get-client-certificates

The following code example shows how to use `get-client-certificates`.

AWS CLI

To get a list of client certificates

Command:

```
aws apigateway get-client-certificates
```

Output:

```
{
  "items": [
    {
      "pemEncodedCertificate": "-----BEGIN CERTIFICATE----- <certificate
content> -----END CERTIFICATE-----",
      "clientCertificateId": "a1b2c3",
      "expirationDate": 1483556561,
      "description": "My Client Certificate",
      "createdDate": 1452020561
    }
  ]
}
```

- For API details, see [GetClientCertificates](#) in *AWS CLI Command Reference*.

get-deployment

The following code example shows how to use `get-deployment`.

AWS CLI

To get information about a deployment

Command:

```
aws apigateway get-deployment --rest-api-id 1234123412 --deployment-id ztt4m2
```

Output:

```
{
  "description": "myDeployment",
  "id": "ztt4m2",
  "createdDate": 1455218022
}
```

- For API details, see [GetDeployment](#) in *AWS CLI Command Reference*.

get-deployments

The following code example shows how to use `get-deployments`.

AWS CLI

To get a list of deployments for a REST API

Command:

```
aws apigateway get-deployments --rest-api-id 1234123412
```

Output:

```
{
  "items": [
    {
      "createdDate": 1453797217,
      "id": "0a2b4c",
      "description": "Deployed my API for the first time"
    }
  ]
}
```

- For API details, see [GetDeployments](#) in *AWS CLI Command Reference*.

get-domain-name

The following code example shows how to use `get-domain-name`.

AWS CLI

To get information about a custom domain name

Command:

```
aws apigateway get-domain-name --domain-name api.domain.tld
```

Output:

```
{
  "domainName": "api.domain.tld",
  "distributionDomainName": "d1a2f3a4c5o6d.cloudfront.net",
  "certificateName": "uploadedCertificate",
}
```

```
"certificateUploadDate": 1462565487
}
```

- For API details, see [GetDomainName](#) in *AWS CLI Command Reference*.

get-domain-names

The following code example shows how to use `get-domain-names`.

AWS CLI

To get a list of custom domain names

Command:

```
aws apigateway get-domain-names
```

Output:

```
{
  "items": [
    {
      "distributionDomainName": "d9511k3109bkd.cloudfront.net",
      "certificateUploadDate": 1452812505,
      "certificateName": "my_custom_domain-certificate",
      "domainName": "subdomain.domain.tld"
    }
  ]
}
```

- For API details, see [GetDomainNames](#) in *AWS CLI Command Reference*.

get-export

The following code example shows how to use `get-export`.

AWS CLI

To get the JSON Swagger template for a stage

Command:

```
aws apigateway get-export --rest-api-id a1b2c3d4e5 --stage-name dev --export-type
swagger /path/to/filename.json
```

To get the JSON Swagger template + API Gateway Extensions for a stage

Command:

```
aws apigateway get-export --parameters extensions='integrations' --rest-api-id
a1b2c3d4e5 --stage-name dev --export-type swagger /path/to/filename.json
```

To get the JSON Swagger template + Postman Extensions for a stage

Command:

```
aws apigateway get-export --parameters extensions='postman' --rest-api-id a1b2c3d4e5
--stage-name dev --export-type swagger /path/to/filename.json
```

- For API details, see [GetExport](#) in *AWS CLI Command Reference*.

get-integration-response

The following code example shows how to use `get-integration-response`.

AWS CLI

To get the integration response configuration for a HTTP method defined under a REST API's resource

Command:

```
aws apigateway get-integration-response --rest-api-id 1234123412 --resource-id
y9h6rt --http-method GET --status-code 200
```

Output:

```
{
  "statusCode": "200",
  "responseTemplates": {
    "application/json": null
  }
}
```

```
}
```

- For API details, see [GetIntegrationResponse](#) in *AWS CLI Command Reference*.

get-integration

The following code example shows how to use `get-integration`.

AWS CLI

To get the integration configuration for a HTTP method defined under a REST API's resource

Command:

```
aws apigateway get-integration --rest-api-id 1234123412 --resource-id y9h6rt --http-method GET
```

Output:

```
{
  "httpMethod": "POST",
  "integrationResponses": {
    "200": {
      "responseTemplates": {
        "application/json": null
      },
      "statusCode": "200"
    }
  },
  "cacheKeyParameters": [],
  "type": "AWS",
  "uri": "arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/arn:aws:lambda:us-west-2:123412341234:function:My_Function/invocations",
  "cacheNamespace": "y9h6rt"
}
```

- For API details, see [GetIntegration](#) in *AWS CLI Command Reference*.

get-method-response

The following code example shows how to use `get-method-response`.

AWS CLI

To get the method response resource configuration for a HTTP method defined under a REST API's resource

Command:

```
aws apigateway get-method-response --rest-api-id 1234123412 --resource-id y9h6rt --http-method GET --status-code 200
```

Output:

```
{
  "responseModels": {
    "application/json": "Empty"
  },
  "statusCode": "200"
}
```

- For API details, see [GetMethodResponse](#) in *AWS CLI Command Reference*.

get-method

The following code example shows how to use `get-method`.

AWS CLI

To get the method resource configuration for a HTTP method defined under a REST API's resource

Command:

```
aws apigateway get-method --rest-api-id 1234123412 --resource-id y9h6rt --http-method GET
```

Output:

```
{
  "apiKeyRequired": false,
  "httpMethod": "GET",
  "methodIntegration": {
    "integrationResponses": {
```

```

        "200": {
            "responseTemplates": {
                "application/json": null
            },
            "statusCode": "200"
        }
    },
    "cacheKeyParameters": [],
    "uri": "arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/arn:aws:lambda:us-west-2:123412341234:function:My_Function/invocations",
    "httpMethod": "POST",
    "cacheNamespace": "y9h6rt",
    "type": "AWS"
},
"requestParameters": {},
"methodResponses": {
    "200": {
        "responseModels": {
            "application/json": "Empty"
        },
        "statusCode": "200"
    }
},
"authorizationType": "NONE"
}

```

- For API details, see [GetMethod](#) in *AWS CLI Command Reference*.

get-model-template

The following code example shows how to use `get-model-template`.

AWS CLI

To get the mapping template for a model defined under a REST API

Command:

```
aws apigateway get-model-template --rest-api-id 1234123412 --model-name Empty
```

Output:

```
{
```



```
"value": "#set($inputRoot = $input.path('$'))\n{ }"  
}
```

- For API details, see [GetModelTemplate](#) in *AWS CLI Command Reference*.

get-model

The following code example shows how to use `get-model`.

AWS CLI

To get the configuration for a model defined under a REST API

Command:

```
aws apigateway get-model --rest-api-id 1234123412 --model-name Empty
```

Output:

```
{  
  "contentType": "application/json",  
  "description": "This is a default empty schema model",  
  "name": "Empty",  
  "id": "etd5w5",  
  "schema": "{\n  \"$schema\": \"http://json-schema.org/draft-04/schema#\",\n  \"title\" : \"Empty Schema\",\n  \"type\" : \"object\"\n}"  
}
```

- For API details, see [GetModel](#) in *AWS CLI Command Reference*.

get-models

The following code example shows how to use `get-models`.

AWS CLI

To get a list of models for a REST API

Command:

```
aws apigateway get-models --rest-api-id 1234123412
```

Output:

```
{
  "items": [
    {
      "description": "This is a default error schema model",
      "schema": "{\n  \"$schema\" : \"http://json-schema.org/draft-04/schema#\n\",\n  \"title\" : \"Error Schema\",\n  \"type\" : \"object\",\n  \"properties\" : {\n    \"message\" : { \"type\" : \"string\" }\n  }\n}",
      "contentType": "application/json",
      "id": "7tpbze",
      "name": "Error"
    },
    {
      "description": "This is a default empty schema model",
      "schema": "{\n  \"$schema\" : \"http://json-schema.org/draft-04/schema#\n\",\n  \"title\" : \"Empty Schema\",\n  \"type\" : \"object\"\n}",
      "contentType": "application/json",
      "id": "etd5w5",
      "name": "Empty"
    }
  ]
}
```

- For API details, see [GetModels](#) in *AWS CLI Command Reference*.

get-resource

The following code example shows how to use `get-resource`.

AWS CLI**To get information about a resource****Command:**

```
aws apigateway get-resource --rest-api-id 1234123412 --resource-id zwo0y3
```

Output:

```
{
```

```
"path": "/path",
"pathPart": "path",
"id": "zwo0y3",
"parentId": "uyokt6ij2g"
}
```

- For API details, see [GetResource](#) in *AWS CLI Command Reference*.

get-resources

The following code example shows how to use `get-resources`.

AWS CLI

To get a list of resources for a REST API

Command:

```
aws apigateway get-resources --rest-api-id 1234123412
```

Output:

```
{
  "items": [
    {
      "path": "/resource/subresource",
      "resourceMethods": {
        "POST": {}
      },
      "id": "024ace",
      "pathPart": "subresource",
      "parentId": "ai5b02"
    }
  ]
}
```

- For API details, see [GetResources](#) in *AWS CLI Command Reference*.

get-rest-api

The following code example shows how to use `get-rest-api`.

AWS CLI

To get information about an API

Command:

```
aws apigateway get-rest-api --rest-api-id 1234123412
```

Output:

```
{
  "name": "myAPI",
  "id": "o1y243m4f5",
  "createdDate": 1453416433
}
```

- For API details, see [GetRestApi](#) in *AWS CLI Command Reference*.

get-rest-apis

The following code example shows how to use `get-rest-apis`.

AWS CLI

To get a list of REST APIs

Command:

```
aws apigateway get-rest-apis
```

Output:

```
{
  "items": [
    {
      "createdDate": 1438884790,
      "id": "12s44z21rb",
      "name": "My First API"
    }
  ]
}
```

```
}
```

- For API details, see [GetRestApis](#) in *AWS CLI Command Reference*.

get-sdk

The following code example shows how to use `get-sdk`.

AWS CLI

To get the Android SDK for a REST API stage

Command:

```
aws apigateway get-sdk --rest-api-id 1234123412 --stage-name dev --sdk-type android
--parameters
  groupId='com.mycompany',invokerPackage='com.mycompany.clientsdk',artifactId='Mycompany-
  client',artifactVersion='1.0.0' /path/to/android_sdk.zip
```

Output:

```
{
  "contentType": "application/octet-stream",
  "contentDisposition": "attachment; filename=\"android_2016-02-22_23-52Z.zip\""
}
```

To get the IOS SDK for a REST API stage

Command:

```
aws apigateway get-sdk --rest-api-id 1234123412 --stage-name dev --sdk-type
objectivec --parameters classPrefix='myprefix' /path/to/iOS_sdk.zip
```

Output:

```
{
  "contentType": "application/octet-stream",
  "contentDisposition": "attachment; filename=\"objectivec_2016-02-22_23-52Z.zip
\""
}
```

To get the Javascript SDK for a REST API stage

Command:

```
aws apigateway get-sdk --rest-api-id 1234123412 --stage-name dev --sdk-type
javascript /path/to/javascript_sdk.zip
```

Output:

```
{
  "contentType": "application/octet-stream",
  "contentDisposition": "attachment; filename=\"javascript_2016-02-22_23-52Z.zip\"
}
```

- For API details, see [GetSdk](#) in *AWS CLI Command Reference*.

get-stage

The following code example shows how to use `get-stage`.

AWS CLI

To get information about an API's stage

Command:

```
aws apigateway get-stage --rest-api-id 1234123412 --stage-name dev
```

Output:

```
{
  "stageName": "dev",
  "cacheClusterSize": "0.5",
  "cacheClusterEnabled": false,
  "cacheClusterStatus": "NOT_AVAILABLE",
  "deploymentId": "rbh1fj",
  "lastUpdatedDate": 1466802961,
  "createdDate": 1460682074,
  "methodSettings": {
```

```

    "*/*": {
      "cacheTtlInSeconds": 300,
      "loggingLevel": "INFO",
      "dataTraceEnabled": false,
      "metricsEnabled": true,
      "unauthorizedCacheControlHeaderStrategy":
"SUCCEED_WITH_RESPONSE_HEADER",
      "throttlingRateLimit": 500.0,
      "cacheDataEncrypted": false,
      "cachingEnabled": false,
      "throttlingBurstLimit": 1000,
      "requireAuthorizationForCacheControl": true
    },
    "~1resource/GET": {
      "cacheTtlInSeconds": 300,
      "loggingLevel": "INFO",
      "dataTraceEnabled": false,
      "metricsEnabled": true,
      "unauthorizedCacheControlHeaderStrategy":
"SUCCEED_WITH_RESPONSE_HEADER",
      "throttlingRateLimit": 500.0,
      "cacheDataEncrypted": false,
      "cachingEnabled": false,
      "throttlingBurstLimit": 1000,
      "requireAuthorizationForCacheControl": true
    }
  }
}

```

- For API details, see [GetStage](#) in *AWS CLI Command Reference*.

get-stages

The following code example shows how to use `get-stages`.

AWS CLI

To get the list of stages for a REST API

Command:

```
aws apigateway get-stages --rest-api-id 1234123412
```

Output:

```
{
  "item": [
    {
      "stageName": "dev",
      "cacheClusterSize": "0.5",
      "cacheClusterEnabled": true,
      "cacheClusterStatus": "AVAILABLE",
      "deploymentId": "123h64",
      "lastUpdatedDate": 1456185138,
      "createdDate": 1453589092,
      "methodSettings": {
        "~1resource~1subresource/POST": {
          "cacheTtlInSeconds": 300,
          "loggingLevel": "INFO",
          "dataTraceEnabled": true,
          "metricsEnabled": true,
          "throttlingRateLimit": 500.0,
          "cacheDataEncrypted": false,
          "cachingEnabled": false,
          "throttlingBurstLimit": 1000
        }
      }
    }
  ]
}
```

- For API details, see [GetStages](#) in *AWS CLI Command Reference*.

get-usage-plan-key

The following code example shows how to use `get-usage-plan-key`.

AWS CLI**To get the details of an API key associated with a Usage Plan**

Command:

```
aws apigateway get-usage-plan-key --usage-plan-id a1b2c3 --key-id
1NbjQzMReAkeEQPNAW8r3dXsU2rDD7fc7f2Sipnu
```


- For API details, see [GetUsagePlanKey](#) in *AWS CLI Command Reference*.

get-usage-plan-keys

The following code example shows how to use `get-usage-plan-keys`.

AWS CLI

To get the list of API keys associated with a Usage Plan

Command:

```
aws apigateway get-usage-plan-keys --usage-plan-id a1b2c3
```

- For API details, see [GetUsagePlanKeys](#) in *AWS CLI Command Reference*.

get-usage-plan

The following code example shows how to use `get-usage-plan`.

AWS CLI

To get the details of a Usage Plan

Command:

```
aws apigateway get-usage-plan --usage-plan-id a1b2c3
```

- For API details, see [GetUsagePlan](#) in *AWS CLI Command Reference*.

get-usage-plans

The following code example shows how to use `get-usage-plans`.

AWS CLI

To get the details of all Usage Plans

Command:

```
aws apigateway get-usage-plans
```

- For API details, see [GetUsagePlans](#) in *AWS CLI Command Reference*.

get-usage

The following code example shows how to use `get-usage`.

AWS CLI

To get the usage details for a Usage Plan

Command:

```
aws apigateway get-usage --usage-plan-id a1b2c3 --start-date "2016-08-16" --end-date "2016-08-17"
```

- For API details, see [GetUsage](#) in *AWS CLI Command Reference*.

import-rest-api

The following code example shows how to use `import-rest-api`.

AWS CLI

To import a Swagger template and create an API

Command:

```
aws apigateway import-rest-api --body 'file:///path/to/API_Swagger_template.json'
```

- For API details, see [ImportRestApi](#) in *AWS CLI Command Reference*.

put-integration-response

The following code example shows how to use `put-integration-response`.

AWS CLI

To create an integration response as the default response with a mapping template defined

Command:

```
aws apigateway put-integration-response --rest-api-id 1234123412 --resource-id
a1b2c3 --http-method GET --status-code 200 --selection-pattern "" --response-
templates '{"application/json": "{\"json\": \"template\"}"}'
```

To create an integration response with a regex of 400 and a statically defined header value**Command:**

```
aws apigateway put-integration-response --rest-api-id 1234123412 --resource-id
a1b2c3 --http-method GET --status-code 400 --selection-pattern 400 --response-
parameters '{"method.response.header.custom-header": "'''''custom-value''''"}'
```

- For API details, see [PutIntegrationResponse](#) in *AWS CLI Command Reference*.

put-integration

The following code example shows how to use `put-integration`.

AWS CLI**To create a MOCK integration request****Command:**

```
aws apigateway put-integration --rest-api-id 1234123412 --resource-id a1b2c3 --http-
method GET --type MOCK --request-templates '{"application/json": "{\"statusCode\":
200}" }'
```

To create a HTTP integration request**Command:**

```
aws apigateway put-integration --rest-api-id 1234123412 --resource-id a1b2c3 --http-
method GET --type HTTP --integration-http-method GET --uri 'https://domain.tld/path'
```

To create an AWS integration request with a Lambda Function endpoint**Command:**

```
aws apigateway put-integration --rest-api-id 1234123412 --resource-id
a1b2c3 --http-method GET --type AWS --integration-http-method POST --uri
'arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/arn:aws:lambda:us-
west-2:123412341234:function:function_name/invocations'
```

- For API details, see [PutIntegration](#) in *AWS CLI Command Reference*.

put-method-response

The following code example shows how to use `put-method-response`.

AWS CLI

To create a method response under the specified status code with a custom method response header

Command:

```
aws apigateway put-method-response --rest-api-id 1234123412 --resource-
id a1b2c3 --http-method GET --status-code 400 --response-parameters
"method.response.header.custom-header=false"
```

- For API details, see [PutMethodResponse](#) in *AWS CLI Command Reference*.

put-method

The following code example shows how to use `put-method`.

AWS CLI

To create a method for a resource in an API with no authorization, no API key, and a custom method request header

Command:

```
aws apigateway put-method --rest-api-id 1234123412 --resource-id a1b2c3 --http-
method PUT --authorization-type "NONE" --no-api-key-required --request-parameters
"method.request.header.custom-header=false"
```

- For API details, see [PutMethod](#) in *AWS CLI Command Reference*.

put-rest-api

The following code example shows how to use `put-rest-api`.

AWS CLI

To overwrite an existing API using a Swagger template

Command:

```
aws apigateway put-rest-api --rest-api-id 1234123412 --mode overwrite --body
'fileb:///path/to/API_Swagger_template.json'
```

To merge a Swagger template into an existing API

Command:

```
aws apigateway put-rest-api --rest-api-id 1234123412 --mode merge --body 'fileb:///
path/to/API_Swagger_template.json'
```

- For API details, see [PutRestApi](#) in *AWS CLI Command Reference*.

test-invoke-authorizer

The following code example shows how to use `test-invoke-authorizer`.

AWS CLI

To test invoke a request to a Custom Authorizer including the required header and value

Command:

```
aws apigateway test-invoke-authorizer --rest-api-id 1234123412 --authorizer-id
5yid1t --headers Authorization='Value'
```

- For API details, see [TestInvokeAuthorizer](#) in *AWS CLI Command Reference*.

test-invoke-method

The following code example shows how to use `test-invoke-method`.

AWS CLI

To test invoke the root resource in an API by making a GET request

Command:

```
aws apigateway test-invoke-method --rest-api-id 1234123412 --resource-id av15sg8fw8
--http-method GET --path-with-query-string '/'
```

To test invoke a sub-resource in an API by making a GET request with a path parameter value specified

Command:

```
aws apigateway test-invoke-method --rest-api-id 1234123412 --resource-id 3gapai --
http-method GET --path-with-query-string '/pets/1'
```

- For API details, see [TestInvokeMethod](#) in *AWS CLI Command Reference*.

update-account

The following code example shows how to use update-account.

AWS CLI

To change the IAM Role ARN for logging to CloudWatch Logs

Command:

```
aws apigateway update-account --patch-operations op='replace',path='/
cloudwatchRoleArn',value='arn:aws:iam::123412341234:role/APIGatewayToCloudWatchLogs'
```

Output:

```
{
  "cloudwatchRoleArn": "arn:aws:iam::123412341234:role/
APIGatewayToCloudWatchLogs",
  "throttleSettings": {
    "rateLimit": 1000.0,
    "burstLimit": 2000
  }
}
```

```
}  
}
```

- For API details, see [UpdateAccount](#) in *AWS CLI Command Reference*.

update-api-key

The following code example shows how to use `update-api-key`.

AWS CLI

To change the name for an API Key

Command:

```
aws apigateway update-api-key --api-key sNvjQDMReA1eEQPNAW8r37XsU2rDD7fc7m2SiMnu --  
patch-operations op='replace',path='/name',value='newName'
```

Output:

```
{  
  "description": "currentDescription",  
  "enabled": true,  
  "stageKeys": [  
    "41t2j324r5/dev"  
  ],  
  "lastUpdatedDate": 1470086052,  
  "createdDate": 1445460347,  
  "id": "sNvjQDMReA1vEQPNzW8r3dXsU2rrD7fcjm2SiMnu",  
  "name": "newName"  
}
```

To disable the API Key

Command:

```
aws apigateway update-api-key --api-key sNvjQDMReA1eEQPNAW8r37XsU2rDD7fc7m2SiMnu --  
patch-operations op='replace',path='/enabled',value='false'
```

Output:

```
{
  "description": "currentDescription",
  "enabled": false,
  "stageKeys": [
    "41t2j324r5/dev"
  ],
  "lastUpdatedDate": 1470086052,
  "createdDate": 1445460347,
  "id": "sNvjQDMReA1vEQPNzW8r3dXsU2rrD7fcjm2SiMnu",
  "name": "newName"
}
```

- For API details, see [UpdateApiKey](#) in *AWS CLI Command Reference*.

update-authorizer

The following code example shows how to use `update-authorizer`.

AWS CLI

To change the name of the Custom Authorizer

Command:

```
aws apigateway update-authorizer --rest-api-id 1234123412 --authorizer-id gfi4n3 --
patch-operations op='replace',path='/name',value='testAuthorizer'
```

Output:

```
{
  "authType": "custom",
  "name": "testAuthorizer",
  "authorizerUri": "arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-west-2:123412341234:function:customAuthorizer/invocations",
  "authorizerResultTtlInSeconds": 300,
  "identitySource": "method.request.header.Authorization",
  "type": "TOKEN",
  "id": "gfi4n3"
}
```

To change the Lambda Function that is invoked by the Custom Authorizer

Command:

```
aws apigateway update-authorizer --rest-api-id 1234123412 --authorizer-id gfi4n3 --
patch-operations op='replace',path='/authorizerUri',value='arn:aws:apigateway:us-
west-2:lambda:path/2015-03-31/functions/arn:aws:lambda:us-
west-2:123412341234:function:newAuthorizer/invocations'
```

Output:

```
{
  "authType": "custom",
  "name": "testAuthorizer",
  "authorizerUri": "arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-west-2:123412341234:function:newAuthorizer/invocations",
  "authorizerResultTtlInSeconds": 300,
  "identitySource": "method.request.header.Authorization",
  "type": "TOKEN",
  "id": "gfi4n3"
}
```

- For API details, see [UpdateAuthorizer](#) in *AWS CLI Command Reference*.

update-base-path-mapping

The following code example shows how to use update-base-path-mapping.

AWS CLI**To change the base path for a custom domain name****Command:**

```
aws apigateway update-base-path-mapping --domain-name api.domain.tld --base-path
prod --patch-operations op='replace',path='/basePath',value='v1'
```

Output:

```
{
  "basePath": "v1",
  "restApiId": "1234123412",
  "stage": "api"
}
```

```
}
```

- For API details, see [UpdateBasePathMapping](#) in *AWS CLI Command Reference*.

update-client-certificate

The following code example shows how to use `update-client-certificate`.

AWS CLI

To update the description of a client certificate

Command:

```
aws apigateway update-client-certificate --client-certificate-id a1b2c3 --patch-operations op='replace',path='/description',value='My new description'
```

- For API details, see [UpdateClientCertificate](#) in *AWS CLI Command Reference*.

update-deployment

The following code example shows how to use `update-deployment`.

AWS CLI

To change the description of a deployment

Command:

```
aws apigateway update-deployment --rest-api-id 1234123412 --deployment-id ztt4m2 --patch-operations op='replace',path='/description',value='newDescription'
```

Output:

```
{
  "description": "newDescription",
  "id": "ztt4m2",
  "createdDate": 1455218022
}
```

- For API details, see [UpdateDeployment](#) in *AWS CLI Command Reference*.

update-domain-name

The following code example shows how to use `update-domain-name`.

AWS CLI

To change the certificate name for a custom domain name

The following `update-domain-name` example changes the certificate name for a custom domain.

```
aws apigateway update-domain-name \  
  --domain-name api.domain.tld \  
  --patch-operations op='replace',path='/certificateArn',value='arn:aws:acm:us-  
west-2:111122223333:certificate/CERTEXAMPLE123EXAMPLE'
```

Output:

```
{  
  "domainName": "api.domain.tld",  
  "distributionDomainName": "d123456789012.cloudfront.net",  
  "certificateArn": "arn:aws:acm:us-west-2:111122223333:certificate/  
CERTEXAMPLE123EXAMPLE",  
  "certificateUploadDate": 1462565487  
}
```

For more information, see [Set up Custom Domain Name for an API in API Gateway](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [UpdateDomainName](#) in *AWS CLI Command Reference*.

update-integration-response

The following code example shows how to use `update-integration-response`.

AWS CLI

To change an integration response header to have a static mapping of '*'

Command:

```
aws apigateway update-integration-response --rest-api-id 1234123412 --  
resource-id 3gapai --http-method GET --status-code 200 --patch-operations
```

```
op='replace',path='/responseParameters/method.response.header.Access-Control-Allow-Origin',value='''''*''''''
```

Output:

```
{
  "statusCode": "200",
  "responseParameters": {
    "method.response.header.Access-Control-Allow-Origin": ""*""
  }
}
```

To remove an integration response header

Command:

```
aws apigateway update-integration-response --rest-api-id 1234123412 --resource-id 3gapai --http-method GET --status-code 200 --patch-operations op='remove',path='/responseParameters/method.response.header.Access-Control-Allow-Origin'
```

- For API details, see [UpdateIntegrationResponse](#) in *AWS CLI Command Reference*.

update-integration

The following code example shows how to use update-integration.

AWS CLI

To add the 'Content-Type: application/json' Mapping Template configured with Input Passthrough

Command:

```
aws apigateway update-integration \  
  --rest-api-id a1b2c3d4e5 \  
  --resource-id a1b2c3 \  
  --http-method POST \  
  --patch-operations "op='add',path='/requestTemplates/application~1json'"
```

To update (replace) the 'Content-Type: application/json' Mapping Template configured with a custom template

Command:

```
aws apigateway update-integration \  
  --rest-api-id a1b2c3d4e5 \  
  --resource-id a1b2c3 \  
  --http-method POST \  
  --patch-operations "op='replace',path='/requestTemplates/  
application~1json',value='{\"example\": \"json\"}'"
```

To update (replace) a custom template associated with 'Content-Type: application/json' with Input Passthrough**Command:**

```
aws apigateway update-integration \  
  --rest-api-id a1b2c3d4e5 \  
  --resource-id a1b2c3 \  
  --http-method POST \  
  --patch-operations "op='replace',path='requestTemplates/application~1json'"
```

To remove the 'Content-Type: application/json' Mapping Template**Command:**

```
aws apigateway update-integration \  
  --rest-api-id a1b2c3d4e5 \  
  --resource-id a1b2c3 \  
  --http-method POST \  
  --patch-operations "op='remove',path='/requestTemplates/application~1json'"
```

- For API details, see [UpdateIntegration](#) in *AWS CLI Command Reference*.

update-method-response

The following code example shows how to use `update-method-response`.

AWS CLI**To create a new method response header for the 200 response in a method and define it as not required (default)**

Command:

```
aws apigateway update-method-response --rest-api-id 1234123412 --resource-id
a1b2c3 --http-method GET --status-code 200 --patch-operations op="add",path="/
responseParameters/method.response.header.custom-header",value="false"
```

To delete a response model for the 200 response in a method**Command:**

```
aws apigateway update-method-response --rest-api-id 1234123412 --resource-id
a1b2c3 --http-method GET --status-code 200 --patch-operations op="remove",path="/
responseModels/application~1json"
```

- For API details, see [UpdateMethodResponse](#) in *AWS CLI Command Reference*.

update-method

The following code example shows how to use `update-method`.

AWS CLI**Example 1: To modify a method to require an API key**

The following `update-method` example modifies the method to require an API key.

```
aws apigateway update-method \  
  --rest-api-id 1234123412 \  
  --resource-id a1b2c3 \  
  --http-method GET \  
  --patch-operations op="replace",path="/apiKeyRequired",value="true"
```

Output:

```
{  
  "httpMethod": "GET",  
  "authorizationType": "NONE",  
  "apiKeyRequired": true,  
  "methodResponses": {  
    "200": {  
      "statusCode": "200",
```

```

        "responseModels": {}
    }
},
"methodIntegration": {
    "type": "AWS",
    "httpMethod": "POST",
    "uri": "arn:aws:apigateway:us-east-1:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-east-1:123456789111:function:hello-world/invocations",
    "passthroughBehavior": "WHEN_NO_MATCH",
    "contentHandling": "CONVERT_TO_TEXT",
    "timeoutInMillis": 29000,
    "cacheNamespace": "h7i8j9",
    "cacheKeyParameters": [],
    "integrationResponses": {
        "200": {
            "statusCode": "200",
            "responseTemplates": {}
        }
    }
}
}
}

```

Example 2: To modify a method to require IAM authorization

The following update-method example modifies the method to require IAM authorization.

```

aws apigateway update-method \
  --rest-api-id 1234123412 \
  --resource-id a1b2c3 \
  --http-method GET \
  --patch-operations op="replace",path="/authorizationType",value="AWS_IAM"

```

Output:

```

{
  "httpMethod": "GET",
  "authorizationType": "AWS_IAM",
  "apiKeyRequired": false,
  "methodResponses": {
    "200": {
      "statusCode": "200",
      "responseModels": {}
    }
  }
}

```

```

    },
    "methodIntegration": {
      "type": "AWS",
      "httpMethod": "POST",
      "uri": "arn:aws:apigateway:us-east-1:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-east-1:123456789111:function:hello-world/invocations",
      "passthroughBehavior": "WHEN_NO_MATCH",
      "contentHandling": "CONVERT_TO_TEXT",
      "timeoutInMillis": 29000,
      "cacheNamespace": "h7i8j9",
      "cacheKeyParameters": [],
      "integrationResponses": {
        "200": {
          "statusCode": "200",
          "responseTemplates": {}
        }
      }
    }
  }
}

```

Example 3: To modify a method to require Lambda authorization

The following `update-method` example modifies the method to required Lambda authorization.

```

aws apigateway update-method --rest-api-id 1234123412 \
  --resource-id a1b2c3 \
  --http-method GET \
  --patch-operations op="replace",path="/authorizationType",value="CUSTOM"
op="replace",path="/authorizerId",value="e4f5g6"

```

Output:

```

{
  "httpMethod": "GET",
  "authorizationType": "CUSTOM",
  "authorizerId" : "e4f5g6",
  "apiKeyRequired": false,
  "methodResponses": {
    "200": {
      "statusCode": "200",
      "responseModels": {}
    }
  }
}

```



```

    },
    "methodIntegration": {
      "type": "AWS",
      "httpMethod": "POST",
      "uri": "arn:aws:apigateway:us-east-1:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-east-1:123456789111:function:hello-world/invocations",
      "passthroughBehavior": "WHEN_NO_MATCH",
      "contentHandling": "CONVERT_TO_TEXT",
      "timeoutInMillis": 29000,
      "cacheNamespace": "h7i8j9",
      "cacheKeyParameters": [],
      "integrationResponses": {
        "200": {
          "statusCode": "200",
          "responseTemplates": {}
        }
      }
    }
  }
}

```

For more information, see [Create, configure, and test usage plans using the API Gateway CLI and REST API](#) and [Controlling and managing access to a REST API in API Gateway](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [UpdateMethod](#) in *AWS CLI Command Reference*.

update-model

The following code example shows how to use `update-model`.

AWS CLI

To change the description of a model in an API

Command:

```
aws apigateway update-model --rest-api-id 1234123412 --model-name 'Empty' --patch-operations op=replace,path=/description,value='New Description'
```

To change the schema of a model in an API

Command:

```
aws apigateway update-model --rest-api-id 1234123412 --model-name 'Empty' --patch-operations op=replace,path=/schema,value='{ \"$schema\": \"http://json-schema.org/draft-04/schema#\", \"title\": \"Empty Schema\", \"type\": \"object\" }''
```

- For API details, see [UpdateModel](#) in *AWS CLI Command Reference*.

update-resource

The following code example shows how to use `update-resource`.

AWS CLI

To move a resource and place it under a different parent resource in an API

Command:

```
aws apigateway update-resource --rest-api-id 1234123412 --resource-id 1a2b3c --patch-operations op=replace,path=/parentId,value='3c2b1a'
```

Output:

```
{
  "path": "/resource",
  "pathPart": "resource",
  "id": "1a2b3c",
  "parentId": "3c2b1a"
}
```

To rename a resource (pathPart) in an API

Command:

```
aws apigateway update-resource --rest-api-id 1234123412 --resource-id 1a2b3c --patch-operations op=replace,path=/pathPart,value=newresourcename
```

Output:

```
{
  "path": "/newresourcename",
  "pathPart": "newresourcename",
}
```

```
"id": "1a2b3c",  
"parentId": "3c2b1a"  
}
```

- For API details, see [UpdateResource](#) in *AWS CLI Command Reference*.

update-rest-api

The following code example shows how to use `update-rest-api`.

AWS CLI

To change the name of an API

Command:

```
aws apigateway update-rest-api --rest-api-id 1234123412 --patch-operations  
op=replace,path=/name,value='New Name'
```

To change the description of an API

Command:

```
aws apigateway update-rest-api --rest-api-id 1234123412 --patch-operations  
op=replace,path=/description,value='New Description'
```

- For API details, see [UpdateRestApi](#) in *AWS CLI Command Reference*.

update-stage

The following code example shows how to use `update-stage`.

AWS CLI

Example 1: To override the stage settings for a resource and method

The following `update-stage` example overrides the stage settings and turns off full request/response logging for a specific resource and method.

```
aws apigateway update-stage \  

```

```
--rest-api-id 1234123412 \  
--stage-name 'dev' \  
--patch-operations op=replace,path=~1resourceName/GET/logging/  
dataTrace,value=false
```

Output:

```
{  
  "deploymentId": "5ubd17",  
  "stageName": "dev",  
  "cacheClusterEnabled": false,  
  "cacheClusterStatus": "NOT_AVAILABLE",  
  "methodSettings": {  
    "~1resourceName/GET": {  
      "metricsEnabled": false,  
      "dataTraceEnabled": false,  
      "throttlingBurstLimit": 5000,  
      "throttlingRateLimit": 10000.0,  
      "cachingEnabled": false,  
      "cacheTtlInSeconds": 300,  
      "cacheDataEncrypted": false,  
      "requireAuthorizationForCacheControl": true,  
      "unauthorizedCacheControlHeaderStrategy": "SUCCEED_WITH_RESPONSE_HEADER"  
    }  
  },  
  "tracingEnabled": false,  
  "createdDate": "2022-07-18T10:11:18-07:00",  
  "lastUpdatedDate": "2022-07-18T10:19:04-07:00"  
}
```

For more information, see [Setting up a stage for a REST API](#) in the *Amazon API Gateway Developer Guide*.

Example 2: To update the stage settings for all resources and methods of an API stage

The following update-stage example turns on full request/response logging for all resources and methods of an API stage.

```
aws apigateway update-stage \  
  --rest-api-id 1234123412 \  
  --stage-name 'dev' \  
  --patch-operations 'op=replace,path=/**/logging/dataTrace,value=true'
```

Output:

```
{
  "deploymentId": "5ubd17",
  "stageName": "dev",
  "cacheClusterEnabled": false,
  "cacheClusterStatus": "NOT_AVAILABLE",
  "methodSettings": {
    "/*/*": {
      "metricsEnabled": false,
      "dataTraceEnabled": true,
      "throttlingBurstLimit": 5000,
      "throttlingRateLimit": 10000.0,
      "cachingEnabled": false,
      "cacheTtlInSeconds": 300,
      "cacheDataEncrypted": false,
      "requireAuthorizationForCacheControl": true,
      "unauthorizedCacheControlHeaderStrategy": "SUCCEED_WITH_RESPONSE_HEADER"
    }
  },
  "tracingEnabled": false,
  "createdDate": "2022-07-18T10:11:18-07:00",
  "lastUpdatedDate": "2022-07-18T10:31:04-07:00"
}
```

For more information, see [Setting up a stage for a REST API](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [UpdateStage](#) in *AWS CLI Command Reference*.

update-usage-plan

The following code example shows how to use `update-usage-plan`.

AWS CLI**To change the period defined in a Usage Plan**

Command:

```
aws apigateway update-usage-plan --usage-plan-id a1b2c3 --patch-operations
  op="replace",path="/quota/period",value="MONTH"
```

To change the quota limit defined in a Usage Plan

Command:

```
aws apigateway update-usage-plan --usage-plan-id a1b2c3 --patch-operations
op="replace",path="/quota/limit",value="500"
```

To change the throttle rate limit defined in a Usage Plan

Command:

```
aws apigateway update-usage-plan --usage-plan-id a1b2c3 --patch-operations
op="replace",path="/throttle/rateLimit",value="10"
```

To change the throttle burst limit defined in a Usage Plan

Command:

```
aws apigateway update-usage-plan --usage-plan-id a1b2c3 --patch-operations
op="replace",path="/throttle/burstLimit",value="20"
```

- For API details, see [UpdateUsagePlan](#) in *AWS CLI Command Reference*.

update-usage

The following code example shows how to use `update-usage`.

AWS CLI

To temporarily modify the quota on an API key for the current period defined in the Usage Plan

Command:

```
aws apigateway update-usage --usage-plan-id a1b2c3 --key-id
1NbjQzMReAkeEQPNAW8r3dXsU2rDD7fc7f2Sipnu --patch-operations op="replace",path="/
remaining",value="50"
```

- For API details, see [UpdateUsage](#) in *AWS CLI Command Reference*.

API Gateway HTTP and WebSocket API examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with API Gateway HTTP and WebSocket API.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-api-mapping

The following code example shows how to use `create-api-mapping`.

AWS CLI

To create an API mapping for an API

The following `create-api-mapping` example maps the `test` stage of an API to the `/myApi` path of the `regional.example.com` custom domain name.

```
aws apigatewayv2 create-api-mapping \  
  --domain-name regional.example.com \  
  --api-mapping-key myApi \  
  --api-id a1b2c3d4 \  
  --stage test
```

Output:

```
{  
  "ApiId": "a1b2c3d4",
```

```
"ApiMappingId": "0qzs2sy7bh",
"ApiMappingKey": "myApi"
"Stage": "test"
}
```

For more information, see [Setting up a regional custom domain name in API Gateway](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [CreateApiMapping](#) in *AWS CLI Command Reference*.

create-api

The following code example shows how to use `create-api`.

AWS CLI

To create an HTTP API

The following `create-api` example creates an HTTP API by using quick create. You can use quick create to create an API with an AWS Lambda or HTTP integration, a default catch-all route, and a default stage that is configured to automatically deploy changes. The following command uses quick create to create an HTTP API that integrates with a Lambda function.

```
aws apigatewayv2 create-api \
  --name my-http-api \
  --protocol-type HTTP \
  --target arn:aws:lambda:us-west-2:123456789012:function:my-lambda-function
```

Output:

```
{
  "ApiEndpoint": "https://a1b2c3d4.execute-api.us-west-2.amazonaws.com",
  "ApiId": "a1b2c3d4",
  "ApiKeySelectionExpression": "$request.header.x-api-key",
  "CreateDate": "2020-04-08T19:05:45+00:00",
  "Name": "my-http-api",
  "ProtocolType": "HTTP",
  "RouteSelectionExpression": "$request.method $request.path"
}
```

For more information, see [Developing an HTTP API in API Gateway](#) in the *Amazon API Gateway Developer Guide*.

To create a WebSocket API

The following `create-api` example creates a WebSocket API with the specified name.

```
aws apigatewayv2 create-api \  
  --name "myWebSocketApi" \  
  --protocol-type WEBSOCKET \  
  --route-selection-expression '$request.body.action'
```

Output:

```
{  
  "ApiKeySelectionExpression": "$request.header.x-api-key",  
  "Name": "myWebSocketApi",  
  "CreateDate": "2018-11-15T06:23:51Z",  
  "ProtocolType": "WEBSOCKET",  
  "RouteSelectionExpression": "'$request.body.action'",  
  "ApiId": "aabbccdee"  
}
```

For more information, see [Create a WebSocket API in API Gateway](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [CreateApi](#) in *AWS CLI Command Reference*.

create-authorizer

The following code example shows how to use `create-authorizer`.

AWS CLI

To create a JWT authorizer for an HTTP API

The following `create-authorizer` example creates a JWT authorizer that uses Amazon Cognito as an identity provider.

```
aws apigatewayv2 create-authorizer \  
  --name my-jwt-authorizer \  
  --api-id a1b2c3d4 \  
  --authorizer-type JWT \  
  --identity-source '$request.header.Authorization' \  
  --
```

```
--jwt-configuration Audience=123456abc,Issuer=https://cognito-idp.us-west-2.amazonaws.com/us-west-2_abc123
```

Output:

```
{
  "AuthorizerId": "a1b2c3",
  "AuthorizerType": "JWT",
  "IdentitySource": [
    "$request.header.Authorization"
  ],
  "JwtConfiguration": {
    "Audience": [
      "123456abc"
    ],
    "Issuer": "https://cognito-idp.us-west-2.amazonaws.com/us-west-2_abc123"
  },
  "Name": "my-jwt-authorizer"
}
```

For more information, see [Controlling access to HTTP APIs with JWT authorizers](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [CreateAuthorizer](#) in *AWS CLI Command Reference*.

create-deployment

The following code example shows how to use `create-deployment`.

AWS CLI

To create a deployment for an API

The following `create-deployment` example creates a deployment for an API and associates that deployment with the `dev` stage of the API.

```
aws apigatewayv2 create-deployment \  
  --api-id a1b2c3d4 \  
  --stage-name dev
```

Output:

```
{
  "AutoDeployed": false,
  "CreateDate": "2020-04-06T23:38:08Z",
  "DeploymentId": "531z91",
  "DeploymentStatus": "DEPLOYED"
}
```

For more information, see [API deployment](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [CreateDeployment](#) in *AWS CLI Command Reference*.

create-domain-name

The following code example shows how to use `create-domain-name`.

AWS CLI

To create a custom domain name

The following `create-domain-name` example creates a regional custom domain name for an API.

```
aws apigatewayv2 create-domain-name \
  --domain-name regional.example.com \
  --domain-name-configurations CertificateArn=arn:aws:acm:us-
west-2:123456789012:certificate/123456789012-1234-1234-1234-12345678
```

Output:

```
{
  "ApiMappingSelectionExpression": "$request.basepath",
  "DomainName": "regional.example.com",
  "DomainNameConfigurations": [
    {
      "ApiGatewayDomainName": "d-id.execute-api.us-west-2.amazonaws.com",
      "CertificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/123456789012-1234-1234-1234-12345678",
      "EndpointType": "REGIONAL",
      "HostedZoneId": "123456789111",
      "SecurityPolicy": "TLS_1_2",
      "DomainNameStatus": "AVAILABLE"
    }
  ]
}
```

```
]
}
```

For more information, see [Setting up a regional custom domain name in API Gateway](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [CreateDomainName](#) in *AWS CLI Command Reference*.

create-integration

The following code example shows how to use create-integration.

AWS CLI

To create a WebSocket API integration

The following create-integration example creates a mock integration for a WebSocket API.

```
aws apigatewayv2 create-integration \  
  --api-id aabbccdde \\  
  --passthrough-behavior WHEN_NO_MATCH \  
  --timeout-in-millis 29000 \  
  --connection-type INTERNET \  
  --integration-type MOCK
```

Output:

```
{  
  "ConnectionType": "INTERNET",  
  "IntegrationId": "0abcdef",  
  "IntegrationResponseSelectionExpression": "${integration.response.statuscode}",  
  "IntegrationType": "MOCK",  
  "PassthroughBehavior": "WHEN_NO_MATCH",  
  "PayloadFormatVersion": "1.0",  
  "TimeoutInMillis": 29000  
}
```

For more information, see [Set up a WebSocket API integration request in API Gateway](#) in the *Amazon API Gateway Developer Guide*.

To create an HTTP API integration

The following `create-integration` example creates an AWS Lambda integration for an HTTP API.

```
aws apigatewayv2 create-integration \  
  --api-id a1b2c3d4 \  
  --integration-type AWS_PROXY \  
  --integration-uri arn:aws:lambda:us-west-2:123456789012:function:my-function \  
  --payload-format-version 2.0
```

Output:

```
{  
  "ConnectionType": "INTERNET",  
  "IntegrationId": "0abcdef",  
  "IntegrationMethod": "POST",  
  "IntegrationType": "AWS_PROXY",  
  "IntegrationUri": "arn:aws:lambda:us-west-2:123456789012:function:my-function",  
  "PayloadFormatVersion": "2.0",  
  "TimeoutInMillis": 30000  
}
```

For more information, see [Configuring integrations for HTTP APIs](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [CreateIntegration](#) in *AWS CLI Command Reference*.

create-route

The following code example shows how to use `create-route`.

AWS CLI

To create a `$default` route for a WebSocket or HTTP API

The following `create-route` example creates a `$default` route for a WebSocket or HTTP API.

```
aws apigatewayv2 create-route \  
  --api-id aabbccddeee \  
  --route-key '$default'
```

Output:

```
{
  "ApiKeyRequired": false,
  "AuthorizationType": "NONE",
  "RouteKey": "$default",
  "RouteId": "1122334"
}
```

For more information, see [Working with routes for WebSocket APIs](#) in the *Amazon API Gateway Developer Guide*

To create a route for an HTTP API

The following `create-route` example creates a route named `signup` that accepts POST requests.

```
aws apigatewayv2 create-route \
  --api-id aabbccdde \
  --route-key 'POST /signup'
```

Output:

```
{
  "ApiKeyRequired": false,
  "AuthorizationType": "NONE",
  "RouteKey": "POST /signup",
  "RouteId": "1122334"
}
```

For more information, see [Working with routes for HTTP APIs](#) in the *Amazon API Gateway Developer Guide*

- For API details, see [CreateRoute](#) in *AWS CLI Command Reference*.

create-stage

The following code example shows how to use `create-stage`.

AWS CLI**To create a stage**

The following `create-stage` example creates a stage named `dev` for an API.

```
aws apigatewayv2 create-stage \  
  --api-id a1b2c3d4 \  
  --stage-name dev
```

Output:

```
{  
  "CreateDate": "2020-04-06T23:23:46Z",  
  "DefaultRouteSettings": {  
    "DetailedMetricsEnabled": false  
  },  
  "LastUpdatedDate": "2020-04-06T23:23:46Z",  
  "RouteSettings": {},  
  "StageName": "dev",  
  "StageVariables": {},  
  "Tags": {}  
}
```

For more information, see [Working with stages for HTTP APIs](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [CreateStage](#) in *AWS CLI Command Reference*.

create-vpc-link

The following code example shows how to use `create-vpc-link`.

AWS CLI

To create a VPC link for an HTTP API

The following `create-vpc-link` example creates a VPC link for HTTP APIs.

```
aws apigatewayv2 create-vpc-link \  
  --name MyVpcLink \  
  --subnet-ids subnet-aaaa subnet-bbbb \  
  --security-group-ids sg1234 sg5678
```

Output:

```
{
  "CreateDate": "2020-04-07T00:11:46Z",
  "Name": "MyVpcLink",
  "SecurityGroupIds": [
    "sg1234",
    "sg5678"
  ],
  "SubnetIds": [
    "subnet-aaaa",
    "subnet-bbbb"
  ],
  "Tags": {},
  "VpcLinkId": "abcd123",
  "VpcLinkStatus": "PENDING",
  "VpcLinkStatusMessage": "VPC link is provisioning ENIs",
  "VpcLinkVersion": "V2"
}
```

For more information, see [Working with VPC links for HTTP APIs](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [CreateVpcLink](#) in *AWS CLI Command Reference*.

delete-access-log-settings

The following code example shows how to use `delete-access-log-settings`.

AWS CLI

To disable access logging for an API

The following `delete-access-log-settings` example deletes the access log settings for the `$default` stage of an API. To disable access logging for a stage, delete its access log settings.

```
aws apigatewayv2 delete-access-log-settings \
  --api-id a1b2c3d4 \
  --stage-name '$default'
```

This command produces no output.

For more information, see [Configuring logging for an HTTP API](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [DeleteAccessLogSettings](#) in *AWS CLI Command Reference*.

delete-api-mapping

The following code example shows how to use `delete-api-mapping`.

AWS CLI

To delete an API mapping

The following `delete-api-mapping` example deletes an API mapping for the `api.example.com` custom domain name.

```
aws apigatewayv2 delete-api-mapping \  
  --api-mapping-id a1b2c3 \  
  --domain-name api.example.com
```

This command produces no output.

For more information, see [Setting up a regional custom domain name in API Gateway](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [DeleteApiMapping](#) in *AWS CLI Command Reference*.

delete-api

The following code example shows how to use `delete-api`.

AWS CLI

To delete an API

The following `delete-api` example deletes an API.

```
aws apigatewayv2 delete-api \  
  --api-id a1b2c3d4
```

This command produces no output.

For more information, see [Working with HTTP APIs](#) and [Working with WebSocket APIs](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [DeleteApi](#) in *AWS CLI Command Reference*.

delete-authorizer

The following code example shows how to use `delete-authorizer`.

AWS CLI

To delete an authorizer

The following `delete-authorizer` example deletes an authorizer.

```
aws apigatewayv2 delete-authorizer \  
  --api-id a1b2c3d4 \  
  --authorizer-id a1b2c3
```

This command produces no output.

For more information, see [Controlling access to HTTP APIs with JWT authorizers](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [DeleteAuthorizer](#) in *AWS CLI Command Reference*.

delete-cors-configuration

The following code example shows how to use `delete-cors-configuration`.

AWS CLI

To delete the CORS configuration for an HTTP API

The following `delete-cors-configuration` example disables CORS for an HTTP API by deleting its CORS configuration.

```
aws apigatewayv2 delete-cors-configuration \  
  --api-id a1b2c3d4
```

This command produces no output.

For more information, see [Configuring CORS for an HTTP API](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [DeleteCorsConfiguration](#) in *AWS CLI Command Reference*.

delete-deployment

The following code example shows how to use delete-deployment.

AWS CLI

To delete a deployment

The following delete-deployment example deletes a deployment of an API.

```
aws apigatewayv2 delete-deployment \  
  --api-id a1b2c3d4 \  
  --deployment-id a1b2c3
```

This command produces no output.

For more information, see [API deployment](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [DeleteDeployment](#) in *AWS CLI Command Reference*.

delete-domain-name

The following code example shows how to use delete-domain-name.

AWS CLI

To delete a custom domain name

The following delete-domain-name example deletes a custom domain name.

```
aws apigatewayv2 delete-domain-name \  
  --domain-name api.example.com
```

This command produces no output.

For more information, see [Setting up a regional custom domain name in API Gateway](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [DeleteDomainName](#) in *AWS CLI Command Reference*.

delete-integration

The following code example shows how to use delete-integration.

AWS CLI

To delete an integration

The following delete-integration example deletes an API integration.

```
aws apigatewayv2 delete-integration \  
  --api-id a1b2c3d4 \  
  --integration-id a1b2c3
```

This command produces no output.

For more information, see [Configuring integrations for HTTP APIs](#) and [Setting up WebSocket API integrations](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [DeleteIntegration](#) in *AWS CLI Command Reference*.

delete-route-settings

The following code example shows how to use delete-route-settings.

AWS CLI

To delete route settings

The following delete-route-settings example deletes the route settings for the specified route.

```
aws apigatewayv2 delete-route-settings \  
  --api-id a1b2c3d4 \  
  --stage-name dev \  
  --route-key 'GET /pets'
```

This command produces no output.

For more information, see [Working with routes for HTTP APIs](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [DeleteRouteSettings](#) in *AWS CLI Command Reference*.

delete-route

The following code example shows how to use `delete-route`.

AWS CLI

To delete a route

The following `delete-route` example deletes an API route.

```
aws apigatewayv2 delete-route \  
  --api-id a1b2c3d4 \  
  --route-id a1b2c3
```

This command produces no output.

For more information, see [Working with routes for HTTP APIs](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [DeleteRoute](#) in *AWS CLI Command Reference*.

delete-stage

The following code example shows how to use `delete-stage`.

AWS CLI

To delete a stage

The following `delete-stage` example deletes the test stage of an API.

```
aws apigatewayv2 delete-stage \  
  --api-id a1b2c3d4 \  
  --stage-name test
```

This command produces no output.

For more information, see [Working with stages for HTTP APIs](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [DeleteStage](#) in *AWS CLI Command Reference*.

delete-vpc-link

The following code example shows how to use `delete-vpc-link`.

AWS CLI

To delete a VPC link for an HTTP API

The following `delete-vpc-link` example deletes a VPC link.

```
aws apigatewayv2 delete-vpc-link \  
  --vpc-link-id abcd123
```

This command produces no output.

For more information, see [Working with VPC links for HTTP APIs](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [DeleteVpcLink](#) in *AWS CLI Command Reference*.

export-api

The following code example shows how to use `export-api`.

AWS CLI

To export an OpenAPI definition of an HTTP API

The following `export-api` example exports an OpenAPI 3.0 definition of an API stage named `prod` to a YAML file named `stage-definition.yaml`. The exported definition file includes API Gateway extensions by default.

```
aws apigatewayv2 export-api \  
  --api-id a1b2c3d4 \  
  --output-type YAML \  
  --specification OAS30 \  
  --stage-name prod \  
  stage-definition.yaml
```

This command produces no output.

For more information, see [Exporting an HTTP API from API Gateway](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [ExportApi](#) in *AWS CLI Command Reference*.

get-api-mapping

The following code example shows how to use `get-api-mapping`.

AWS CLI

To get information about an API mapping for a custom domain name

The following `get-api-mapping` example displays information about an API mapping for the `api.example.com` custom domain name.

```
aws apigatewayv2 get-api-mapping \  
  --api-mapping-id a1b2c3 \  
  --domain-name api.example.com
```

Output:

```
{  
  "ApiId": "a1b2c3d4",  
  "ApiMappingId": "a1b2c3d5",  
  "ApiMappingKey": "myTestApi"  
  "Stage": "test"  
}
```

For more information, see [Setting up a regional custom domain name in API Gateway](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [GetApiMapping](#) in *AWS CLI Command Reference*.

get-api-mappings

The following code example shows how to use `get-api-mappings`.

AWS CLI

To get API mappings for a custom domain name

The following `get-api-mappings` example displays a list of all of the API mappings for the `api.example.com` custom domain name.

```
aws apigatewayv2 get-api-mappings \
  --domain-name api.example.com
```

Output:

```
{
  "Items": [
    {
      "ApiId": "a1b2c3d4",
      "ApiMappingId": "a1b2c3d5",
      "ApiMappingKey": "myTestApi"
      "Stage": "test"
    },
    {
      "ApiId": "a5b6c7d8",
      "ApiMappingId": "a1b2c3d6",
      "ApiMappingKey": "myDevApi"
      "Stage": "dev"
    },
  ],
}
```

For more information, see [Setting up a regional custom domain name in API Gateway](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [GetApiMappings](#) in *AWS CLI Command Reference*.

get-api

The following code example shows how to use `get-api`.

AWS CLI

To retrieve information about an API

The following `get-api` example displays information about an API.

```
aws apigatewayv2 get-api \
```



```
--api-id a1b2c3d4
```

Output:

```
{
  "ApiEndpoint": "https://a1b2c3d4.execute-api.us-west-2.amazonaws.com",
  "ApiId": "a1b2c3d4",
  "ApiKeySelectionExpression": "$request.header.x-api-key",
  "CreateDate": "2020-03-28T00:32:37Z",
  "Name": "my-api",
  "ProtocolType": "HTTP",
  "RouteSelectionExpression": "$request.method $request.path",
  "Tags": {
    "department": "finance"
  }
}
```

- For API details, see [GetApi](#) in *AWS CLI Command Reference*.

get-apis

The following code example shows how to use `get-apis`.

AWS CLI**To retrieve a list of APIs**

The following `get-apis` example lists all of the APIs for the current user.

```
aws apigatewayv2 get-apis
```

Output:

```
{
  "Items": [
    {
      "ApiEndpoint": "wss://a1b2c3d4.execute-api.us-west-2.amazonaws.com",
      "ApiId": "a1b2c3d4",
      "ApiKeySelectionExpression": "$request.header.x-api-key",
      "CreateDate": "2020-04-07T20:21:59Z",
      "Name": "my-websocket-api",
    }
  ]
}
```

```

        "ProtocolType": "WEBSOCKET",
        "RouteSelectionExpression": "$request.body.message",
        "Tags": {}
    },
    {
        "ApiEndpoint": "https://a1b2c3d5.execute-api.us-west-2.amazonaws.com",
        "ApiId": "a1b2c3d5",
        "ApiKeySelectionExpression": "$request.header.x-api-key",
        "CreateDate": "2020-04-07T20:23:50Z",
        "Name": "my-http-api",
        "ProtocolType": "HTTP",
        "RouteSelectionExpression": "$request.method $request.path",
        "Tags": {}
    }
]
}

```

For more information, see [Working with HTTP APIs](#) and [Working with WebSocket APIs](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [GetApis](#) in *AWS CLI Command Reference*.

get-authorizer

The following code example shows how to use `get-authorizer`.

AWS CLI

To retrieve information about an authorizer

The following `get-authorizer` example displays information about an authorizer.

```

aws apigatewayv2 get-authorizer \
  --api-id a1b2c3d4 \
  --authorizer-id a1b2c3

```

Output:

```

{
  "AuthorizerId": "a1b2c3",
  "AuthorizerType": "JWT",
  "IdentitySource": [

```

```

    "$request.header.Authorization"
  ],
  "JwtConfiguration": {
    "Audience": [
      "123456abc"
    ],
    "Issuer": "https://cognito-idp.us-west-2.amazonaws.com/us-west-2_abc123"
  },
  "Name": "my-jwt-authorizer"
}

```

For more information, see [Controlling access to HTTP APIs with JWT authorizers](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [GetAuthorizer](#) in *AWS CLI Command Reference*.

get-authorizers

The following code example shows how to use `get-authorizers`.

AWS CLI

To retrieve a list of authorizers for an API

The following `get-authorizers` example displays a list of all of the authorizers for an API.

```

aws apigatewayv2 get-authorizers \
  --api-id a1b2c3d4

```

Output:

```

{
  "Items": [
    {
      "AuthorizerId": "a1b2c3",
      "AuthorizerType": "JWT",
      "IdentitySource": [
        "$request.header.Authorization"
      ],
      "JwtConfiguration": {
        "Audience": [
          "123456abc"
        ]
      }
    }
  ]
}

```

```

        ],
        "Issuer": "https://cognito-idp.us-west-2.amazonaws.com/us-
west-2_abc123"
    },
    "Name": "my-jwt-authorizer"
},
{
    "AuthorizerId": "a1b2c4",
    "AuthorizerType": "JWT",
    "IdentitySource": [
        "$request.header.Authorization"
    ],
    "JwtConfiguration": {
        "Audience": [
            "6789abcde"
        ],
        "Issuer": "https://cognito-idp.us-west-2.amazonaws.com/us-
west-2_abc234"
    },
    "Name": "new-jwt-authorizer"
}
]
}

```

For more information, see [Controlling access to HTTP APIs with JWT authorizers](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [GetAuthorizers](#) in *AWS CLI Command Reference*.

get-deployment

The following code example shows how to use `get-deployment`.

AWS CLI

To retrieve information about a deployment

The following `get-deployment` example displays information about a deployment.

```

aws apigatewayv2 get-deployment \
  --api-id a1b2c3d4 \
  --deployment-id abcdef

```

Output:

```
{
  "AutoDeployed": true,
  "CreatedDate": "2020-04-07T23:58:40Z",
  "DeploymentId": "abcdef",
  "DeploymentStatus": "DEPLOYED",
  "Description": "Automatic deployment triggered by changes to the Api
configuration"
}
```

For more information, see [API deployment](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [GetDeployment](#) in *AWS CLI Command Reference*.

get-deployments

The following code example shows how to use `get-deployments`.

AWS CLI**To retrieve a list of deployments**

The following `get-deployments` example displays a list of all of an API's deployments.

```
aws apigatewayv2 get-deployments \
  --api-id a1b2c3d4
```

Output:

```
{
  "Items": [
    {
      "AutoDeployed": true,
      "CreatedDate": "2020-04-07T23:58:40Z",
      "DeploymentId": "abcdef",
      "DeploymentStatus": "DEPLOYED",
      "Description": "Automatic deployment triggered by changes to the Api
configuration"
    },
    {
      "AutoDeployed": true,
```

```

        "CreatedDate": "2020-04-06T00:33:00Z",
        "DeploymentId": "bcdefg",
        "DeploymentStatus": "DEPLOYED",
        "Description": "Automatic deployment triggered by changes to the Api
configuration"
      }
    ]
  }

```

For more information, see [API deployment](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [GetDeployments](#) in *AWS CLI Command Reference*.

get-domain-name

The following code example shows how to use `get-domain-name`.

AWS CLI

To retrieve information about a custom domain name

The following `get-domain-name` example displays information about a custom domain name.

```
aws apigatewayv2 get-domain-name \
  --domain-name api.example.com
```

Output:

```

{
  "ApiMappingSelectionExpression": "$request.basepath",
  "DomainName": "api.example.com",
  "DomainNameConfigurations": [
    {
      "ApiGatewayDomainName": "d-1234.execute-api.us-west-2.amazonaws.com",
      "CertificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/123456789012-1234-1234-1234-12345678",
      "EndpointType": "REGIONAL",
      "HostedZoneId": "123456789111",
      "SecurityPolicy": "TLS_1_2",
      "DomainNameStatus": "AVAILABLE"
    }
  ],
  "Tags": {}
}

```

```
}
```

For more information, see [Setting up a regional custom domain name in API Gateway](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [GetDomainName](#) in *AWS CLI Command Reference*.

get-domain-names

The following code example shows how to use `get-domain-names`.

AWS CLI

To retrieve a list of custom domain names

The following `get-domain-names` example displays a list of all of the custom domain names for the current user.

```
aws apigatewayv2 get-domain-names
```

Output:

```
{
  "Items": [
    {
      "ApiMappingSelectionExpression": "$request.basepath",
      "DomainName": "api.example.com",
      "DomainNameConfigurations": [
        {
          "ApiGatewayDomainName": "d-1234.execute-api.us-
west-2.amazonaws.com",
          "CertificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/123456789012-1234-1234-1234-12345678",
          "EndpointType": "REGIONAL",
          "HostedZoneId": "123456789111",
          "SecurityPolicy": "TLS_1_2",
          "DomainNameStatus": "AVAILABLE"
        }
      ]
    },
    {
      "ApiMappingSelectionExpression": "$request.basepath",
```

```

    "DomainName": "newApi.example.com",
    "DomainNameConfigurations": [
      {
        "ApiGatewayDomainName": "d-5678.execute-api.us-
west-2.amazonaws.com",
        "CertificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/123456789012-1234-1234-1234-12345678",
        "EndpointType": "REGIONAL",
        "HostedZoneId": "123456789222",
        "SecurityPolicy": "TLS_1_2",
        "DomainNameStatus": "AVAILABLE"
      }
    ]
  }
]
}

```

For more information, see [Setting up a regional custom domain name in API Gateway](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [GetDomainNames](#) in *AWS CLI Command Reference*.

get-integration

The following code example shows how to use `get-integration`.

AWS CLI

To retrieve information about an integration

The following `get-integration` example displays information about an integration.

```

aws apigatewayv2 get-integration \
  --api-id a1b2c3d4 \
  --integration-id a1b2c3

```

Output:

```

{
  "ApiGatewayManaged": true,
  "ConnectionType": "INTERNET",
  "IntegrationId": "a1b2c3",

```



```
"IntegrationMethod": "POST",
"IntegrationType": "AWS_PROXY",
"IntegrationUri": "arn:aws:lambda:us-west-2:12356789012:function:hello12",
"PayloadFormatVersion": "2.0",
"TimeoutInMillis": 30000
}
```

For more information, see [Configuring integrations for HTTP APIs](#) and [Setting up WebSocket API integrations](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [GetIntegration](#) in *AWS CLI Command Reference*.

get-integrations

The following code example shows how to use `get-integrations`.

AWS CLI

To retrieve a list of integrations

The following `get-integrations` example displays a list of all of an API's integrations.

```
aws apigatewayv2 get-integrations \
  --api-id a1b2c3d4
```

Output:

```
{
  "Items": [
    {
      "ApiGatewayManaged": true,
      "ConnectionType": "INTERNET",
      "IntegrationId": "a1b2c3",
      "IntegrationMethod": "POST",
      "IntegrationType": "AWS_PROXY",
      "IntegrationUri": "arn:aws:lambda:us-west-2:123456789012:function:my-
function",
      "PayloadFormatVersion": "2.0",
      "TimeoutInMillis": 30000
    },
    {
      "ConnectionType": "INTERNET",
      "IntegrationId": "a1b2c4",
```

```
        "IntegrationMethod": "ANY",
        "IntegrationType": "HTTP_PROXY",
        "IntegrationUri": "https://www.example.com",
        "PayloadFormatVersion": "1.0",
        "TimeoutInMillis": 30000
    }
]
}
```

For more information, see [Configuring integrations for HTTP APIs](#) and [Setting up WebSocket API integrations](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [GetIntegrations](#) in *AWS CLI Command Reference*.

get-route

The following code example shows how to use `get-route`.

AWS CLI

To retrieve information about a route

The following `get-route` example displays information about a route.

```
aws apigatewayv2 get-route \
  --api-id a1b2c3d4 \
  --route-id 72jz1wk
```

Output:

```
{
  "ApiKeyRequired": false,
  "AuthorizationType": "NONE",
  "RouteId": "72jz1wk",
  "RouteKey": "ANY /pets",
  "Target": "integrations/a1b2c3"
}
```

For more information, see [Working with routes for HTTP APIs](#) in the *Amazon API Gateway Developer Guide*

- For API details, see [GetRoute](#) in *AWS CLI Command Reference*.

get-routes

The following code example shows how to use `get-routes`.

AWS CLI

To retrieve a list of routes

The following `get-routes` example displays a list of all of an API's routes.

```
aws apigatewayv2 get-routes \  
  --api-id a1b2c3d4
```

Output:

```
{  
  "Items": [  
    {  
      "ApiKeyRequired": false,  
      "AuthorizationType": "NONE",  
      "RouteId": "72jz1wk",  
      "RouteKey": "ANY /admin",  
      "Target": "integrations/a1b2c3"  
    },  
    {  
      "ApiGatewayManaged": true,  
      "ApiKeyRequired": false,  
      "AuthorizationType": "NONE",  
      "RouteId": "go65gqi",  
      "RouteKey": "$default",  
      "Target": "integrations/a1b2c4"  
    }  
  ]  
}
```

For more information, see [Working with routes for HTTP APIs](#) in the *Amazon API Gateway Developer Guide*

- For API details, see [GetRoutes](#) in *AWS CLI Command Reference*.

get-stage

The following code example shows how to use `get-stage`.

AWS CLI

To retrieve information about a stage

The following `get-stage` example displays information about the `prod` stage of an API.

```
aws apigatewayv2 get-stage \  
  --api-id a1b2c3d4 \  
  --stage-name prod
```

Output:

```
{  
  "CreateDate": "2020-04-08T00:36:05Z",  
  "DefaultRouteSettings": {  
    "DetailedMetricsEnabled": false  
  },  
  "DeploymentId": "x1zwyv",  
  "LastUpdatedDate": "2020-04-08T00:36:13Z",  
  "RouteSettings": {},  
  "StageName": "prod",  
  "StageVariables": {  
    "function": "my-prod-function"  
  },  
  "Tags": {}  
}
```

For more information, see [Working with stages for HTTP APIs](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [GetStage](#) in *AWS CLI Command Reference*.

get-stages

The following code example shows how to use `get-stages`.

AWS CLI

To retrieve a list of stages

The following `get-stages` example lists all of an API's stages.

```
aws apigatewayv2 get-stages \  
  --api-id a1b2c3d4
```

Output:

```
{  
  "Items": [  
    {  
      "ApiGatewayManaged": true,  
      "AutoDeploy": true,  
      "CreateDate": "2020-04-08T00:08:44Z",  
      "DefaultRouteSettings": {  
        "DetailedMetricsEnabled": false  
      },  
      "DeploymentId": "dty748",  
      "LastDeploymentStatusMessage": "Successfully deployed stage with  
deployment ID 'dty748'",  
      "LastUpdatedDate": "2020-04-08T00:09:49Z",  
      "RouteSettings": {},  
      "StageName": "$default",  
      "StageVariables": {},  
      "Tags": {}  
    },  
    {  
      "AutoDeploy": true,  
      "CreateDate": "2020-04-08T00:35:06Z",  
      "DefaultRouteSettings": {  
        "DetailedMetricsEnabled": false  
      },  
      "LastUpdatedDate": "2020-04-08T00:35:48Z",  
      "RouteSettings": {},  
      "StageName": "dev",  
      "StageVariables": {  
        "function": "my-dev-function"  
      },  
      "Tags": {}  
    },  
    {  
      "CreateDate": "2020-04-08T00:36:05Z",  
      "DefaultRouteSettings": {  
        "DetailedMetricsEnabled": false  
      },  
      "DeploymentId": "x1zwyv",
```

```
        "LastUpdatedDate": "2020-04-08T00:36:13Z",
        "RouteSettings": {},
        "StageName": "prod",
        "StageVariables": {
            "function": "my-prod-function"
        },
        "Tags": {}
    }
]
}
```

For more information, see [Working with stages for HTTP APIs](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [GetStages](#) in *AWS CLI Command Reference*.

get-tags

The following code example shows how to use `get-tags`.

AWS CLI

To retrieve a list of tags for a resource

The following `get-tags` example lists all of an API's tags.

```
aws apigatewayv2 get-tags \
  --resource-arn arn:aws:apigateway:us-west-2::/apis/a1b2c3d4
```

Output:

```
{
  "Tags": {
    "owner": "dev-team",
    "environment": "prod"
  }
}
```

For more information, see [Tagging your API Gateway resources](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [GetTags](#) in *AWS CLI Command Reference*.

get-vpc-link

The following code example shows how to use `get-vpc-link`.

AWS CLI

To retrieve information about a VPC link

The following `get-vpc-link` example displays information about a VPC link.

```
aws apigatewayv2 get-vpc-link \
  --vpc-link-id abcd123
```

Output:

```
{
  "CreateDate": "2020-04-07T00:27:47Z",
  "Name": "MyVpcLink",
  "SecurityGroupIds": [
    "sg1234",
    "sg5678"
  ],
  "SubnetIds": [
    "subnet-aaaa",
    "subnet-bbbb"
  ],
  "Tags": {},
  "VpcLinkId": "abcd123",
  "VpcLinkStatus": "AVAILABLE",
  "VpcLinkStatusMessage": "VPC link is ready to route traffic",
  "VpcLinkVersion": "V2"
}
```

For more information, see [Working with VPC links for HTTP APIs](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [GetVpcLink](#) in *AWS CLI Command Reference*.

get-vpc-links

The following code example shows how to use `get-vpc-links`.

AWS CLI

To retrieve a list of VPC links

The following `get-vpc-links` example displays a list of all of the VPC links for the current user.

```
aws apigatewayv2 get-vpc-links
```

Output:

```
{
  "Items": [
    {
      "CreateDate": "2020-04-07T00:27:47Z",
      "Name": "MyVpcLink",
      "SecurityGroupIds": [
        "sg1234",
        "sg5678"
      ],
      "SubnetIds": [
        "subnet-aaaa",
        "subnet-bbbb"
      ],
      "Tags": {},
      "VpcLinkId": "abcd123",
      "VpcLinkStatus": "AVAILABLE",
      "VpcLinkStatusMessage": "VPC link is ready to route traffic",
      "VpcLinkVersion": "V2"
    }
  ]
}
```



```
        "VpcLinkStatus": "AVAILABLE",
        "VpcLinkStatusMessage": "VPC link is ready to route traffic",
        "VpcLinkVersion": "V2"
    }
]
}
```

For more information, see [Working with VPC links for HTTP APIs](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [GetVpcLinks](#) in *AWS CLI Command Reference*.

import-api

The following code example shows how to use `import-api`.

AWS CLI

To import an HTTP API

The following `import-api` example creates an HTTP API from an OpenAPI 3.0 definition file named `api-definition.yaml`.

```
aws apigatewayv2 import-api \  
  --body file://api-definition.yaml
```

Contents of `api-definition.yaml`:

```
openapi: 3.0.1  
info:  
  title: My Lambda API  
  version: v1.0  
paths:  
  /hello:  
    x-amazon-apigateway-any-method:  
      x-amazon-apigateway-integration:  
        payloadFormatVersion: 2.0  
        type: aws_proxy  
        httpMethod: POST  
        uri: arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/  
arn:aws:lambda:us-west-2:123456789012:function:hello/invocations  
        connectionType: INTERNET
```

Output:

```
{
  "ApiEndpoint": "https://a1b2c3d4.execute-api.us-west-2.amazonaws.com",
  "ApiId": "a1b2c3d4",
  "ApiKeySelectionExpression": "$request.header.x-api-key",
  "CreateDate": "2020-04-08T17:19:38+00:00",
  "Name": "My Lambda API",
  "ProtocolType": "HTTP",
  "RouteSelectionExpression": "$request.method $request.path",
  "Tags": {},
  "Version": "v1.0"
}
```

For more information, see [Working with OpenAPI definitions for HTTP APIs](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [ImportApi](#) in *AWS CLI Command Reference*.

reimport-api

The following code example shows how to use `reimport-api`.

AWS CLI

To reimport an HTTP API

The following `reimport-api` example updates an existing HTTP API to use the OpenAPI 3.0 definition specified in `api-definition.yaml`.

```
aws apigatewayv2 reimport-api \
  --body file://api-definition.yaml \
  --api-id a1b2c3d4
```

Contents of `api-definition.yaml`:

```
openapi: 3.0.1
info:
  title: My Lambda API
  version: v1.0
paths:
```

```

/hello:
  x-amazon-apigateway-any-method:
    x-amazon-apigateway-integration:
      payloadFormatVersion: 2.0
      type: aws_proxy
      httpMethod: POST
      uri: arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-west-2:12356789012:function:hello/invocations
      connectionType: INTERNET

```

Output:

```

{
  "ApiEndpoint": "https://a1b2c3d4.execute-api.us-west-2.amazonaws.com",
  "ApiId": "a1b2c3d4",
  "ApiKeySelectionExpression": "$request.header.x-api-key",
  "CreateDate": "2020-04-08T17:19:38+00:00",
  "Name": "My Lambda API",
  "ProtocolType": "HTTP",
  "RouteSelectionExpression": "$request.method $request.path",
  "Tags": {},
  "Version": "v1.0"
}

```

For more information, see [Working with OpenAPI definitions for HTTP APIs](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [ReimportApi](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To tag a resource

The following tag-resource example adds a tag with the key name Department and a value of Accounting to the specified API.

```

aws apigatewayv2 tag-resource \
  --resource-arn arn:aws:apigateway:us-west-2::/apis/a1b2c3d4 \

```

```
--tags Department=Accounting
```

This command produces no output.

For more information, see [Tagging your API Gateway resources](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from a resource

The following `untag-resource` example removes tags with the key names `Project` and `Owner` from the specified API.

```
aws apigatewayv2 untag-resource \  
  --resource-arn arn:aws:apigateway:us-west-2::/apis/a1b2c3d4 \  
  --tag-keys Project Owner
```

This command produces no output.

For more information, see [Tagging your API Gateway resources](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-api-mapping

The following code example shows how to use `update-api-mapping`.

AWS CLI

To update an API mapping

The following `update-api-mapping` example changes an API mapping for a custom domain name. As a result, the base URL using the custom domain name for the specified API and stage becomes `https://api.example.com/dev`.

```
aws apigatewayv2 update-api-mapping \  
  --api-id a1b2c3d4 \  
  --stage dev \  
  --domain-name api.example.com \  
  --api-mapping-id 0qzs2sy7bh \  
  --api-mapping-key dev
```

Output:

```
{  
  "ApiId": "a1b2c3d4",  
  "ApiMappingId": "0qzs2sy7bh",  
  "ApiMappingKey": "dev"  
  "Stage": "dev"  
}
```

For more information, see [Setting up a regional custom domain name in API Gateway](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [UpdateApiMapping](#) in *AWS CLI Command Reference*.

update-api

The following code example shows how to use `update-api`.

AWS CLI

To enable CORS for an HTTP API

The following `update-api` example updates the specified API's CORS configuration to allow requests from `https://www.example.com`.

```
aws apigatewayv2 update-api \  
  --api-id a1b2c3d4 \  
  --cors-configuration AllowOrigins=https://www.example.com
```

Output:

```
{  
  "ApiEndpoint": "https://a1b2c3d4.execute-api.us-west-2.amazonaws.com",
```

```
"ApiId": "a1b2c3d4",
"ApiKeySelectionExpression": "$request.header.x-api-key",
"CorsConfiguration": {
  "AllowCredentials": false,
  "AllowHeaders": [
    "header1",
    "header2"
  ],
  "AllowMethods": [
    "GET",
    "OPTIONS"
  ],
  "AllowOrigins": [
    "https://www.example.com"
  ]
},
"CreateDate": "2020-04-08T18:39:37+00:00",
"Name": "my-http-api",
"ProtocolType": "HTTP",
"RouteSelectionExpression": "$request.method $request.path",
"Tags": {},
"Version": "v1.0"
}
```

For more information, see [Configuring CORS for an HTTP API](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [UpdateApi](#) in *AWS CLI Command Reference*.

update-authorizer

The following code example shows how to use `update-authorizer`.

AWS CLI

To update an authorizer

The following `update-authorizer` example changes a JWT authorizer's identity source to a header named `Authorization`.

```
aws apigatewayv2 update-authorizer \
  --api-id a1b2c3d4 \
  --authorizer-id a1b2c3 \
```

```
--identity-source '$request.header.Authorization'
```

Output:

```
{
  "AuthorizerId": "a1b2c3",
  "AuthorizerType": "JWT",
  "IdentitySource": [
    "$request.header.Authorization"
  ],
  "JwtConfiguration": {
    "Audience": [
      "123456abc"
    ],
    "Issuer": "https://cognito-idp.us-west-2.amazonaws.com/us-west-2_abc123"
  },
  "Name": "my-jwt-authorizer"
}
```

For more information, see [Controlling access to HTTP APIs with JWT authorizers](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [UpdateAuthorizer](#) in *AWS CLI Command Reference*.

update-deployment

The following code example shows how to use update-deployment.

AWS CLI

To change a deployment's description

The following update-deployment example updates a deployment's description.

```
aws apigatewayv2 update-deployment \  
  --api-id a1b2c3d4 \  
  --deployment-id abcdef \  
  --description 'Manual deployment to fix integration test failures.'
```

Output:

```
{
```

```
"AutoDeployed": false,
"CreateDate": "2020-02-05T16:21:48+00:00",
"DeploymentId": "abcdef",
"DeploymentStatus": "DEPLOYED",
"Description": "Manual deployment to fix integration test failures."
}
```

For more information, see [Developing an HTTP API in API Gateway](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [UpdateDeployment](#) in *AWS CLI Command Reference*.

update-domain-name

The following code example shows how to use `update-domain-name`.

AWS CLI

To update a custom domain name

The following `update-domain-name` example specifies a new ACM certificate for the `api.example.com` custom domain name.

```
aws apigatewayv2 update-domain-name \
  --domain-name api.example.com \
  --domain-name-configurations CertificateArn=arn:aws:acm:us-
west-2:123456789012:certificate/123456789012-1234-1234-1234-12345678
```

Output:

```
{
  "ApiMappingSelectionExpression": "$request.basepath",
  "DomainName": "regional.example.com",
  "DomainNameConfigurations": [
    {
      "ApiGatewayDomainName": "d-id.execute-api.us-west-2.amazonaws.com",
      "CertificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/123456789012-1234-1234-1234-12345678",
      "EndpointType": "REGIONAL",
      "HostedZoneId": "123456789111",
      "SecurityPolicy": "TLS_1_2",
      "DomainNameStatus": "AVAILABLE"
    }
  ]
}
```



```

    }
  ]
}

```

For more information, see [Setting up a regional custom domain name in API Gateway](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [UpdateDomainName](#) in *AWS CLI Command Reference*.

update-integration

The following code example shows how to use `update-integration`.

AWS CLI

To update a Lambda integration

The following `update-integration` example updates an existing AWS Lambda integration to use the specified Lambda function.

```

aws apigatewayv2 update-integration \
  --api-id a1b2c3d4 \
  --integration-id a1b2c3 \
  --integration-uri arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/functions/
arn:aws:lambda:us-west-2:123456789012:function:my-new-function/invocations

```

Output:

```

{
  "ConnectionType": "INTERNET",
  "IntegrationId": "a1b2c3",
  "IntegrationMethod": "POST",
  "IntegrationType": "AWS_PROXY",
  "IntegrationUri": "arn:aws:apigateway:us-west-2:lambda:path/2015-03-31/
functions/arn:aws:lambda:us-west-2:123456789012:function:my-new-function/
invocations",
  "PayloadFormatVersion": "2.0",
  "TimeoutInMillis": 5000
}

```

For more information, see [Configuring integrations for HTTP APIs](#) and [Setting up WebSocket API integrations](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [UpdateIntegration](#) in *AWS CLI Command Reference*.

update-route

The following code example shows how to use update-route.

AWS CLI

Example 1: To update the integration of a route

The following update-route example updates the integration of a specified route.

```
aws apigatewayv2 update-route \  
  --api-id a1b2c3d4 \  
  --route-id a1b2c3 \  
  --target integrations/a1b2c6
```

Output:

```
{  
  "ApiKeyRequired": false,  
  "AuthorizationType": "NONE",  
  "RouteId": "a1b2c3",  
  "RouteKey": "ANY /pets",  
  "Target": "integrations/a1b2c6"  
}
```

Example 2: To add an authorizer to a route

The following update-route example updates the specified route to use a JWT authorizer.

```
aws apigatewayv2 update-route \  
  --api-id a1b2c3d4 \  
  --route-id a1b2c3 \  
  --authorization-type JWT \  
  --authorizer-id a1b2c5 \  
  --authorization-scopes user.id user.email
```

Output:

```
{
```

```
"ApiKeyRequired": false,
"AuthorizationScopes": [
  "user.id",
  "user.email"
],
"AuthorizationType": "JWT",
"AuthorizerId": "a1b2c5",
"OperationName": "GET HTTP",
"RequestParameters": {},
"RouteId": "a1b2c3",
"RouteKey": "GET /pets",
"Target": "integrations/a1b2c6"
}
```

For more information, see [Controlling access to HTTP APIs with JWT authorizers](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [UpdateRoute](#) in *AWS CLI Command Reference*.

update-stage

The following code example shows how to use `update-stage`.

AWS CLI

To configure custom throttling

The following `update-stage` example configures custom throttling for the specified stage and route of an API.

```
aws apigatewayv2 update-stage \
  --api-id a1b2c3d4 \
  --stage-name dev \
  --route-settings '{"GET /pets":
{"ThrottlingBurstLimit":100,"ThrottlingRateLimit":2000}}'
```

Output:

```
{
  "CreateDate": "2020-04-05T16:21:16+00:00",
  "DefaultRouteSettings": {
    "DetailedMetricsEnabled": false
```

```
  },
  "DeploymentId": "shktxb",
  "LastUpdatedDate": "2020-04-08T22:23:17+00:00",
  "RouteSettings": {
    "GET /pets": {
      "ThrottlingBurstLimit": 100,
      "ThrottlingRateLimit": 2000.0
    }
  },
  "StageName": "dev",
  "StageVariables": {},
  "Tags": {}
}
```

For more information, see [Protecting your HTTP API](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [UpdateStage](#) in *AWS CLI Command Reference*.

update-vpc-link

The following code example shows how to use `update-vpc-link`.

AWS CLI

To update a VPC link

The following `update-vpc-link` example updates the name of a VPC link. After you've created a VPC link, you can't change its security groups or subnets.

```
aws apigatewayv2 update-vpc-link \
  --vpc-link-id abcd123 \
  --name MyUpdatedVpcLink
```

Output:

```
{
  "CreatedDate": "2020-04-07T00:27:47Z",
  "Name": "MyUpdatedVpcLink",
  "SecurityGroupIds": [
    "sg1234",
    "sg5678"
  ]
}
```

```
  ],
  "SubnetIds": [
    "subnet-aaaa",
    "subnet-bbbb"
  ],
  "Tags": {},
  "VpcLinkId": "abcd123",
  "VpcLinkStatus": "AVAILABLE",
  "VpcLinkStatusMessage": "VPC link is ready to route traffic",
  "VpcLinkVersion": "V2"
}
```

For more information, see [Working with VPC links for HTTP APIs](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [UpdateVpcLink](#) in *AWS CLI Command Reference*.

API Gateway Management API examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with API Gateway Management API.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

delete-connection

The following code example shows how to use `delete-connection`.

AWS CLI

To delete a WebSocket connection

The following `delete-connection` example disconnects a client from the specified WebSocket API.

```
aws apigatewaymanagementapi delete-connection \  
  --connection-id L0SM9c0FvHcCIhw= \  
  --endpoint-url https://aabbccdde.execute-api.us-west-2.amazonaws.com/prod
```

This command produces no output.

For more information, see [Use @connections commands in your backend service](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [DeleteConnection](#) in *AWS CLI Command Reference*.

get-connection

The following code example shows how to use `get-connection`.

AWS CLI

To get information about a WebSocket connection

The following `get-connection` example describes a connection to the specified WebSocket API.

```
aws apigatewaymanagementapi get-connection \  
  --connection-id L0SM9c0FvHcCIhw= \  
  --endpoint-url https://aabbccdde.execute-api.us-west-2.amazonaws.com/prod
```

Output:

```
{  
  "ConnectedAt": "2020-04-30T20:10:33.236Z",  
  "Identity": {  
    "SourceIp": "192.0.2.1"  
  },  
  "LastActiveAt": "2020-04-30T20:10:42.997Z"
```

```
}
```

For more information, see [Use @connections commands in your backend service](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [GetConnection](#) in *AWS CLI Command Reference*.

post-to-connection

The following code example shows how to use `post-to-connection`.

AWS CLI

To send data to a WebSocket connection

The following `post-to-connection` example sends a message to a client that's connected to the specified WebSocket API.

```
aws apigatewaymanagementapi post-to-connection \  
  --connection-id L0SM9c0FvHcCIhw= \  
  --data "Hello from API Gateway!" \  
  --endpoint-url https://aabbccddee.execute-api.us-west-2.amazonaws.com/prod
```

This command produces no output.

For more information, see [Use @connections commands in your backend service](#) in the *Amazon API Gateway Developer Guide*.

- For API details, see [PostToConnection](#) in *AWS CLI Command Reference*.

App Mesh examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with App Mesh.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-mesh

The following code example shows how to use create-mesh.

AWS CLI

Example 1: To create a new service mesh

The following create-mesh example creates a service mesh.

```
aws appmesh create-mesh \  
  --mesh-name app1
```

Output:

```
{  
  "mesh":{  
    "meshName":"app1",  
    "metadata":{  
      "arn":"arn:aws:appmesh:us-east-1:123456789012:mesh/app1",  
      "createdAt":1563809909.282,  
      "lastUpdatedAt":1563809909.282,  
      "uid":"a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
      "version":1  
    },  
    "spec":{ },  
    "status":{  
      "status":"ACTIVE"  
    }  
  }  
}
```

Example 2: To create a new service mesh with multiple tags

The following `create-mesh` example creates a service mesh with multiple tags.

```
aws appmesh create-mesh \  
  --mesh-name app2 \  
  --tags key=key1,value=value1 key=key2,value=value2 key=key3,value=value3
```

Output:

```
{  
  "mesh":{  
    "meshName":"app2",  
    "metadata":{  
      "arn":"arn:aws:appmesh:us-east-1:123456789012:mesh/app2",  
      "createdAt":1563822121.877,  
      "lastUpdatedAt":1563822121.877,  
      "uid":"a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
      "version":1  
    },  
    "spec":{ },  
    "status":{  
      "status":"ACTIVE"  
    }  
  }  
}
```

For more information, see [Service Meshes](#) in the *AWS App Mesh User Guide*.

- For API details, see [CreateMesh](#) in *AWS CLI Command Reference*.

create-route

The following code example shows how to use `create-route`.

AWS CLI

To create a new gRPC route

The following `create-route` example uses a JSON input file to create a gRPC route. gRPC traffic that has metadata that starts with 123 is routed to a virtual node named `serviceBgrpc`. If there are specific gRPC, HTTP, or TCP failures when attempting to communicate with the target of the route, the route is retried three times. There is a 15 second delay between each retry attempt.

```
aws appmesh create-route \  
  --cli-input-json file://create-route-grpc.json
```

Contents of create-route-grpc.json:

```
{  
  "meshName" : "apps",  
  "routeName" : "grpcRoute",  
  "spec" : {  
    "grpcRoute" : {  
      "action" : {  
        "weightedTargets" : [  
          {  
            "virtualNode" : "serviceBgrpc",  
            "weight" : 100  
          }  
        ]  
      },  
      "match" : {  
        "metadata" : [  
          {  
            "invert" : false,  
            "match" : {  
              "prefix" : "123"  
            },  
            "name" : "myMetadata"  
          }  
        ],  
        "methodName" : "GetColor",  
        "serviceName" : "com.amazonaws.services.ColorService"  
      },  
      "retryPolicy" : {  
        "grpcRetryEvents" : [ "deadline-exceeded" ],  
        "httpRetryEvents" : [ "server-error", "gateway-error" ],  
        "maxRetries" : 3,  
        "perRetryTimeout" : {  
          "unit" : "s",  
          "value" : 15  
        },  
        "tcpRetryEvents" : [ "connection-error" ]  
      }  
    },  
    "priority" : 100  
  }  
}
```

```

    },
    "virtualRouterName" : "serviceBgrpc"
  }

```

Output:

```

{
  "route": {
    "meshName": "apps",
    "metadata": {
      "arn": "arn:aws:appmesh:us-west-2:123456789012:mesh/apps/virtualRouter/
serviceBgrpc/route/grpcRoute",
      "createdAt": 1572010806.008,
      "lastUpdatedAt": 1572010806.008,
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
      "version": 1
    },
    "routeName": "grpcRoute",
    "spec": {
      "grpcRoute": {
        "action": {
          "weightedTargets": [
            {
              "virtualNode": "serviceBgrpc",
              "weight": 100
            }
          ]
        },
        "match": {
          "metadata": [
            {
              "invert": false,
              "match": {
                "prefix": "123"
              },
              "name": "mymetadata"
            }
          ],
          "methodName": "GetColor",
          "serviceName": "com.amazonaws.services.ColorService"
        },
        "retryPolicy": {
          "grpcRetryEvents": [

```

```

        "deadline-exceeded"
    ],
    "httpRetryEvents": [
        "server-error",
        "gateway-error"
    ],
    "maxRetries": 3,
    "perRetryTimeout": {
        "unit": "s",
        "value": 15
    },
    "tcpRetryEvents": [
        "connection-error"
    ]
    }
    },
    "priority": 100
},
"status": {
    "status": "ACTIVE"
},
"virtualRouterName": "serviceBgRPC"
}
}

```

To create a new HTTP or HTTP/2 route

The following `create-route` example uses a JSON input file to create an HTTP/2 route. To create an HTTP route, replace `http2Route` with `httpRoute` under `spec`. All HTTP/2 traffic addressed to any URL prefix that has a header value that starts with 123 is routed to a virtual node named `serviceBhttp2`. If there are specific HTTP or TCP failures when attempting to communicate with the target of the route, the route is retried three times. There is a 15 second delay between each retry attempt.

```

aws appmesh create-route \
  --cli-input-json file://create-route-http2.json

```

Contents of `create-route-http2.json`:

```

{
  "meshName": "apps",
  "routeName": "http2Route",

```

```
"spec": {
  "http2Route": {
    "action": {
      "weightedTargets": [
        {
          "virtualNode": "serviceBhttp2",
          "weight": 100
        }
      ]
    },
    "match": {
      "headers": [
        {
          "invert": false,
          "match": {
            "prefix": "123"
          },
          "name": "clientRequestId"
        }
      ],
      "method": "POST",
      "prefix": "/",
      "scheme": "http"
    },
    "retryPolicy": {
      "httpRetryEvents": [
        "server-error",
        "gateway-error"
      ],
      "maxRetries": 3,
      "perRetryTimeout": {
        "unit": "s",
        "value": 15
      },
      "tcpRetryEvents": [
        "connection-error"
      ]
    }
  },
  "priority": 200
},
"virtualRouterName": "serviceBhttp2"
}
```

Output:

```
{
  "route": {
    "meshName": "apps",
    "metadata": {
      "arn": "arn:aws:appmesh:us-west-2:123456789012:mesh/apps/virtualRouter/
serviceBhttp2/route/http2Route",
      "createdAt": 1572011008.352,
      "lastUpdatedAt": 1572011008.352,
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
      "version": 1
    },
    "routeName": "http2Route",
    "spec": {
      "http2Route": {
        "action": {
          "weightedTargets": [
            {
              "virtualNode": "serviceBhttp2",
              "weight": 100
            }
          ]
        },
        "match": {
          "headers": [
            {
              "invert": false,
              "match": {
                "prefix": "123"
              },
              "name": "clientRequestId"
            }
          ],
          "method": "POST",
          "prefix": "/",
          "scheme": "http"
        },
        "retryPolicy": {
          "httpRetryEvents": [
            "server-error",
            "gateway-error"
          ],
          "maxRetries": 3,

```

```

        "perRetryTimeout": {
            "unit": "s",
            "value": 15
        },
        "tcpRetryEvents": [
            "connection-error"
        ]
    },
    "priority": 200
},
"status": {
    "status": "ACTIVE"
},
"virtualRouterName": "serviceBhttp2"
}
}

```

To create a new TCP route

The following `create-route` example uses a JSON input file to create a TCP route. 75 percent of traffic is routed to a virtual node named `serviceBtcp`, and 25 percent of traffic is routed to a virtual node named `serviceBv2tcp`. Specifying different weightings for different targets is an effective way to do a deployment of a new version of an application. You can adjust the weights so that eventually, 100 percent of all traffic is routed to a target that has the new version of an application.

```

aws appmesh create-route \
  --cli-input-json file://create-route-tcp.json

```

Contents of `create-route-tcp.json`:

```

{
  "meshName": "apps",
  "routeName": "tcpRoute",
  "spec": {
    "priority": 300,
    "tcpRoute": {
      "action": {
        "weightedTargets": [
          {
            "virtualNode": "serviceBtcp",

```

```

        "weight": 75
      },
      {
        "virtualNode": "serviceBv2tcp",
        "weight": 25
      }
    ]
  }
},
"virtualRouterName": "serviceBtcp"
}

```

Output:

```

{
  "route": {
    "meshName": "apps",
    "metadata": {
      "arn": "arn:aws:appmesh:us-west-2:123456789012:mesh/apps/virtualRouter/
serviceBtcp/route/tcpRoute",
      "createdAt": 1572011436.26,
      "lastUpdatedAt": 1572011436.26,
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
      "version": 1
    },
    "routeName": "tcpRoute",
    "spec": {
      "priority": 300,
      "tcpRoute": {
        "action": {
          "weightedTargets": [
            {
              "virtualNode": "serviceBtcp",
              "weight": 75
            },
            {
              "virtualNode": "serviceBv2tcp",
              "weight": 25
            }
          ]
        }
      }
    }
  }
}

```



```
    },
    "status": {
      "status": "ACTIVE"
    },
    "virtualRouterName": "serviceBtcp"
  }
}
```

For more information, see [Routes](#) in the *AWS App Mesh User Guide*.

- For API details, see [CreateRoute](#) in *AWS CLI Command Reference*.

create-virtual-gateway

The following code example shows how to use `create-virtual-gateway`.

AWS CLI

To create a new virtual gateway

The following `create-virtual-gateway` example uses a JSON input file to create a virtual gateway with a listener for HTTP using port 9080.

```
aws appmesh create-virtual-gateway \
  --mesh-name meshName \
  --virtual-gateway-name virtualGatewayName \
  --cli-input-json file://create-virtual-gateway.json
```

Contents of `create-virtual-gateway.json`:

```
{
  "spec": {
    "listeners": [
      {
        "portMapping": {
          "port": 9080,
          "protocol": "http"
        }
      }
    ]
  }
}
```

Output:

```
{
  "virtualGateway": {
    "meshName": "meshName",
    "metadata": {
      "arn": "arn:aws:appmesh:us-west-2:123456789012:mesh/meshName/
virtualGateway/virtualGatewayName",
      "createdAt": "2022-04-06T10:42:42.015000-05:00",
      "lastUpdatedAt": "2022-04-06T10:42:42.015000-05:00",
      "meshOwner": "123456789012",
      "resourceOwner": "123456789012",
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
      "version": 1
    },
    "spec": {
      "listeners": [
        {
          "portMapping": {
            "port": 9080,
            "protocol": "http"
          }
        }
      ]
    },
    "status": {
      "status": "ACTIVE"
    },
    "virtualGatewayName": "virtualGatewayName"
  }
}
```

For more information, see [Virtual Gateways](#) in the *AWS App Mesh User Guide*.

- For API details, see [CreateVirtualGateway](#) in *AWS CLI Command Reference*.

create-virtual-node

The following code example shows how to use `create-virtual-node`.

AWS CLI**Example 1: To create a new virtual node that uses DNS for discovery**

The following `create-virtual-node` example uses a JSON input file to create a virtual node that uses DNS for service discovery.

```
aws appmesh create-virtual-node \  
  --cli-input-json file://create-virtual-node-dns.json
```

Contents of `create-virtual-node-dns.json`:

```
{  
  "meshName": "app1",  
  "spec": {  
    "listeners": [  
      {  
        "portMapping": {  
          "port": 80,  
          "protocol": "http"  
        }  
      }  
    ],  
    "serviceDiscovery": {  
      "dns": {  
        "hostname": "serviceBv1.svc.cluster.local"  
      }  
    }  
  },  
  "virtualNodeName": "vnServiceBv1"  
}
```

Output:

```
{  
  "virtualNode": {  
    "meshName": "app1",  
    "metadata": {  
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualNode/  
vnServiceBv1",  
      "createdAt": 1563810019.874,  
      "lastUpdatedAt": 1563810019.874,  
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
      "version": 1  
    },  
    "spec": {
```

```

    "listeners": [
      {
        "portMapping": {
          "port": 80,
          "protocol": "http"
        }
      }
    ],
    "serviceDiscovery": {
      "dns": {
        "hostname": "serviceBv1.svc.cluster.local"
      }
    }
  },
  "status": {
    "status": "ACTIVE"
  },
  "virtualNodeName": "vnServiceBv1"
}

```

Example 2: To create a new virtual node that uses AWS Cloud Map for discovery

The following `create-virtual-node` example uses a JSON input file to create a virtual node that uses AWS Cloud Map for service discovery.

```

aws appmesh create-virtual-node \
  --cli-input-json file://create-virtual-node-cloud-map.json

```

Contents of `create-virtual-node-cloud-map.json`:

```

{
  "meshName": "app1",
  "spec": {
    "backends": [
      {
        "virtualService": {
          "virtualServiceName": "serviceA.svc.cluster.local"
        }
      }
    ],
    "listeners": [
      {

```

```

        "portMapping": {
            "port": 80,
            "protocol": "http"
        }
    ],
    "serviceDiscovery": {
        "awsCloudMap": {
            "attributes": [
                {
                    "key": "Environment",
                    "value": "Testing"
                }
            ],
            "namespaceName": "namespace1",
            "serviceName": "serviceA"
        }
    },
    "virtualNodeName": "vnServiceA"
}

```

Output:

```

{
  "virtualNode": {
    "meshName": "app1",
    "metadata": {
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualNode/vnServiceA",
      "createdAt": 1563810859.465,
      "lastUpdatedAt": 1563810859.465,
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
      "version": 1
    },
    "spec": {
      "backends": [
        {
          "virtualService": {
            "virtualServiceName": "serviceA.svc.cluster.local"
          }
        }
      ],
    },
  },
}

```

```
    "listeners": [
      {
        "portMapping": {
          "port": 80,
          "protocol": "http"
        }
      }
    ],
    "serviceDiscovery": {
      "awsCloudMap": {
        "attributes": [
          {
            "key": "Environment",
            "value": "Testing"
          }
        ],
        "namespaceName": "namespace1",
        "serviceName": "serviceA"
      }
    },
    "status": {
      "status": "ACTIVE"
    },
    "virtualNodeName": "vnServiceA"
  }
}
```

For more information, see [Virtual Nodes](#) in the *AWS App Mesh User Guide*.

- For API details, see [CreateVirtualNode](#) in *AWS CLI Command Reference*.

create-virtual-router

The following code example shows how to use `create-virtual-router`.

AWS CLI

To create a new virtual router

The following `create-virtual-router` example uses a JSON input file to create a virtual router with a listener for HTTP using port 80.

```
aws appmesh create-virtual-router \  
  --cli-input-json file://create-virtual-router.json
```

Contents of create-virtual-router.json:

```
{  
  "meshName": "app1",  
  "spec": {  
    "listeners": [  
      {  
        "portMapping": {  
          "port": 80,  
          "protocol": "http"  
        }  
      }  
    ]  
  },  
  "virtualRouterName": "vrServiceB"  
}
```

Output:

```
{  
  "virtualRouter": {  
    "meshName": "app1",  
    "metadata": {  
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualRouter/  
vrServiceB",  
      "createdAt": 1563810546.59,  
      "lastUpdatedAt": 1563810546.59,  
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
      "version": 1  
    },  
    "spec": {  
      "listeners": [  
        {  
          "portMapping": {  
            "port": 80,  
            "protocol": "http"  
          }  
        }  
      ]  
    }  
  }  
}
```

```

    },
    "status": {
      "status": "ACTIVE"
    },
    "virtualRouterName": "vrServiceB"
  }
}

```

For more information, see [Virtual Routers](#) in the *AWS App Mesh User Guide*.

- For API details, see [CreateVirtualRouter](#) in *AWS CLI Command Reference*.

create-virtual-service

The following code example shows how to use `create-virtual-service`.

AWS CLI

Example 1: To create a new virtual service with a virtual node provider

The following `create-virtual-service` example uses a JSON input file to create a virtual service with a virtual node provider.

```

aws appmesh create-virtual-service \
  --cli-input-json file://create-virtual-service-virtual-node.json

```

Contents of `create-virtual-service-virtual-node.json`:

```

{
  "meshName": "app1",
  "spec": {
    "provider": {
      "virtualNode": {
        "virtualNodeName": "vnServiceA"
      }
    }
  },
  "virtualServiceName": "serviceA.svc.cluster.local"
}

```

Output:


```
{
  "virtualService": {
    "meshName": "app1",
    "metadata": {
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualService/
serviceA.svc.cluster.local",
      "createdAt": 1563810859.474,
      "lastUpdatedAt": 1563810967.179,
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
      "version": 2
    },
    "spec": {
      "provider": {
        "virtualNode": {
          "virtualNodeName": "vnServiceA"
        }
      }
    },
    "status": {
      "status": "ACTIVE"
    },
    "virtualServiceName": "serviceA.svc.cluster.local"
  }
}
```

For more information, see [Virtual Node](#) in the *AWS App Mesh User Guide*.

Example 2: To create a new virtual service with a virtual router provider

The following `create-virtual-service` example uses a JSON input file to create a virtual service with a virtual router provider.

```
aws appmesh create-virtual-service \
  --cli-input-json file://create-virtual-service-virtual-router.json
```

Contents of `create-virtual-service-virtual-router.json`:

```
{
  "meshName": "app1",
  "spec": {
    "provider": {
      "virtualRouter": {
```

```

        "virtualRouterName": "vrServiceB"
    }
}
},
"virtualServiceName": "serviceB.svc.cluster.local"
}

```

Output:

```

{
  "virtualService": {
    "meshName": "app1",
    "metadata": {
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualService/
serviceB.svc.cluster.local",
      "createdAt": 1563908363.999,
      "lastUpdatedAt": 1563908363.999,
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
      "version": 1
    },
    "spec": {
      "provider": {
        "virtualRouter": {
          "virtualRouterName": "vrServiceB"
        }
      }
    },
    "status": {
      "status": "ACTIVE"
    },
    "virtualServiceName": "serviceB.svc.cluster.local"
  }
}

```

For more information, see [Virtual Services](https://docs.aws.amazon.com/app-mesh/latest/userguide/virtual_services.html) in the *AWS App Mesh User Guide*

- For API details, see [CreateVirtualService](#) in *AWS CLI Command Reference*.

delete-mesh

The following code example shows how to use `delete-mesh`.

AWS CLI

To delete a service mesh

The following `delete-mesh` example deletes the specified service mesh.

```
aws appmesh delete-mesh \  
  --mesh-name app1
```

Output:

```
{  
  "mesh": {  
    "meshName": "app1",  
    "metadata": {  
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1",  
      "createdAt": 1563809909.282,  
      "lastUpdatedAt": 1563824981.248,  
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
      "version": 2  
    },  
    "spec": {  
      "egressFilter": {  
        "type": "ALLOW_ALL"  
      }  
    },  
    "status": {  
      "status": "DELETED"  
    }  
  }  
}
```

For more information, see [Service Meshes](#) in the *AWS App Mesh User Guide*.

- For API details, see [DeleteMesh](#) in *AWS CLI Command Reference*.

delete-route

The following code example shows how to use `delete-route`.

AWS CLI

To delete a route

The following `delete-route` example deletes the specified route.

```
aws appmesh delete-route \  
  --mesh-name app1 \  
  --virtual-router-name vrServiceB \  
  --route-name toVnServiceB-weighted
```

Output:

```
{  
  "route": {  
    "meshName": "app1",  
    "metadata": {  
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualRouter/  
vrServiceB/route/toVnServiceB-weighted",  
      "createdAt": 1563811384.015,  
      "lastUpdatedAt": 1563823915.936,  
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
      "version": 3  
    },  
    "routeName": "toVnServiceB-weighted",  
    "spec": {  
      "httpRoute": {  
        "action": {  
          "weightedTargets": [  
            {  
              "virtualNode": "vnServiceBv1",  
              "weight": 80  
            },  
            {  
              "virtualNode": "vnServiceBv2",  
              "weight": 20  
            }  
          ]  
        },  
        "match": {  
          "prefix": "/"  
        }  
      }  
    },  
    "status": {  
      "status": "DELETED"  
    }  
  },  
}
```

```

    "virtualRouterName": "vrServiceB"
  }
}

```

For more information, see [Routes](#) in the *AWS App Mesh User Guide*.

- For API details, see [DeleteRoute](#) in *AWS CLI Command Reference*.

delete-virtual-node

The following code example shows how to use `delete-virtual-node`.

AWS CLI

To delete a virtual node

The following `delete-virtual-node` example deletes the specified virtual node.

```

aws appmesh delete-virtual-node \
  --mesh-name app1 \
  --virtual-node-name vnServiceBv2

```

Output:

```

{
  "virtualNode": {
    "meshName": "app1",
    "metadata": {
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualNode/vnServiceBv2",
      "createdAt": 1563810117.297,
      "lastUpdatedAt": 1563824700.678,
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
      "version": 2
    },
    "spec": {
      "backends": [],
      "listeners": [
        {
          "portMapping": {
            "port": 80,
            "protocol": "http"
          }
        }
      ]
    }
  }
}

```

```

        }
      }
    ],
    "serviceDiscovery": {
      "dns": {
        "hostname": "serviceBv2.svc.cluster.local"
      }
    }
  },
  "status": {
    "status": "DELETED"
  },
  "virtualNodeName": "vnServiceBv2"
}
}

```

For more information, see [Virtual Nodes](#) in the *AWS App Mesh User Guide*.

- For API details, see [DeleteVirtualNode](#) in *AWS CLI Command Reference*.

delete-virtual-router

The following code example shows how to use `delete-virtual-router`.

AWS CLI

To delete a virtual router

The following `delete-virtual-router` example deletes the specified virtual router.

```

aws appmesh delete-virtual-router \
  --mesh-name app1 \
  --virtual-router-name vrServiceB

```

Output:

```

{
  "virtualRouter": {
    "meshName": "app1",
    "metadata": {
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualRouter/
vrServiceB",

```

```
    "createdAt": 1563810546.59,
    "lastUpdatedAt": 1563824253.467,
    "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "version": 3
  },
  "spec": {
    "listeners": [
      {
        "portMapping": {
          "port": 80,
          "protocol": "http"
        }
      }
    ]
  },
  "status": {
    "status": "DELETED"
  },
  "virtualRouterName": "vrServiceB"
}
}
```

For more information, see [Virtual Routers](#) in the *AWS App Mesh User Guide*.

- For API details, see [DeleteVirtualRouter](#) in *AWS CLI Command Reference*.

delete-virtual-service

The following code example shows how to use `delete-virtual-service`.

AWS CLI

To delete a virtual service

The following `delete-virtual-service` example deletes the specified virtual service.

```
aws appmesh delete-virtual-service \
  --mesh-name app1 \
  --virtual-service-name serviceB.svc.cluster.local
```

Output:

```
{
```

```

    "virtualService": {
      "meshName": "app1",
      "metadata": {
        "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualService/
serviceB.svc.cluster.local",
        "createdAt": 1563908363.999,
        "lastUpdatedAt": 1563913940.866,
        "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
        "version": 3
      },
      "spec": {},
      "status": {
        "status": "DELETED"
      },
      "virtualServiceName": "serviceB.svc.cluster.local"
    }
  }
}

```

For more information, see [Virtual Service](#) in the *AWS App Mesh User Guide*.

- For API details, see [DeleteVirtualService](#) in *AWS CLI Command Reference*.

describe-mesh

The following code example shows how to use describe-mesh.

AWS CLI

To describe a service mesh

The following describe-mesh example returns details about the specified service mesh.

```

aws appmesh describe-mesh \
  --mesh-name app1

```

Output:

```

{
  "mesh": {
    "meshName": "app1",
    "metadata": {
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1",

```



```

        "createdAt": 1563809909.282,
        "lastUpdatedAt": 1563809909.282,
        "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
        "version": 1
    },
    "spec": {},
    "status": {
        "status": "ACTIVE"
    }
}
}

```

For more information, see [Service Meshes](#) in the *AWS App Mesh User Guide*.

- For API details, see [DescribeMesh](#) in *AWS CLI Command Reference*.

describe-route

The following code example shows how to use `describe-route`.

AWS CLI

To describe a route

The following `describe-route` example returns details about the specified route.

```

aws appmesh describe-route \
  --mesh-name app1 \
  --virtual-router-name vrServiceB \
  --route-name toVnServiceB-weighted

```

Output:

```

{
  "route": {
    "meshName": "app1",
    "metadata": {
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualRouter/vrServiceB/route/toVnServiceB-weighted",
      "createdAt": 1563811384.015,
      "lastUpdatedAt": 1563811384.015,
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",

```

```
    "version": 1
  },
  "routeName": "toVnServiceB-weighted",
  "spec": {
    "httpRoute": {
      "action": {
        "weightedTargets": [
          {
            "virtualNode": "vnServiceBv1",
            "weight": 90
          },
          {
            "virtualNode": "vnServiceBv2",
            "weight": 10
          }
        ]
      },
      "match": {
        "prefix": "/"
      }
    }
  },
  "status": {
    "status": "ACTIVE"
  },
  "virtualRouterName": "vrServiceB"
}
```

For more information, see [Routes](#) in the *AWS App Mesh User Guide*.

- For API details, see [DescribeRoute](#) in *AWS CLI Command Reference*.

describe-virtual-node

The following code example shows how to use `describe-virtual-node`.

AWS CLI

To describe a virtual node

The following `describe-virtual-node` example returns details about the specified virtual node.

```
aws appmesh describe-virtual-node \  
  --mesh-name app1 \  
  --virtual-node-name vnServiceBv1
```

Output:

```
{  
  "virtualNode": {  
    "meshName": "app1",  
    "metadata": {  
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualNode/  
vnServiceBv1",  
      "createdAt": 1563810019.874,  
      "lastUpdatedAt": 1563810019.874,  
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
      "version": 1  
    },  
    "spec": {  
      "backends": [],  
      "listeners": [  
        {  
          "portMapping": {  
            "port": 80,  
            "protocol": "http"  
          }  
        }  
      ],  
      "serviceDiscovery": {  
        "dns": {  
          "hostname": "serviceBv1.svc.cluster.local"  
        }  
      }  
    },  
    "status": {  
      "status": "ACTIVE"  
    },  
    "virtualNodeName": "vnServiceBv1"  
  }  
}
```

For more information, see [Virtual Nodes](#) in the *AWS App Mesh User Guide*.

- For API details, see [DescribeVirtualNode](#) in *AWS CLI Command Reference*.

describe-virtual-router

The following code example shows how to use `describe-virtual-router`.

AWS CLI

To describe a virtual router

The following `describe-virtual-router` example returns details about the specified virtual router.

```
aws appmesh describe-virtual-router \
  --mesh-name app1 \
  --virtual-router-name vrServiceB
```

Output:

```
{
  "virtualRouter": {
    "meshName": "app1",
    "metadata": {
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualRouter/
vrServiceB",
      "createdAt": 1563810546.59,
      "lastUpdatedAt": 1563810546.59,
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
      "version": 1
    },
    "spec": {
      "listeners": [
        {
          "portMapping": {
            "port": 80,
            "protocol": "http"
          }
        }
      ]
    },
    "status": {
      "status": "ACTIVE"
    },
    "virtualRouterName": "vrServiceB"
  }
}
```

```
}
```

For more information, see [Virtual Routers](#) in the *AWS App Mesh User Guide*.

- For API details, see [DescribeVirtualRouter](#) in *AWS CLI Command Reference*.

describe-virtual-service

The following code example shows how to use `describe-virtual-service`.

AWS CLI

To describe a virtual service

The following `describe-virtual-service` example returns details about the specified virtual service.

```
aws appmesh describe-virtual-service \  
  --mesh-name app1 \  
  --virtual-service-name serviceB.svc.cluster.local
```

Output:

```
{  
  "virtualService": {  
    "meshName": "app1",  
    "metadata": {  
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualService/  
serviceB.svc.cluster.local",  
      "createdAt": 1563908363.999,  
      "lastUpdatedAt": 1563908363.999,  
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
      "version": 1  
    },  
    "spec": {  
      "provider": {  
        "virtualRouter": {  
          "virtualRouterName": "vrServiceB"  
        }  
      }  
    },  
    "status": {  
      "status": "ACTIVE"  
    }  
  }  
}
```

```
    },
    "virtualServiceName": "serviceB.svc.cluster.local"
  }
}
```

For more information, see [Virtual Services](#) in the *AWS App Mesh User Guide*.

- For API details, see [DescribeVirtualService](#) in *AWS CLI Command Reference*.

list-meshes

The following code example shows how to use `list-meshes`.

AWS CLI

To list service meshes

The following `list-meshes` example lists all of the service meshes in the current AWS Region.

```
aws appmesh list-meshes
```

Output:

```
{
  "meshes": [
    {
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1",
      "meshName": "app1"
    }
  ]
}
```

For more information, see [Service Meshes](#) in the *AWS App Mesh User Guide*.

- For API details, see [ListMeshes](#) in *AWS CLI Command Reference*.

list-routes

The following code example shows how to use `list-routes`.

AWS CLI

To list routes

The following `list-routes` example lists all of the routes for the specified virtual router.

```
aws appmesh list-routes \  
  --mesh-name app1 \  
  --virtual-router-name vrServiceB
```

Output:

```
{  
  "routes": [  
    {  
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualRouter/  
vrServiceB/route/toVnServiceB",  
      "meshName": "app1",  
      "routeName": "toVnServiceB-weighted",  
      "virtualRouterName": "vrServiceB"  
    }  
  ]  
}
```

For more information, see [Routes](#) in the *AWS App Mesh User Guide*.

- For API details, see [ListRoutes](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags for a resource

The following `list-tags-for-resource` example lists all of the tags assigned to the specified resource.

```
aws appmesh list-tags-for-resource \  
  --resource-arn arn:aws:appmesh:us-east-1:123456789012:mesh/app1
```

Output:

```
{
```

```
"tags": [  
  {  
    "key": "key1",  
    "value": "value1"  
  },  
  {  
    "key": "key2",  
    "value": "value2"  
  },  
  {  
    "key": "key3",  
    "value": "value3"  
  }  
]  
}
```

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

list-virtual-nodes

The following code example shows how to use `list-virtual-nodes`.

AWS CLI

To list virtual nodes

The following `list-virtual-nodes` example lists all of the virtual nodes in the specified service mesh.

```
aws appmesh list-virtual-nodes \  
  --mesh-name app1
```

Output:

```
{  
  "virtualNodes": [  
    {  
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualNode/  
vnServiceBv1",  
      "meshName": "app1",  
      "virtualNodeName": "vnServiceBv1"  
    },  
  ],  
}
```



```
{
  "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualNode/
vnServiceBv2",
  "meshName": "app1",
  "virtualNodeName": "vnServiceBv2"
}
]
```

For more information, see [Virtual Nodes](#) in the *AWS App Mesh User Guide*.

- For API details, see [ListVirtualNodes](#) in *AWS CLI Command Reference*.

list-virtual-routers

The following code example shows how to use `list-virtual-routers`.

AWS CLI

To list virtual routers

The following `list-virtual-routers` example lists all of the virtual routers in the specified service mesh.

```
aws appmesh list-virtual-routers \
  --mesh-name app1
```

Output:

```
{
  "virtualRouters": [
    {
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualRouter/
vrServiceB",
      "meshName": "app1",
      "virtualRouterName": "vrServiceB"
    }
  ]
}
```

For more information, see [Virtual Routers](#) in the *AWS App Mesh User Guide*.

- For API details, see [ListVirtualRouters](#) in *AWS CLI Command Reference*.

list-virtual-services

The following code example shows how to use `list-virtual-services`.

AWS CLI

To list virtual services

The following `list-virtual-services` example lists all of the virtual services in the specified service mesh.

```
aws appmesh list-virtual-services \
  --mesh-name app1
```

Output:

```
{
  "virtualServices": [
    {
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualService/
serviceA.svc.cluster.local",
      "meshName": "app1",
      "virtualServiceName": "serviceA.svc.cluster.local"
    },
    {
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualService/
serviceB.svc.cluster.local",
      "meshName": "app1",
      "virtualServiceName": "serviceB.svc.cluster.local"
    }
  ]
}
```

For more information, see [Virtual Services](#) in the *AWS App Mesh User Guide*.

- For API details, see [ListVirtualServices](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To tag a resource

The following `tag-resource` example adds the tag `key1` with the value `value1` to the specified resource.

```
aws appmesh tag-resource \  
  --resource-arn arn:aws:appmesh:us-east-1:123456789012:mesh/app1 \  
  --tags key=key1,value=value1
```

This command produces no output.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To untag a resource

The following `untag-resource` example removes a tag with the key `key1` from the specified resource.

```
aws appmesh untag-resource \  
  --resource-arn arn:aws:appmesh:us-east-1:123456789012:mesh/app1 \  
  --tag-keys key1
```

This command produces no output.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-mesh

The following code example shows how to use `update-mesh`.

AWS CLI

To update a service mesh

The following `update-mesh` example uses a JSON input file to update a service mesh to allow all external egress traffic to be forwarded through the Envoy proxy untouched.

```
aws appmesh update-mesh \  
  --cli-input-json file://update-mesh.json
```

Contents of `update-mesh.json`:

```
{  
  "meshName": "app1",  
  "spec": {  
    "egressFilter": {  
      "type": "ALLOW_ALL"  
    }  
  }  
}
```

Output:

```
{  
  "mesh": {  
    "meshName": "app1",  
    "metadata": {  
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1",  
      "createdAt": 1563809909.282,  
      "lastUpdatedAt": 1563812829.687,  
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
      "version": 2  
    },  
    "spec": {  
      "egressFilter": {  
        "type": "ALLOW_ALL"  
      }  
    },  
    "status": {  
      "status": "ACTIVE"  
    }  
  }  
}
```

For more information, see [Service Meshes](#) in the *AWS App Mesh User Guide*.

- For API details, see [UpdateMesh](#) in *AWS CLI Command Reference*.

update-route

The following code example shows how to use `update-route`.

AWS CLI

To update a route

The following `update-route` example uses a JSON input file to update the weights for a route.

```
aws appmesh update-route \  
  --cli-input-json file://update-route-weighted.json
```

Contents of `update-route-weighted.json`:

```
{  
  "meshName": "app1",  
  "routeName": "toVnServiceB-weighted",  
  "spec": {  
    "httpRoute": {  
      "action": {  
        "weightedTargets": [  
          {  
            "virtualNode": "vnServiceBv1",  
            "weight": 80  
          },  
          {  
            "virtualNode": "vnServiceBv2",  
            "weight": 20  
          }  
        ]  
      },  
      "match": {  
        "prefix": "/"  
      }  
    }  
  },  
  "virtualRouterName": "vrServiceB"  
}
```

Output:

```
{
  "route": {
    "meshName": "app1",
    "metadata": {
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualRouter/vrServiceB/route/toVnServiceB-weighted",
      "createdAt": 1563811384.015,
      "lastUpdatedAt": 1563819600.022,
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
      "version": 2
    },
    "routeName": "toVnServiceB-weighted",
    "spec": {
      "httpRoute": {
        "action": {
          "weightedTargets": [
            {
              "virtualNode": "vnServiceBv1",
              "weight": 80
            },
            {
              "virtualNode": "vnServiceBv2",
              "weight": 20
            }
          ]
        },
        "match": {
          "prefix": "/"
        }
      }
    },
    "status": {
      "status": "ACTIVE"
    },
    "virtualRouterName": "vrServiceB"
  }
}
```

For more information, see [Routes](#) in the *AWS App Mesh User Guide*.

- For API details, see [UpdateRoute](#) in *AWS CLI Command Reference*.

update-virtual-node

The following code example shows how to use `update-virtual-node`.

AWS CLI

To update a virtual node

The following `update-virtual-node` example uses a JSON input file to add a health check to a virtual node.

```
aws appmesh update-virtual-node \  
  --cli-input-json file://update-virtual-node.json
```

Contents of `update-virtual-node.json`:

```
{  
  "clientToken": "500",  
  "meshName": "app1",  
  "spec": {  
    "listeners": [  
      {  
        "healthCheck": {  
          "healthyThreshold": 5,  
          "intervalMillis": 10000,  
          "path": "/",  
          "port": 80,  
          "protocol": "http",  
          "timeoutMillis": 3000,  
          "unhealthyThreshold": 3  
        },  
        "portMapping": {  
          "port": 80,  
          "protocol": "http"  
        }  
      }  
    ],  
    "serviceDiscovery": {  
      "dns": {  
        "hostname": "serviceBv1.svc.cluster.local"  
      }  
    }  
  },  
}
```

```
"virtualNodeName": "vnServiceBv1"
}
```

Output:

```
{
  "virtualNode": {
    "meshName": "app1",
    "metadata": {
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualNode/vnServiceBv1",
      "createdAt": 1563810019.874,
      "lastUpdatedAt": 1563819234.825,
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
      "version": 2
    },
    "spec": {
      "listeners": [
        {
          "healthCheck": {
            "healthyThreshold": 5,
            "intervalMillis": 10000,
            "path": "/",
            "port": 80,
            "protocol": "http",
            "timeoutMillis": 3000,
            "unhealthyThreshold": 3
          },
          "portMapping": {
            "port": 80,
            "protocol": "http"
          }
        }
      ],
      "serviceDiscovery": {
        "dns": {
          "hostname": "serviceBv1.svc.cluster.local"
        }
      }
    },
    "status": {
      "status": "ACTIVE"
    }
  },
}
```



```
    "virtualNodeName": "vnServiceBv1"
  }
}
```

For more information, see [Virtual Nodes](#) in the *AWS App Mesh User Guide*.

- For API details, see [UpdateVirtualNode](#) in *AWS CLI Command Reference*.

update-virtual-router

The following code example shows how to use `update-virtual-router`.

AWS CLI

To update a virtual router

The following `update-virtual-router` example uses a JSON input file to update a virtual router listener port.

```
aws appmesh update-virtual-router \
  --cli-input-json file://update-virtual-router.json
```

Contents of `update-virtual-router.json`:

```
{
  "meshName": "app1",
  "spec": {
    "listeners": [
      {
        "portMapping": {
          "port": 8080,
          "protocol": "http"
        }
      }
    ]
  },
  "virtualRouterName": "vrServiceB"
}
```

Output:

```
{
```

```
"virtualRouter": {
  "meshName": "app1",
  "metadata": {
    "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualRouter/
vrServiceB",
    "createdAt": 1563810546.59,
    "lastUpdatedAt": 1563819431.352,
    "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "version": 2
  },
  "spec": {
    "listeners": [
      {
        "portMapping": {
          "port": 8080,
          "protocol": "http"
        }
      }
    ]
  },
  "status": {
    "status": "ACTIVE"
  },
  "virtualRouterName": "vrServiceB"
}
}
```

For more information, see [Virtual Routers](#) in the *AWS App Mesh User Guide*.

- For API details, see [UpdateVirtualRouter](#) in *AWS CLI Command Reference*.

update-virtual-service

The following code example shows how to use `update-virtual-service`.

AWS CLI

To update a virtual service

The following `update-virtual-service` example uses a JSON input file to update a virtual service to use a virtual router provider.

```
aws appmesh update-virtual-service \  

```

```
--cli-input-json file://update-virtual-service.json
```

Contents of update-virtual-service.json:

```
{
  "meshName": "app1",
  "spec": {
    "provider": {
      "virtualRouter": {
        "virtualRouterName": "vrServiceA"
      }
    }
  },
  "virtualServiceName": "serviceA.svc.cluster.local"
}
```

Output:

```
{
  "virtualService": {
    "meshName": "app1",
    "metadata": {
      "arn": "arn:aws:appmesh:us-east-1:123456789012:mesh/app1/virtualService/
serviceA.svc.cluster.local",
      "createdAt": 1563810859.474,
      "lastUpdatedAt": 1563820257.411,
      "uid": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
      "version": 3
    },
    "spec": {
      "provider": {
        "virtualRouter": {
          "virtualRouterName": "vrServiceA"
        }
      }
    },
    "status": {
      "status": "ACTIVE"
    },
    "virtualServiceName": "serviceA.svc.cluster.local"
  }
}
```

For more information, see [Virtual Services](#) in the *AWS App Mesh User Guide*.

- For API details, see [UpdateVirtualService](#) in *AWS CLI Command Reference*.

App Runner examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with App Runner.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-custom-domain

The following code example shows how to use `associate-custom-domain`.

AWS CLI

To associate a domain name and the www subdomain with a service

The following `associate-custom-domain` example associates a custom domain name that you control with an App Runner service. The domain name is the root domain `example.com`, including the special-case subdomain `www.example.com`.

```
aws apprunner associate-custom-domain \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
  "DomainName": "example.com",
  "EnableWWWSubdomain": true
}
```

Output:

```
{
  "CustomDomain": {
    "CertificateValidationRecords": [
      {
        "Name": "_70d3f50a94f7c72dc28784cf55db2f6b.example.com",
        "Status": "PENDING_VALIDATION",
        "Type": "CNAME",
        "Value": "_1270c137383c6307b6832db02504c4b0.bsgbmzkfwj.acm-
validations.aws."
      },
      {
        "Name": "_287870d3f50a94f7c72dc4cf55db2f6b.www.example.com",
        "Status": "PENDING_VALIDATION",
        "Type": "CNAME",
        "Value": "_832db01270c137383c6307b62504c4b0.mzkbsgbfwj.acm-
validations.aws."
      }
    ],
    "DomainName": "example.com",
    "EnableWWWSubdomain": true,
    "Status": "CREATING"
  },
  "DNSTarget": "psbqam834h.us-east-1.awsapprunner.com",
  "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa"
}
```

- For API details, see [AssociateCustomDomain](#) in *AWS CLI Command Reference*.

create-auto-scaling-configuration

The following code example shows how to use create-auto-scaling-configuration.

AWS CLI

To create a high availability auto scaling configuration

The following `create-auto-scaling-configuration` example creates an auto scaling configuration optimized for high availability by setting `MinSize` to 5. With this configuration, App Runner attempts to spread your service instances over the most Availability Zones possible, up to five, depending on the AWS Region.

The call returns an `AutoScalingConfiguration` object with the other settings set to their defaults. In the example, this is the first call to create a configuration named `high-availability`. The revision is set to 1, and it's the latest revision.

```
aws apprunner create-auto-scaling-configuration \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "AutoScalingConfigurationName": "high-availability",  
  "MinSize": 5  
}
```

Output:

```
{  
  "AutoScalingConfiguration": {  
    "AutoScalingConfigurationArn": "arn:aws:apprunner:us-  
east-1:123456789012:autoscalingconfiguration/high-  
availability/1/2f50e7656d7819fead0f59672e68042e",  
    "AutoScalingConfigurationName": "high-availability",  
    "AutoScalingConfigurationRevision": 1,  
    "CreatedAt": "2020-11-03T00:29:17Z",  
    "Latest": true,  
    "Status": "ACTIVE",  
    "MaxConcurrency": 100,  
    "MaxSize": 50,  
    "MinSize": 5  
  }  
}
```

- For API details, see [CreateAutoScalingConfiguration](#) in *AWS CLI Command Reference*.

create-connection

The following code example shows how to use `create-connection`.

AWS CLI

To create a GitHub connection

The following `create-connection` example creates a connection to a private GitHub code repository. The connection status after a successful call is `PENDING_HANDSHAKE`. This is because an authentication handshake with the provider still hasn't happened. Complete the handshake using the App Runner console.

```
aws apprunner create-connection \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "ConnectionName": "my-github-connection",  
  "ProviderType": "GITHUB"  
}
```

Output:

```
{  
  "Connection": {  
    "ConnectionArn": "arn:aws:apprunner:us-east-1:123456789012:connection/my-github-connection",  
    "ConnectionName": "my-github-connection",  
    "Status": "PENDING_HANDSHAKE",  
    "CreatedAt": "2020-11-03T00:32:51Z",  
    "ProviderType": "GITHUB"  
  }  
}
```

For more information, see [Managing App Runner connections](#) in the *AWS App Runner Developer Guide*.

- For API details, see [CreateConnection](#) in *AWS CLI Command Reference*.

create-service

The following code example shows how to use `create-service`.

AWS CLI

Example 1: To create a source code repository service

The following `create-service` example creates an App Runner service based on a Python source code repository.

```
aws apprunner create-service \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "ServiceName": "python-app",  
  "SourceConfiguration": {  
    "AuthenticationConfiguration": {  
      "ConnectionArn": "arn:aws:apprunner:us-east-1:123456789012:connection/  
my-github-connection/e7656250f67242d7819feade6800f59e"  
    },  
    "AutoDeploymentsEnabled": true,  
    "CodeRepository": {  
      "RepositoryUrl": "https://github.com/my-account/python-hello",  
      "SourceCodeVersion": {  
        "Type": "BRANCH",  
        "Value": "main"  
      },  
      "CodeConfiguration": {  
        "ConfigurationSource": "API",  
        "CodeConfigurationValues": {  
          "Runtime": "PYTHON_3",  
          "BuildCommand": "pip install -r requirements.txt",  
          "StartCommand": "python server.py",  
          "Port": "8080",  
          "RuntimeEnvironmentVariables": [  
            {  
              "NAME": "Jane"  
            }  
          ]  
        }  
      }  
    }  
  }  
}
```



```

    }
  ]
}
},
"InstanceConfiguration": {
  "CPU": "1 vCPU",
  "Memory": "3 GB"
}
}

```

Output:

```

{
  "OperationId": "17fe9f55-7e91-4097-b243-fcabbb69a4cf",
  "Service": {
    "CreatedAt": "2020-11-20T19:05:25Z",
    "UpdatedAt": "2020-11-20T19:05:25Z",
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
    "ServiceId": "8fe1e10304f84fd2b0df550fe98a71fa",
    "ServiceName": "python-app",
    "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
    "SourceConfiguration": {
      "AuthenticationConfiguration": {
        "ConnectionArn": "arn:aws:apprunner:us-
east-1:123456789012:connection/my-github-connection/
e7656250f67242d7819feade6800f59e"
      },
      "AutoDeploymentsEnabled": true,
      "CodeRepository": {
        "CodeConfiguration": {
          "CodeConfigurationValues": {
            "BuildCommand": "pip install -r requirements.txt",
            "Port": "8080",
            "Runtime": "PYTHON_3",
            "RuntimeEnvironmentVariables": [
              {
                "NAME": "Jane"
              }
            ]
          },
          "StartCommand": "python server.py"
        }
      }
    }
  }
}

```

```

        },
        "ConfigurationSource": "Api"
    },
    "RepositoryUrl": "https://github.com/my-account/python-hello",
    "SourceCodeVersion": {
        "Type": "BRANCH",
        "Value": "main"
    }
}
},
"Status": "OPERATION_IN_PROGRESS",
"InstanceConfiguration": {
    "CPU": "1 vCPU",
    "Memory": "3 GB"
}
}
}

```

Example 2: To create a source code repository service

The following `create-service` example creates an App Runner service based on a Python source code repository.

```
aws apprunner create-service \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```

{
  "ServiceName": "python-app",
  "SourceConfiguration": {
    "AuthenticationConfiguration": {
      "ConnectionArn": "arn:aws:apprunner:us-east-1:123456789012:connection/my-github-connection/e7656250f67242d7819feade6800f59e"
    },
    "AutoDeploymentsEnabled": true,
    "CodeRepository": {
      "RepositoryUrl": "https://github.com/my-account/python-hello",
      "SourceCodeVersion": {
        "Type": "BRANCH",
        "Value": "main"
      }
    },
    "CodeConfiguration": {

```

```

        "ConfigurationSource": "API",
        "CodeConfigurationValues": {
            "Runtime": "PYTHON_3",
            "BuildCommand": "pip install -r requirements.txt",
            "StartCommand": "python server.py",
            "Port": "8080",
            "RuntimeEnvironmentVariables": [
                {
                    "NAME": "Jane"
                }
            ]
        }
    },
    "InstanceConfiguration": {
        "CPU": "1 vCPU",
        "Memory": "3 GB"
    }
}

```

Output:

```

{
  "OperationId": "17fe9f55-7e91-4097-b243-fcabbb69a4cf",
  "Service": {
    "CreatedAt": "2020-11-20T19:05:25Z",
    "UpdatedAt": "2020-11-20T19:05:25Z",
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
    "ServiceId": "8fe1e10304f84fd2b0df550fe98a71fa",
    "ServiceName": "python-app",
    "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
    "SourceConfiguration": {
      "AuthenticationConfiguration": {
        "ConnectionArn": "arn:aws:apprunner:us-
east-1:123456789012:connection/my-github-connection/
e7656250f67242d7819feade6800f59e"
      },
      "AutoDeploymentsEnabled": true,
      "CodeRepository": {
        "CodeConfiguration": {
          "CodeConfigurationValues": {

```



```

    "AutoDeploymentsEnabled": true,
    "ImageRepository": {
      "ImageIdentifier": "123456789012.dkr.ecr.us-east-1.amazonaws.com/golang-
app:latest",
      "ImageConfiguration": {
        "Port": "8080",
        "RuntimeEnvironmentVariables": [
          {
            "NAME": "Jane"
          }
        ]
      },
      "ImageRepositoryType": "ECR"
    }
  },
  "InstanceConfiguration": {
    "CPU": "1 vCPU",
    "Memory": "3 GB"
  }
}

```

Output:

```

{
  "OperationId": "17fe9f55-7e91-4097-b243-fcabbb69a4cf",
  "Service": {
    "CreatedAt": "2020-11-06T23:15:30Z",
    "UpdatedAt": "2020-11-06T23:15:30Z",
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/golang-
container-app/51728f8a20ce46d39b25398a6c8e9d1a",
    "ServiceId": "51728f8a20ce46d39b25398a6c8e9d1a",
    "ServiceName": "golang-container-app",
    "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
    "SourceConfiguration": {
      "AuthenticationConfiguration": {
        "AccessRoleArn": "arn:aws:iam::123456789012:role/my-ecr-role"
      },
      "AutoDeploymentsEnabled": true,
      "ImageRepository": {
        "ImageIdentifier": "123456789012.dkr.ecr.us-east-1.amazonaws.com/
golang-app:latest",
        "ImageConfiguration": {
          "Port": "8080",

```

```

        "RuntimeEnvironmentVariables": [
            {
                "NAME": "Jane"
            }
        ],
        "ImageRepositoryType": "ECR"
    },
    "Status": "OPERATION_IN_PROGRESS",
    "InstanceConfiguration": {
        "CPU": "1 vCPU",
        "Memory": "3 GB"
    }
}
}

```

- For API details, see [CreateService](#) in *AWS CLI Command Reference*.

delete-auto-scaling-configuration

The following code example shows how to use `delete-auto-scaling-configuration`.

AWS CLI

Example 1: To delete the latest active revision of an auto scaling configuration

The following `delete-auto-scaling-configuration` example deletes the latest active revision of an App Runner auto scaling configuration. To delete the latest active revision, specify an Amazon Resource Name (ARN) that ends with the configuration name, without the revision component.

In the example, two revisions exist before this action. Therefore, revision 2 (the latest) is deleted. However, it now shows `"Latest": false`, because, after being deleted, it isn't the latest active revision anymore.

```
aws apprunner delete-auto-scaling-configuration \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
```

```
"AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-1:123456789012:autoscalingconfiguration/high-availability"
}
```

Output:

```
{
  "AutoScalingConfiguration": {
    "AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-1:123456789012:autoscalingconfiguration/high-availability/2/
e76562f50d78042e819fead0f59672e6",
    "AutoScalingConfigurationName": "high-availability",
    "AutoScalingConfigurationRevision": 2,
    "CreatedAt": "2021-02-25T17:42:59Z",
    "DeletedAt": "2021-03-02T08:07:06Z",
    "Latest": false,
    "Status": "INACTIVE",
    "MaxConcurrency": 30,
    "MaxSize": 90,
    "MinSize": 5
  }
}
```

Example 2: To delete a specific revision of an auto scaling configuration

The following `delete-auto-scaling-configuration` example deletes a specific revision of an App Runner auto scaling configuration. To delete a specific revision, specify an ARN that includes the revision number.

In the example, several revisions exist before this action. The action deletes revision 1.

```
aws apprunner delete-auto-scaling-configuration \
--cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-1:123456789012:autoscalingconfiguration/high-availability/1"
}
```

Output:

```
{
  "AutoScalingConfiguration": {
    "AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-1:123456789012:autoscalingconfiguration/high-
availability/1/2f50e7656d7819fead0f59672e68042e",
    "AutoScalingConfigurationName": "high-availability",
    "AutoScalingConfigurationRevision": 1,
    "CreatedAt": "2020-11-03T00:29:17Z",
    "DeletedAt": "2021-03-02T08:07:06Z",
    "Latest": false,
    "Status": "INACTIVE",
    "MaxConcurrency": 100,
    "MaxSize": 50,
    "MinSize": 5
  }
}
```

- For API details, see [DeleteAutoScalingConfiguration](#) in *AWS CLI Command Reference*.

delete-connection

The following code example shows how to use `delete-connection`.

AWS CLI

To delete a connection

The following `delete-connection` example deletes an App Runner connection. The connection status after a successful call is `DELETED`. This is because the connection is no longer available.

```
aws apprunner delete-connection \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "ConnectionArn": "arn:aws:apprunner:us-east-1:123456789012:connection/my-github-
connection"
}
```


Output:

```
{
  "Connection": {
    "ConnectionArn": "arn:aws:apprunner:us-east-1:123456789012:connection/my-github-connection",
    "ConnectionName": "my-github-connection",
    "Status": "DELETED",
    "CreatedAt": "2020-11-03T00:32:51Z",
    "ProviderType": "GITHUB"
  }
}
```

- For API details, see [DeleteConnection](#) in *AWS CLI Command Reference*.

delete-service

The following code example shows how to use `delete-service`.

AWS CLI**To delete a service**

The following `delete-service` example deletes an App Runner service.

```
aws apprunner delete-service \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-app/8fe1e10304f84fd2b0df550fe98a71fa"
}
```

Output:

```
{
  "OperationId": "17fe9f55-7e91-4097-b243-fcabbb69a4cf",
  "Service": {
    "CreatedAt": "2020-11-20T19:05:25Z",
```

```

    "UpdatedAt": "2020-11-20T19:05:25Z",
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
    "ServiceId": "8fe1e10304f84fd2b0df550fe98a71fa",
    "ServiceName": "python-app",
    "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
    "SourceConfiguration": {
      "AuthenticationConfiguration": {
        "ConnectionArn": "arn:aws:apprunner:us-
east-1:123456789012:connection/my-github-connection/
e7656250f67242d7819feade6800f59e"
      },
      "AutoDeploymentsEnabled": true,
      "CodeRepository": {
        "CodeConfiguration": {
          "CodeConfigurationValues": {
            "BuildCommand": "pip install -r requirements.txt",
            "Port": "8080",
            "Runtime": "PYTHON_3",
            "RuntimeEnvironmentVariables": [
              {
                "NAME": "Jane"
              }
            ],
            "StartCommand": "python server.py"
          },
          "ConfigurationSource": "Api"
        },
        "RepositoryUrl": "https://github.com/my-account/python-hello",
        "SourceCodeVersion": {
          "Type": "BRANCH",
          "Value": "main"
        }
      }
    },
    "Status": "OPERATION_IN_PROGRESS",
    "InstanceConfiguration": {
      "CPU": "1 vCPU",
      "Memory": "3 GB"
    }
  }
}

```

- For API details, see [DeleteService](#) in *AWS CLI Command Reference*.

describe-auto-scaling-configuration

The following code example shows how to use `describe-auto-scaling-configuration`.

AWS CLI

Example 1: To describe the latest active revision of an auto scaling configuration

The following `describe-auto-scaling-configuration` example gets a description of the latest active revision of an App Runner auto scaling configuration. To describe the latest active revision, specify an ARN that ends with the configuration name, without the revision component.

In the example, two revisions exist. Therefore, revision 2 (the latest) is described. The resulting object shows `"Latest": true`.

```
aws apprunner describe-auto-scaling-configuration \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "AutoScalingConfigurationArn": "arn:aws:apprunner:us-  
east-1:123456789012:autoscalingconfiguration/high-availability"  
}
```

Output:

```
{  
  "AutoScalingConfiguration": {  
    "AutoScalingConfigurationArn": "arn:aws:apprunner:us-  
east-1:123456789012:autoscalingconfiguration/high-availability/2/  
e76562f50d78042e819fead0f59672e6",  
    "AutoScalingConfigurationName": "high-availability",  
    "AutoScalingConfigurationRevision": 2,  
    "CreatedAt": "2021-02-25T17:42:59Z",  
    "Latest": true,  
    "Status": "ACTIVE",  
    "MaxConcurrency": 30,  
    "MaxSize": 90,  
    "MinSize": 5
```

```
}  
}
```

Example 2: To describe a specific revision of an auto scaling configuration

The following `describe-auto-scaling-configuration` example get a description of a specific revision of an App Runner auto scaling configuration. To describe a specific revision, specify an ARN that includes the revision number.

In the example, several revisions exist and revision 1 is queried. The resulting object shows `"Latest": false`.

```
aws apprunner describe-auto-scaling-configuration \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "AutoScalingConfigurationArn": "arn:aws:apprunner:us-  
east-1:123456789012:autoscalingconfiguration/high-availability/1"  
}
```

Output:

```
{  
  "AutoScalingConfiguration": {  
    "AutoScalingConfigurationArn": "arn:aws:apprunner:us-  
east-1:123456789012:autoscalingconfiguration/high-  
availability/1/2f50e7656d7819fead0f59672e68042e",  
    "AutoScalingConfigurationName": "high-availability",  
    "AutoScalingConfigurationRevision": 1,  
    "CreatedAt": "2020-11-03T00:29:17Z",  
    "Latest": false,  
    "Status": "ACTIVE",  
    "MaxConcurrency": 100,  
    "MaxSize": 50,  
    "MinSize": 5  
  }  
}
```

- For API details, see [DescribeAutoScalingConfiguration](#) in *AWS CLI Command Reference*.

describe-custom-domains

The following code example shows how to use describe-custom-domains.

AWS CLI

To get descriptions of custom domain names associated with a service

The following describe-custom-domains example get descriptions and status of the custom domain names associated with an App Runner service.

```
aws apprunner describe-custom-domains \  
  --cli-input-json file://input.json
```

Contents of input.json:

```
{  
  "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-  
app/8fe1e10304f84fd2b0df550fe98a71fa",  
  "DomainName": "example.com",  
  "EnableWWWSubdomain": true  
}
```

Output:

```
{  
  "CustomDomains": [  
    {  
      "CertificateValidationRecords": [  
        {  
          "Name": "_70d3f50a94f7c72dc28784cf55db2f6b.example.com",  
          "Status": "PENDING_VALIDATION",  
          "Type": "CNAME",  
          "Value": "_1270c137383c6307b6832db02504c4b0.bsgbmzkfwj.acm-  
validations.aws."  
        },  
        {  
          "Name": "_287870d3f50a94f7c72dc4cf55db2f6b.www.example.com",  
          "Status": "PENDING_VALIDATION",  
          "Type": "CNAME",  
          "Value": "_832db01270c137383c6307b62504c4b0.mzkbsgbfwj.acm-  
validations.aws."  
        }  
      ]  
    }  
  ]  
}
```

```

    }
  ],
  "DomainName": "example.com",
  "EnableWWWSubdomain": true,
  "Status": "PENDING_CERTIFICATE_DNS_VALIDATION"
},
{
  "CertificateValidationRecords": [
    {
      "Name": "_a94f784c70d3f507c72dc28f55db2f6b.deals.example.com",
      "Status": "SUCCESS",
      "Type": "CNAME",
      "Value": "_2db02504c1270c137383c6307b6834b0.bsgbmzkfwj.acm-
validations.aws."
    }
  ],
  "DomainName": "deals.example.com",
  "EnableWWWSubdomain": false,
  "Status": "ACTIVE"
}
],
  "DNSTarget": "psbqam834h.us-east-1.awsapprunner.com",
  "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa"
}

```

- For API details, see [DescribeCustomDomains](#) in *AWS CLI Command Reference*.

describe-service

The following code example shows how to use describe-service.

AWS CLI

To describe a service

The following describe-service example gets a description of an App Runner service.

```
aws apprunner describe-service \
  --cli-input-json file://input.json
```

Contents of input.json:

```
{
  "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa"
}
```

Output:

```
{
  "Service": {
    "CreatedAt": "2020-11-20T19:05:25Z",
    "UpdatedAt": "2020-11-20T19:05:25Z",
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
    "ServiceId": "8fe1e10304f84fd2b0df550fe98a71fa",
    "ServiceName": "python-app",
    "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
    "SourceConfiguration": {
      "AuthenticationConfiguration": {
        "ConnectionArn": "arn:aws:apprunner:us-
east-1:123456789012:connection/my-github-connection/
e7656250f67242d7819feade6800f59e"
      },
      "AutoDeploymentsEnabled": true,
      "CodeRepository": {
        "CodeConfiguration": {
          "CodeConfigurationValues": {
            "BuildCommand": "pip install -r requirements.txt",
            "Port": "8080",
            "Runtime": "PYTHON_3",
            "RuntimeEnvironmentVariables": [
              {
                "NAME": "Jane"
              }
            ]
          },
          "StartCommand": "python server.py"
        },
        "ConfigurationSource": "Api"
      },
      "RepositoryUrl": "https://github.com/my-account/python-hello",
      "SourceCodeVersion": {
        "Type": "BRANCH",
        "Value": "main"
      }
    }
  }
}
```

```

    }
  },
  "Status": "RUNNING",
  "InstanceConfiguration": {
    "CPU": "1 vCPU",
    "Memory": "3 GB"
  }
}
}

```

- For API details, see [DescribeService](#) in *AWS CLI Command Reference*.

disassociate-custom-domain

The following code example shows how to use `disassociate-custom-domain`.

AWS CLI

To disassociate a domain name from a service

The following `disassociate-custom-domain` example disassociates the domain `example.com` from an App Runner service. The call also disassociates the subdomain `www.example.com` that was associated together with the root domain.

```
aws apprunner disassociate-custom-domain \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
  "DomainName": "example.com"
}
```

Output:

```
{
  "CustomDomain": {
    "CertificateValidationRecords": [
      {
```



```

        "Name": "_70d3f50a94f7c72dc28784cf55db2f6b.example.com",
        "Status": "PENDING_VALIDATION",
        "Type": "CNAME",
        "Value": "_1270c137383c6307b6832db02504c4b0.bsgbmzkfwj.acm-
validations.aws."
    },
    {
        "Name": "_287870d3f50a94f7c72dc4cf55db2f6b.www.example.com",
        "Status": "PENDING_VALIDATION",
        "Type": "CNAME",
        "Value": "_832db01270c137383c6307b62504c4b0.mzkbsgbfwj.acm-
validations.aws."
    }
],
"DomainName": "example.com",
"EnableWWWSubdomain": true,
"Status": "DELETING"
},
"DNSTarget": "psbqam834h.us-east-1.awsapprunner.com",
"ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa"
}

```

- For API details, see [DisassociateCustomDomain](#) in *AWS CLI Command Reference*.

list-auto-scaling-configurations

The following code example shows how to use `list-auto-scaling-configurations`.

AWS CLI

To get a paginated listing of App Runner auto scaling configurations

The following `list-auto-scaling-configurations` example lists all App Runner auto scaling configurations in your AWS account. Up to five auto scaling configurations are listed in each response. `AutoScalingConfigurationName` and `LatestOnly` aren't specified. Their defaults cause the latest revision of all active configurations to be listed.

In this example, the response includes two results and there aren't additional ones, so no `NextToken` is returned.

```
aws apprunner list-auto-scaling-configurations \
```

```
--cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "MaxResults": 5
}
```

Output:

```
{
  "AutoScalingConfigurationSummaryList": [
    {
      "AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-1:123456789012:autoscalingconfiguration/high-availability/2/
e76562f50d78042e819fead0f59672e6",
      "AutoScalingConfigurationName": "high-availability",
      "AutoScalingConfigurationRevision": 2
    },
    {
      "AutoScalingConfigurationArn": "arn:aws:apprunner:us-
east-1:123456789012:autoscalingconfiguration/low-
cost/1/50d7804e7656fead0f59672e62f2e819",
      "AutoScalingConfigurationName": "low-cost",
      "AutoScalingConfigurationRevision": 1
    }
  ]
}
```

- For API details, see [ListAutoScalingConfigurations](#) in *AWS CLI Command Reference*.

list-connections

The following code example shows how to use `list-connections`.

AWS CLI

Example 1: To list all connections

The following `list-connections` example lists all App Runner connections in the AWS account.

```
aws apprunner list-connections
```

Output:

```
{
  "ConnectionSummaryList": [
    {
      "ConnectionArn": "arn:aws:apprunner:us-east-1:123456789012:connection/
my-github-connection",
      "ConnectionName": "my-github-connection",
      "Status": "AVAILABLE",
      "CreatedAt": "2020-11-03T00:32:51Z",
      "ProviderType": "GITHUB"
    },
    {
      "ConnectionArn": "arn:aws:apprunner:us-east-1:123456789012:connection/
my-github-org-connection",
      "ConnectionName": "my-github-org-connection",
      "Status": "AVAILABLE",
      "CreatedAt": "2020-11-03T02:54:17Z",
      "ProviderType": "GITHUB"
    }
  ]
}
```

Example 2: To list a connection by name

The following `list-connections` example lists a connection by its name.

```
aws apprunner list-connections \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "ConnectionName": "my-github-org-connection"
}
```

Output:

```
{
```

```
"ConnectionSummaryList": [
  {
    "ConnectionArn": "arn:aws:apprunner:us-east-1:123456789012:connection/
my-github-org-connection",
    "ConnectionName": "my-github-org-connection",
    "Status": "AVAILABLE",
    "CreatedAt": "2020-11-03T02:54:17Z",
    "ProviderType": "GITHUB"
  }
]
```

- For API details, see [ListConnections](#) in *AWS CLI Command Reference*.

list-operations

The following code example shows how to use `list-operations`.

AWS CLI

To list operations that occurred on a service

The following `list-operations` example lists all operations that occurred on an App Runner service so far. In this example, the service is new and only a single operation of type `CREATE_SERVICE` has occurred.

```
aws apprunner list-operations \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa"
}
```

Output:

```
{
  "OperationSummaryList": [
    {
      "EndedAt": 1606156217,
```

```

        "Id": "17fe9f55-7e91-4097-b243-fcabbb69a4cf",
        "StartedAt": 1606156014,
        "Status": "SUCCEEDED",
        "TargetArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
        "Type": "CREATE_SERVICE",
        "UpdatedAt": 1606156217
    }
]
}

```

- For API details, see [ListOperations](#) in *AWS CLI Command Reference*.

list-services

The following code example shows how to use `list-services`.

AWS CLI

To get a paginated listing of App Runner services

The following `list-services` example lists all App Runner services in the AWS account. Up to two services are listed in each response. This example shows the first request. The response includes two results and a token that can be used in the next request. When a subsequent response doesn't include a token, all services have been listed.

```
aws apprunner list-services \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "MaxResults": 2
}
```

Output:

```
{
  "NextToken":
  "eyJmZDdXN0b21lckFjY291bnRjZCI6IjI3MDEwNTQwMjg0NSIsI1NlcnZpY2VTdGF0dXNDb2R1IjojUFJpVmk1TSU90SU
  "ServiceSummaryList": [
```

```

    {
      "CreatedAt": "2020-11-20T19:05:25Z",
      "UpdatedAt": "2020-11-23T12:41:37Z",
      "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
      "ServiceId": "8fe1e10304f84fd2b0df550fe98a71fa",
      "ServiceName": "python-app",
      "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
      "Status": "RUNNING"
    },
    {
      "CreatedAt": "2020-11-06T23:15:30Z",
      "UpdatedAt": "2020-11-23T13:21:22Z",
      "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/golang-
container-app/ab8f94cfe29a460fb8760afd2ee87555",
      "ServiceId": "ab8f94cfe29a460fb8760afd2ee87555",
      "ServiceName": "golang-container-app",
      "ServiceUrl": "e2m8rrrx33.us-east-1.awsapprunner.com",
      "Status": "RUNNING"
    }
  ]
}

```

- For API details, see [ListServices](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags associated with an App Runner service

The following `list-tags-for-resource` example lists all the tags that are associated with an App Runner service.

```
aws apprunner list-tags-for-resource \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
```

```
"ResourceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa"
}
```

Output:

```
{
  "Tags": [
    {
      "Key": "Department",
      "Value": "Retail"
    },
    {
      "Key": "CustomerId",
      "Value": "56439872357912"
    }
  ]
}
```

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

pause-service

The following code example shows how to use `pause-service`.

AWS CLI

To pause a service

The following `pause-service` example pauses an App Runner service.

```
aws apprunner pause-service \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa"
}
```

Output:

```
{
  "OperationId": "17fe9f55-7e91-4097-b243-fcabbb69a4cf",
  "Service": {
    "CreatedAt": "2020-11-20T19:05:25Z",
    "UpdatedAt": "2020-11-23T12:41:37Z",
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
    "ServiceId": "8fe1e10304f84fd2b0df550fe98a71fa",
    "ServiceName": "python-app",
    "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
    "SourceConfiguration": {
      "AuthenticationConfiguration": {
        "ConnectionArn": "arn:aws:apprunner:us-
east-1:123456789012:connection/my-github-connection/
e7656250f67242d7819feade6800f59e"
      },
      "AutoDeploymentsEnabled": true,
      "CodeRepository": {
        "CodeConfiguration": {
          "CodeConfigurationValues": {
            "BuildCommand": "pip install -r requirements.txt",
            "Port": "8080",
            "Runtime": "PYTHON_3",
            "RuntimeEnvironmentVariables": [
              {
                "NAME": "Jane"
              }
            ],
            "StartCommand": "python server.py"
          },
          "ConfigurationSource": "Api"
        },
        "RepositoryUrl": "https://github.com/my-account/python-hello",
        "SourceCodeVersion": {
          "Type": "BRANCH",
          "Value": "main"
        }
      }
    },
    "Status": "OPERATION_IN_PROGRESS",
    "InstanceConfiguration": {
      "CPU": "1 vCPU",
```



```

        "Memory": "3 GB"
      }
    }
  }
}

```

- For API details, see [PauseService](#) in *AWS CLI Command Reference*.

resume-service

The following code example shows how to use `resume-service`.

AWS CLI

To resume a service

The following `resume-service` example resumes an App Runner service.

```
aws apprunner resume-service \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa"
}
```

Output:

```
{
  "OperationId": "17fe9f55-7e91-4097-b243-fcabbb69a4cf",
  "Service": {
    "CreatedAt": "2020-11-20T19:05:25Z",
    "UpdatedAt": "2020-11-23T12:41:37Z",
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
    "ServiceId": "8fe1e10304f84fd2b0df550fe98a71fa",
    "ServiceName": "python-app",
    "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
    "SourceConfiguration": {
      "AuthenticationConfiguration": {
```

```

        "ConnectionArn": "arn:aws:apprunner:us-
east-1:123456789012:connection/my-github-connection/
e7656250f67242d7819feade6800f59e"
    },
    "AutoDeploymentsEnabled": true,
    "CodeRepository": {
        "CodeConfiguration": {
            "CodeConfigurationValues": {
                "BuildCommand": "pip install -r requirements.txt",
                "Port": "8080",
                "Runtime": "PYTHON_3",
                "RuntimeEnvironmentVariables": [
                    {
                        "NAME": "Jane"
                    }
                ],
                "StartCommand": "python server.py"
            },
            "ConfigurationSource": "Api"
        },
        "RepositoryUrl": "https://github.com/my-account/python-hello",
        "SourceCodeVersion": {
            "Type": "BRANCH",
            "Value": "main"
        }
    }
},
"Status": "OPERATION_IN_PROGRESS",
"InstanceConfiguration": {
    "CPU": "1 vCPU",
    "Memory": "3 GB"
}
}
}

```

- For API details, see [ResumeService](#) in *AWS CLI Command Reference*.

start-deployment

The following code example shows how to use `start-deployment`.

AWS CLI

To initiate a manual deployment

The following `start-deployment` example performs a manual deployment to an App Runner service.

```
aws apprunner start-deployment \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-  
app/8fe1e10304f84fd2b0df550fe98a71fa"  
}
```

Output:

```
{  
  "OperationId": "853a7d5b-fc9f-4730-831b-fd8037ab832a"  
}
```

- For API details, see [StartDeployment](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add tags to an App Runner service

The following `tag-resource` example adds two tags to an App Runner service.

```
aws apprunner tag-resource \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "ResourceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
  "Tags": [
    {
      "Key": "Department",
      "Value": "Retail"
    },
    {
      "Key": "CustomerId",
      "Value": "56439872357912"
    }
  ]
}
```

This command produces no output.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from an App Runner service

The following `untag-resource` example removes two tags from an App Runner service.

```
aws apprunner untag-resource \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "ResourceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
  "TagKeys": [
    "Department",
    "CustomerId"
  ]
}
```

```
}
```

This command produces no output.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-service

The following code example shows how to use `update-service`.

AWS CLI

To update memory size

The following `update-service` example updates the memory size of instances (scaling units) of an App Runner service to 2048 MiB.

When the call succeeds, App Runner starts an asynchronous update process. The `Service` structure that's returned by the call reflects the new memory value that's being applied by this call.

```
aws apprunner update-service \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-  
app/8fe1e10304f84fd2b0df550fe98a71fa",  
  "InstanceConfiguration": {  
    "Memory": "4 GB"  
  }  
}
```

Output:

```
{  
  "OperationId": "17fe9f55-7e91-4097-b243-fcabbb69a4cf",  
  "Service": {  
    "CreatedAt": "2020-11-20T19:05:25Z",
```

```

    "UpdatedAt": "2020-11-23T12:41:37Z",
    "ServiceArn": "arn:aws:apprunner:us-east-1:123456789012:service/python-
app/8fe1e10304f84fd2b0df550fe98a71fa",
    "ServiceId": "8fe1e10304f84fd2b0df550fe98a71fa",
    "ServiceName": "python-app",
    "ServiceUrl": "psbqam834h.us-east-1.awsapprunner.com",
    "SourceConfiguration": {
      "AuthenticationConfiguration": {
        "ConnectionArn": "arn:aws:apprunner:us-
east-1:123456789012:connection/my-github-connection/
e7656250f67242d7819feade6800f59e"
      },
      "AutoDeploymentsEnabled": true,
      "CodeRepository": {
        "CodeConfiguration": {
          "CodeConfigurationValues": {
            "BuildCommand": "pip install -r requirements.txt",
            "Port": "8080",
            "Runtime": "PYTHON_3",
            "RuntimeEnvironmentVariables": [
              {
                "NAME": "Jane"
              }
            ],
            "StartCommand": "python server.py"
          },
          "ConfigurationSource": "Api"
        },
        "RepositoryUrl": "https://github.com/my-account/python-hello",
        "SourceCodeVersion": {
          "Type": "BRANCH",
          "Value": "main"
        }
      }
    },
    "Status": "OPERATION_IN_PROGRESS",
    "InstanceConfiguration": {
      "CPU": "1 vCPU",
      "Memory": "4 GB"
    }
  }
}

```

- For API details, see [UpdateService](#) in *AWS CLI Command Reference*.

AWS AppConfig examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS AppConfig.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-application

The following code example shows how to use create-application.

AWS CLI

To create an application

The following create-application example creates an application in AWS AppConfig.

```
aws appconfig create-application \  
  --name "example-application" \  
  --description "An application used for creating an example."
```

Output:

```
{  
  "Description": "An application used for creating an example.",  
  "Id": "339ohji",  
  "Name": "example-application"
```

```
}
```

For more information, see [Step 1: Creating an AWS AppConfig application](#) in the *AWS AppConfig User Guide*.

- For API details, see [CreateApplication](#) in *AWS CLI Command Reference*.

create-configuration-profile

The following code example shows how to use `create-configuration-profile`.

AWS CLI

To create a configuration profile

The following `create-configuration-profile` example creates a configuration profile using a configuration stored in Parameter Store, a capability of Systems Manager.

```
aws appconfig create-configuration-profile \  
  --application-id "339ohji" \  
  --name "Example-Configuration-Profile" \  
  --location-uri "ssm-parameter://Example-Parameter" \  
  --retrieval-role-arn "arn:aws:iam::111122223333:role/Example-App-Config-Role"
```

Output:

```
{  
  "ApplicationId": "339ohji",  
  "Description": null,  
  "Id": "ur8hx2f",  
  "LocationUri": "ssm-parameter://Example-Parameter",  
  "Name": "Example-Configuration-Profile",  
  "RetrievalRoleArn": "arn:aws:iam::111122223333:role/Example-App-Config-Role",  
  "Type": null,  
  "Validators": null  
}
```

For more information, see [Step 3: Creating a configuration and a configuration profile](#) in the *AWS AppConfig User Guide*.

- For API details, see [CreateConfigurationProfile](#) in *AWS CLI Command Reference*.

create-environment

The following code example shows how to use create-environment.

AWS CLI

To create an environment

The following create-environment example creates an AWS AppConfig environment named Example-Environment using the application you created using create-application.

```
aws appconfig create-environment \  
  --application-id "339ohji" \  
  --name "Example-Environment"
```

Output:

```
{  
  "ApplicationId": "339ohji",  
  "Description": null,  
  "Id": "54j1r29",  
  "Monitors": null,  
  "Name": "Example-Environment",  
  "State": "ReadyForDeployment"  
}
```

For more information, see [Step 2: Creating an environment](#) in the *AWS AppConfig User Guide*.

- For API details, see [CreateEnvironment](#) in *AWS CLI Command Reference*.

create-extension-association

The following code example shows how to use create-extension-association.

AWS CLI

To create an extension association

The following create-extension-association example creates a new extension association in AWS AppConfig.

```
aws appconfig create-extension-association \  
  --application-id "339ohji" \  
  --extension-id "54j1r29" \  
  --name "Example-Extension-Association"
```

```

--region us-west-2 \
--extension-identifier S3-backup-extension \
--resource-identifier "arn:aws:appconfig:us-west-2:123456789012:application/
Finance" \
--parameters S3bucket=FinanceConfigurationBackup

```

Output:

```

{
  "Id": "a1b2c3d4",
  "ExtensionArn": "arn:aws:appconfig:us-west-2:123456789012:extension/S3-backup-
extension/1",
  "ResourceArn": "arn:aws:appconfig:us-west-2:123456789012:application/Finance",
  "Parameters": {
    "S3bucket": "FinanceConfigurationBackup"
  },
  "ExtensionVersionNumber": 1
}

```

For more information, see [Working with AWS AppConfig extensions](#) in the *AWS AppConfig User Guide*.

- For API details, see [CreateExtensionAssociation](#) in *AWS CLI Command Reference*.

create-extension

The following code example shows how to use create-extension.

AWS CLI

To create an extension

The following create-extension example creates a new extension in AWS AppConfig.

```

aws appconfig create-extension \
  --region us-west-2 \
  --name S3-backup-extension \
  --actions
  PRE_CREATE_HOSTED_CONFIGURATION_VERSION=[{Name=S3backup,Uri=arn:aws:lambda:us-
west-2:123456789012:function:s3backupfunction,RoleArn=arn:aws:iam::123456789012:role/
appconfigextensionrole}] \
  --parameters S3bucket={Required=true}

```

Output:

```
{
  "Id": "1A2B3C4D",
  "Name": "S3-backup-extension",
  "VersionNumber": 1,
  "Arn": "arn:aws:appconfig:us-west-2:123456789012:extension/1A2B3C4D/1",
  "Actions": {
    "PRE_CREATE_HOSTED_CONFIGURATION_VERSION": [
      {
        "Name": "S3backup",
        "Uri": "arn:aws:lambda:us-
west-2:123456789012:function:s3backupfunction",
        "RoleArn": "arn:aws:iam::123456789012:role/appconfigextensionrole"
      }
    ]
  },
  "Parameters": {
    "S3bucket": {
      "Required": true
    }
  }
}
```

For more information, see [Working with AWS AppConfig extensions](#) in the *AWS AppConfig User Guide*.

- For API details, see [CreateExtension](#) in *AWS CLI Command Reference*.

create-hosted-configuration-version

The following code example shows how to use `create-hosted-configuration-version`.

AWS CLI**To create a hosted configuration version**

The following `create-hosted-configuration-version` example creates a new configuration in the AWS AppConfig hosted configuration store. The configuration content must first be converted to base64.

```
aws appconfig create-hosted-configuration-version \
```

```

--application-id "339ohji" \
--configuration-profile-id "ur8hx2f" \
--content
eyAiTmFtZSI6ICJFeGFtcGxlQXBwbGljYXRpb24iLCAiSWQiOiBFcGFtcGxlSUQsICJSYW5rIjogNyB9 \
--content-type "application/json" \
configuration_version_output_file

```

Contents of `configuration_version_output_file`:

```
{ "Name": "ExampleApplication", "Id": ExampleID, "Rank": 7 }
```

Output:

```

{
  "ApplicationId": "339ohji",
  "ConfigurationProfileId": "ur8hx2f",
  "VersionNumber": "1",
  "ContentType": "application/json"
}

```

For more information, see [About the AWS AppConfig hosted configuration store](#) in the *AWS AppConfig User Guide*.

- For API details, see [CreateHostedConfigurationVersion](#) in *AWS CLI Command Reference*.

delete-application

The following code example shows how to use `delete-application`.

AWS CLI

To delete an application

The following `delete-application` example deletes the specified application.

```

aws appconfig delete-application \
--application-id 339ohji

```

This command produces no output.

For more information, see [Step 1: Creating an AWS AppConfig application](#) in the *AWS AppConfig User Guide*.

- For API details, see [DeleteApplication](#) in *AWS CLI Command Reference*.

delete-configuration-profile

The following code example shows how to use delete-configuration-profile.

AWS CLI

To delete a configuration profile

The following delete-configuration-profile example deletes the specified configuration profile.

```
aws appconfig delete-configuration-profile \  
  --application-id 339ohji \  
  --configuration-profile-id ur8hx2f
```

This command produces no output.

For more information, see [Step 3: Creating a configuration and a configuration profile](#) in the *AWS AppConfig User Guide*.

- For API details, see [DeleteConfigurationProfile](#) in *AWS CLI Command Reference*.

delete-deployment-strategy

The following code example shows how to use delete-deployment-strategy.

AWS CLI

To delete a deployment strategy

The following delete-deployment-strategy example deletes the specified deployment strategy.

```
aws appconfig delete-deployment-strategy \  
  --deployment-strategy-id 1225qzk
```

This command produces no output.

For more information, see [Step 4: Creating a deployment strategy](#) in the *AWS AppConfig User Guide*.

- For API details, see [DeleteDeploymentStrategy](#) in *AWS CLI Command Reference*.

delete-environment

The following code example shows how to use `delete-environment`.

AWS CLI

To delete an environment

The following `delete-environment` example deletes the specified application environment.

```
aws appconfig delete-environment \  
  --application-id 339ohji \  
  --environment-id 54j1r29
```

This command produces no output.

For more information, see [Step 2: Creating an environment](#) in the *AWS AppConfig User Guide*.

- For API details, see [DeleteEnvironment](#) in *AWS CLI Command Reference*.

delete-extension-association

The following code example shows how to use `delete-extension-association`.

AWS CLI

To delete an extension association

The following `delete-extension-association` example deletes an extension association from AWS AppConfig.

```
aws appconfig delete-extension-association \  
  --region us-west-2 \  
  --extension-association-id a1b2c3d4
```

This command produces no output.

For more information, see [Working with AWS AppConfig extensions](#) in the *AWS AppConfig User Guide*.

- For API details, see [DeleteExtensionAssociation](#) in *AWS CLI Command Reference*.

delete-extension

The following code example shows how to use delete-extension.

AWS CLI

To delete an extension

The following delete-extension example deletes an extension from AWS AppConfig.

```
aws appconfig delete-extension \  
  --region us-west-2 \  
  --extension-identifier S3-backup-extension
```

This command produces no output.

For more information, see [Working with AWS AppConfig extensions](#) in the *AWS AppConfig User Guide*.

- For API details, see [DeleteExtension](#) in *AWS CLI Command Reference*.

delete-hosted-configuration-version

The following code example shows how to use delete-hosted-configuration-version.

AWS CLI

To delete a hosted configuration version

The following delete-hosted-configuration-version example deletes a configuration version hosted in the AWS AppConfig hosted configuration store.

```
aws appconfig delete-hosted-configuration-version \  
  --application-id 339ohji \  
  --configuration-profile-id ur8hx2f \  
  --version-number 1
```

Output:: This command produces no output.

For more information, see [Step 3: Creating a configuration and a configuration profile](#) in the *AWS AppConfig User Guide*.

- For API details, see [DeleteHostedConfigurationVersion](#) in *AWS CLI Command Reference*.

get-application

The following code example shows how to use `get-application`.

AWS CLI

To list details of an application

The following `get-application` example lists the details of the specified application.

```
aws appconfig get-application \  
  --application-id 339ohji
```

Output:

```
{  
  "Description": "An application used for creating an example.",  
  "Id": "339ohji",  
  "Name": "example-application"  
}
```

For more information, see [How AWS AppConfig works](#) in the *AWS AppConfig User Guide*.

- For API details, see [GetApplication](#) in *AWS CLI Command Reference*.

get-configuration-profile

The following code example shows how to use `get-configuration-profile`.

AWS CLI

To retrieve configuration profile details

The following `get-configuration-profile` example returns the details of the specified configuration profile.

```
aws appconfig get-configuration-profile \  
  --configuration-profile-id 339ohji
```



```
--application-id 339ohji \  
--configuration-profile-id ur8hx2f
```

Output:

```
{  
  "ApplicationId": "339ohji",  
  "Id": "ur8hx2f",  
  "Name": "Example-Configuration-Profile",  
  "LocationUri": "ssm-parameter://Example-Parameter",  
  "RetrievalRoleArn": "arn:aws:iam::111122223333:role/Example-App-Config-Role"  
}
```

For more information, see [Step 3: Creating a configuration and a configuration profile](#) in the *AWS AppConfig User Guide*.

- For API details, see [GetConfigurationProfile](#) in *AWS CLI Command Reference*.

get-configuration

The following code example shows how to use `get-configuration`.

AWS CLI

To retrieve configuration details

The following `get-configuration` example returns the configuration details of the example application. On subsequent calls to `get-configuration` use the `client-configuration-version` parameter to only update the configuration of your application if the version has changed. Only updating the configuration when the version has changed avoids excess charges incurred by calling `get-configuration`.

```
aws appconfig get-configuration \  
  --application "example-application" \  
  --environment "Example-Environment" \  
  --configuration "Example-Configuration-Profile" \  
  --client-id "test-id" \  
  configuration-output-file
```

Contents of configuration-output-file:

```
{ "Name": "ExampleApplication", "Id": ExampleID, "Rank": 7 }
```

Output:

```
{  
  "ConfigurationVersion": "1",  
  "ContentType": "application/json"  
}
```

For more information, see [Step 6: Receiving the configuration](#) in the *AWS AppConfig User Guide*.

- For API details, see [GetConfiguration](#) in *AWS CLI Command Reference*.

get-deployment-strategy

The following code example shows how to use `get-deployment-strategy`.

AWS CLI

To retrieve details of a deployment strategy

The following `get-deployment-strategy` example lists the details of the specified deployment strategy.

```
aws appconfig get-deployment-strategy \  
  --deployment-strategy-id 1225qzk
```

Output:

```
{  
  "Id": "1225qzk",  
  "Name": "Example-Deployment",  
  "DeploymentDurationInMinutes": 15,  
  "GrowthType": "LINEAR",  
  "GrowthFactor": 25.0,  
  "FinalBakeTimeInMinutes": 0,  
  "ReplicateTo": "SSM_DOCUMENT"  
}
```

For more information, see [Step 4: Creating a deployment strategy](#) in the *AWS AppConfig User Guide*.

- For API details, see [GetDeploymentStrategy](#) in *AWS CLI Command Reference*.

get-deployment

The following code example shows how to use get-deployment.

AWS CLI

To retrieve deployment details

The following get-deployment example lists details of the deployment to the application in the specified environment and deployment.

```
aws appconfig get-deployment \  
  --application-id 339ohji \  
  --environment-id 54j1r29 \  
  --deployment-number 1
```

Output:

```
{  
  "ApplicationId": "339ohji",  
  "EnvironmentId": "54j1r29",  
  "DeploymentStrategyId": "1225qzk",  
  "ConfigurationProfileId": "ur8hx2f",  
  "DeploymentNumber": 1,  
  "ConfigurationName": "Example-Configuration-Profile",  
  "ConfigurationLocationUri": "ssm-parameter://Example-Parameter",  
  "ConfigurationVersion": "1",  
  "DeploymentDurationInMinutes": 15,  
  "GrowthType": "LINEAR",  
  "GrowthFactor": 25.0,  
  "FinalBakeTimeInMinutes": 0,  
  "State": "COMPLETE",  
  "EventLog": [  
    {  
      "EventType": "DEPLOYMENT_COMPLETED",  
      "TriggeredBy": "APPCONFIG",  
      "Description": "Deployment completed",  
      "OccurredAt": "2021-09-17T21:59:03.888000+00:00"  
    },  
    {
```

```

    "EventType": "BAKE_TIME_STARTED",
    "TriggeredBy": "APPCONFIG",
    "Description": "Deployment bake time started",
    "OccurredAt": "2021-09-17T21:58:57.722000+00:00"
  },
  {
    "EventType": "PERCENTAGE_UPDATED",
    "TriggeredBy": "APPCONFIG",
    "Description": "Configuration available to 100.00% of clients",
    "OccurredAt": "2021-09-17T21:55:56.816000+00:00"
  },
  {
    "EventType": "PERCENTAGE_UPDATED",
    "TriggeredBy": "APPCONFIG",
    "Description": "Configuration available to 75.00% of clients",
    "OccurredAt": "2021-09-17T21:52:56.567000+00:00"
  },
  {
    "EventType": "PERCENTAGE_UPDATED",
    "TriggeredBy": "APPCONFIG",
    "Description": "Configuration available to 50.00% of clients",
    "OccurredAt": "2021-09-17T21:49:55.737000+00:00"
  },
  {
    "EventType": "PERCENTAGE_UPDATED",
    "TriggeredBy": "APPCONFIG",
    "Description": "Configuration available to 25.00% of clients",
    "OccurredAt": "2021-09-17T21:46:55.187000+00:00"
  },
  {
    "EventType": "DEPLOYMENT_STARTED",
    "TriggeredBy": "USER",
    "Description": "Deployment started",
    "OccurredAt": "2021-09-17T21:43:54.205000+00:00"
  }
],
"PercentageComplete": 100.0,
"StartedAt": "2021-09-17T21:43:54.205000+00:00",
"CompletedAt": "2021-09-17T21:59:03.888000+00:00"
}

```

For more information, see [Step 5: Deploying a configuration](#) in the *AWS AppConfig User Guide*.

- For API details, see [GetDeployment](#) in *AWS CLI Command Reference*.

get-environment

The following code example shows how to use `get-environment`.

AWS CLI

To retrieve environment details

The following `get-environment` example returns the details and state of the specified environment.

```
aws appconfig get-environment \  
  --application-id 339ohji \  
  --environment-id 54j1r29
```

Output:

```
{  
  "ApplicationId": "339ohji",  
  "Id": "54j1r29",  
  "Name": "Example-Environment",  
  "State": "ReadyForDeployment"  
}
```

For more information, see [Step 2: Creating an environment](#) in the *AWS AppConfig User Guide*.

- For API details, see [GetEnvironment](#) in *AWS CLI Command Reference*.

get-extension-association

The following code example shows how to use `get-extension-association`.

AWS CLI

To get extension association details

The following `get-extension-association` example displays information about an extension association.

```
aws appconfig get-extension-association \  
  --region us-west-2 \  
  --extension-id 54j1r29
```

```
--extension-association-id a1b2c3d4
```

Output:

```
{
  "Id": "a1b2c3d4",
  "ExtensionArn": "arn:aws:appconfig:us-west-2:123456789012:extension/S3-backup-
extension/1",
  "ResourceArn": "arn:aws:appconfig:us-west-2:123456789012:application/Finance",
  "Parameters": {
    "S3bucket": "FinanceConfigurationBackup"
  },
  "ExtensionVersionNumber": 1
}
```

For more information, see [Working with AWS AppConfig extensions](#) in the *AWS AppConfig User Guide*.

- For API details, see [GetExtensionAssociation](#) in *AWS CLI Command Reference*.

get-extension

The following code example shows how to use `get-extension`.

AWS CLI

To get extension details

The following `get-extension` example displays information about an extension.

```
aws appconfig get-extension \
  --region us-west-2 \
  --extension-identifier S3-backup-extension
```

Output:

```
{
  "Id": "1A2B3C4D",
  "Name": "S3-backup-extension",
  "VersionNumber": 1,
  "Arn": "arn:aws:appconfig:us-west-2:123456789012:extension/S3-backup-
extension/1",
```

```

    "Actions": {
      "PRE_CREATE_HOSTED_CONFIGURATION_VERSION": [
        {
          "Name": "S3backup",
          "Uri": "arn:aws:lambda:us-
west-2:123456789012:function:S3backupfunction",
          "RoleArn": "arn:aws:iam::123456789012:role/appconfigextensionrole"
        }
      ]
    },
    "Parameters": {
      "S3bucket": {
        "Required": true
      }
    }
  }
}

```

For more information, see [Working with AWS AppConfig extensions](#) in the *AWS AppConfig User Guide*.

- For API details, see [GetExtension](#) in *AWS CLI Command Reference*.

get-hosted-configuration-version

The following code example shows how to use `get-hosted-configuration-version`.

AWS CLI

To retrieve hosted configuration details

The following `get-hosted-configuration-version` example retrieves the configuration details of the AWS AppConfig hosted configuration.

```

aws appconfig get-hosted-configuration-version \
  --application-id 339ohji \
  --configuration-profile-id ur8hx2f \
  --version-number 1 \
  hosted-configuration-version-output

```

Contents of `hosted-configuration-version-output`:

```
{ "Name": "ExampleApplication", "Id": ExampleID, "Rank": 7 }
```

Output:

```
{
  "ApplicationId": "339ohji",
  "ConfigurationProfileId": "ur8hx2f",
  "VersionNumber": "1",
  "ContentType": "application/json"
}
```

For more information, see [About the AWS AppConfig hosted configuration store](#) in the *AWS AppConfig User Guide*.

- For API details, see [GetHostedConfigurationVersion](#) in *AWS CLI Command Reference*.

list-applications

The following code example shows how to use `list-applications`.

AWS CLI**To list the available applications**

The following `list-applications` example lists the available applications in your AWS account.

```
aws appconfig list-applications
```

Output:

```
{
  "Items": [
    {
      "Id": "339ohji",
      "Name": "test-application",
      "Description": "An application used for creating an example."
    },
    {
      "Id": "rwalwu7",
      "Name": "Test-Application"
    }
  ]
}
```



```
}
```

For more information, see [Step 1: Creating an AWS AppConfig application](#) in the *AWS AppConfig User Guide*.

- For API details, see [ListApplications](#) in *AWS CLI Command Reference*.

list-configuration-profiles

The following code example shows how to use `list-configuration-profiles`.

AWS CLI

To list the available configuration profiles

The following `list-configuration-profiles` example lists the available configuration profiles for the specified application.

```
aws appconfig list-configuration-profiles \
  --application-id 339ohji
```

Output:

```
{
  "Items": [
    {
      "ApplicationId": "339ohji",
      "Id": "ur8hx2f",
      "Name": "Example-Configuration-Profile",
      "LocationUri": "ssm-parameter://Example-Parameter"
    }
  ]
}
```

For more information, see [Step 3: Creating a configuration and a configuration profile](#) in the *AWS AppConfig User Guide*.

- For API details, see [ListConfigurationProfiles](#) in *AWS CLI Command Reference*.

list-deployment-strategies

The following code example shows how to use `list-deployment-strategies`.

AWS CLI

To list the available deployment strategies

The following `list-deployment-strategies` example lists the available deployment strategies in your AWS account.

```
aws AppConfig list-deployment-strategies
```

Output:

```
{
  "Items": [
    {
      "Id": "1225qzk",
      "Name": "Example-Deployment",
      "DeploymentDurationInMinutes": 15,
      "GrowthType": "LINEAR",
      "GrowthFactor": 25.0,
      "FinalBakeTimeInMinutes": 0,
      "ReplicateTo": "SSM_DOCUMENT"
    },
    {
      "Id": "AppConfig.AllAtOnce",
      "Name": "AppConfig.AllAtOnce",
      "Description": "Quick",
      "DeploymentDurationInMinutes": 0,
      "GrowthType": "LINEAR",
      "GrowthFactor": 100.0,
      "FinalBakeTimeInMinutes": 10,
      "ReplicateTo": "NONE"
    },
    {
      "Id": "AppConfig.Linear50PercentEvery30Seconds",
      "Name": "AppConfig.Linear50PercentEvery30Seconds",
      "Description": "Test/Demo",
      "DeploymentDurationInMinutes": 1,
      "GrowthType": "LINEAR",
      "GrowthFactor": 50.0,
      "FinalBakeTimeInMinutes": 1,
      "ReplicateTo": "NONE"
    },
    {
```

```

        "Id": "AppConfig.Canary10Percent20Minutes",
        "Name": "AppConfig.Canary10Percent20Minutes",
        "Description": "AWS Recommended",
        "DeploymentDurationInMinutes": 20,
        "GrowthType": "EXPONENTIAL",
        "GrowthFactor": 10.0,
        "FinalBakeTimeInMinutes": 10,
        "ReplicateTo": "NONE"
    }
]
}

```

For more information, see [Step 4: Creating a deployment strategy](#) in the *AWS AppConfig User Guide*.

- For API details, see [ListDeploymentStrategies](#) in *AWS CLI Command Reference*.

list-deployments

The following code example shows how to use `list-deployments`.

AWS CLI

To list the available deployments

The following `list-deployments` example lists the available deployments in your AWS account for the specified application and environment.

```

aws appconfig list-deployments \
  --application-id 339ohji \
  --environment-id 54j1r29

```

Output:

```

{
  "Items": [
    {
      "DeploymentNumber": 1,
      "ConfigurationName": "Example-Configuration-Profile",
      "ConfigurationVersion": "1",
      "DeploymentDurationInMinutes": 15,
      "GrowthType": "LINEAR",

```

```
        "GrowthFactor": 25.0,  
        "FinalBakeTimeInMinutes": 0,  
        "State": "COMPLETE",  
        "PercentageComplete": 100.0,  
        "StartedAt": "2021-09-17T21:43:54.205000+00:00",  
        "CompletedAt": "2021-09-17T21:59:03.888000+00:00"  
    }  
]  
}
```

For more information, see [Step 5: Deploying a configuration](#) in the *AWS AppConfig User Guide*.

- For API details, see [ListDeployments](#) in *AWS CLI Command Reference*.

list-environments

The following code example shows how to use `list-environments`.

AWS CLI

To list the available environments

The following `list-environments` example lists the available environments in your AWS account for the specified application.

```
aws appconfig list-environments \  
  --application-id 339ohji
```

Output:

```
{  
  "Items": [  
    {  
      "ApplicationId": "339ohji",  
      "Id": "54j1r29",  
      "Name": "Example-Environment",  
      "State": "ReadyForDeployment"  
    }  
  ]  
}
```

For more information, see [Step 2: Creating an environment](#) in the *AWS AppConfig User Guide*.

- For API details, see [ListEnvironments](#) in *AWS CLI Command Reference*.

list-extension-associations

The following code example shows how to use `list-extension-associations`.

AWS CLI

To list all AWS AppConfig extension associations in your AWS account for an AWS Region

The following `list-extension-associations` example lists all AWS AppConfig extension associations for the current AWS account in a specific AWS Region.

```
aws appconfig list-extension-associations \  
  --region us-west-2
```

Output:

```
{  
  "Items": [  
    {  
      "Id": "a1b2c3d4",  
      "ExtensionArn": "arn:aws:appconfig:us-west-2:123456789012:extension/S3-  
backup-extension/1",  
      "ResourceArn": "arn:aws:appconfig:us-west-2:123456789012:application/  
Finance"  
    }  
  ]  
}
```

For more information, see [Working with AWS AppConfig extensions](#) in the *AWS AppConfig User Guide*.

- For API details, see [ListExtensionAssociations](#) in *AWS CLI Command Reference*.

list-extensions

The following code example shows how to use `list-extensions`.

AWS CLI

To list all AWS AppConfig extensions in your AWS account for an AWS Region

The following `list-extensions` example lists all AWS AppConfig extensions for the current AWS account in a specific AWS Region. The command returns custom and AWS authored extensions.

```
aws appconfig list-extensions \  
  --region us-west-2
```

Output:

```
{  
  "Items": [  
    {  
      "Id": "1A2B3C4D",  
      "Name": "S3-backup-extension",  
      "VersionNumber": 1,  
      "Arn": "arn:aws:appconfig:us-west-2:123456789012:extension/1A2B3C4D/1"  
    },  
    {  
      "Id": "AWS.AppConfig.FeatureFlags",  
      "Name": "AppConfig Feature Flags Helper",  
      "VersionNumber": 1,  
      "Arn": "arn:aws:appconfig:us-west-2::extension/  
AWS.AppConfig.FeatureFlags/1",  
      "Description": "Validates AppConfig feature flag data automatically  
against a JSON schema that includes structure and constraints. Also transforms  
feature flag data prior to sending to the client. This extension is automatically  
associated to configuration profiles with type \"AWS.AppConfig.FeatureFlags\"."  
    },  
    {  
      "Id": "AWS.AppConfig.JiraIntegration",  
      "Name": "AppConfig integration with Atlassian Jira",  
      "VersionNumber": 1,  
      "Arn": "arn:aws:appconfig:us-west-2::extension/  
AWS.AppConfig.JiraIntegration/1",  
      "Description": "Exports feature flag data from AWS AppConfig into  
Jira. The lifecycle of each feature flag in AppConfig is tracked in Jira as an  
individual issue. Customers can see in Jira when flags are updated, turned on or  
off. Works in conjunction with the AppConfig app in the Atlassian Marketplace and  
is automatically associated to configuration profiles configured within that app."  
    },  
    {  
      "Id": "AWS.AppConfig.DeploymentNotificationsToEventBridge",  
      "Name": "AppConfig deployment events to Amazon EventBridge",  

```

```

        "VersionNumber": 1,
        "Arn": "arn:aws:appconfig:us-west-2::extension/
AWS.AppConfig.DeploymentNotificationsToEventBridge/1",
        "Description": "Sends events to Amazon EventBridge when a deployment
of configuration data in AppConfig is started, completed, or rolled back. Can
be associated to the following resources in AppConfig: Application, Environment,
Configuration Profile."
    },
    {
        "Id": "AWS.AppConfig.DeploymentNotificationsToSqs",
        "Name": "AppConfig deployment events to Amazon SQS",
        "VersionNumber": 1,
        "Arn": "arn:aws:appconfig:us-west-2::extension/
AWS.AppConfig.DeploymentNotificationsToSqs/1",
        "Description": "Sends messages to the configured Amazon SQS queue when
a deployment of configuration data in AppConfig is started, completed, or rolled
back. Can be associated to the following resources in AppConfig: Application,
Environment, Configuration Profile."
    },
    {
        "Id": "AWS.AppConfig.DeploymentNotificationsToSns",
        "Name": "AppConfig deployment events to Amazon SNS",
        "VersionNumber": 1,
        "Description": "Sends events to the configured Amazon SNS topic when
a deployment of configuration data in AppConfig is started, completed, or rolled
back. Can be associated to the following resources in AppConfig: Application,
Environment, Configuration Profile."
    }
]
}

```

For more information, see [Working with AWS AppConfig extensions](#) in the *AWS AppConfig User Guide*.

- For API details, see [ListExtensions](#) in *AWS CLI Command Reference*.

list-hosted-configuration-versions

The following code example shows how to use `list-hosted-configuration-versions`.

AWS CLI

To list the available hosted configuration versions

The following `list-hosted-configuration-versions` example lists the configurations versions hosted in the AWS AppConfig hosted configuration store for the specified application and configuration profile.

```
aws appconfig list-hosted-configuration-versions \  
  --application-id 339ohji \  
  --configuration-profile-id ur8hx2f
```

Output:

```
{  
  "Items": [  
    {  
      "ApplicationId": "339ohji",  
      "ConfigurationProfileId": "ur8hx2f",  
      "VersionNumber": 1,  
      "ContentType": "application/json"  
    }  
  ]  
}
```

For more information, see [About the AWS AppConfig hosted configuration store](#) in the *AWS AppConfig User Guide*.

- For API details, see [ListHostedConfigurationVersions](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags of an application

The following `list-tags-for-resource` example lists the tags of a specified application.

```
aws appconfig list-tags-for-resource \  
  --resource-arn arn:aws:appconfig:us-east-1:682428703967:application/339ohji
```

Output:


```
{
  "Tags": {
    "group1": "1"
  }
}
```

For more information, see [Step 1: Creating an AWS AppConfig application](#) in the *AWS AppConfig User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

start-deployment

The following code example shows how to use start-deployment.

AWS CLI

To start a configuration deployment

The following start-deployment example starts a deployment to the application using the specified environment, deployment strategy, and configuration profile.

```
aws appconfig start-deployment \
  --application-id 339ohji \
  --environment-id 54j1r29 \
  --deployment-strategy-id 1225qzk \
  --configuration-profile-id ur8hx2f \
  --configuration-version 1
```

Output:

```
{
  "ApplicationId": "339ohji",
  "EnvironmentId": "54j1r29",
  "DeploymentStrategyId": "1225qzk",
  "ConfigurationProfileId": "ur8hx2f",
  "DeploymentNumber": 1,
  "ConfigurationName": "Example-Configuration-Profile",
  "ConfigurationLocationUri": "ssm-parameter://Example-Parameter",
  "ConfigurationVersion": "1",
  "DeploymentDurationInMinutes": 15,
```

```

"GrowthType": "LINEAR",
"GrowthFactor": 25.0,
"FinalBakeTimeInMinutes": 0,
"State": "DEPLOYING",
"EventLog": [
  {
    "EventType": "DEPLOYMENT_STARTED",
    "TriggeredBy": "USER",
    "Description": "Deployment started",
    "OccurredAt": "2021-09-17T21:43:54.205000+00:00"
  }
],
"PercentageComplete": 0.0,
"StartedAt": "2021-09-17T21:43:54.205000+00:00"
}

```

For more information, see [Step 5: Deploying a configuration](#) in the *AWS AppConfig User Guide*.

- For API details, see [StartDeployment](#) in *AWS CLI Command Reference*.

stop-deployment

The following code example shows how to use stop-deployment.

AWS CLI

To stop configuration deployment

The following stop-deployment example stops the deployment of an application configuration to the specified environment.

```

aws appconfig stop-deployment \
  --application-id 339ohji \
  --environment-id 54j1r29 \
  --deployment-number 2

```

Output:

```

{
  "DeploymentNumber": 0,
  "DeploymentDurationInMinutes": 0,
  "GrowthFactor": 0.0,

```

```
"FinalBakeTimeInMinutes": 0,  
"PercentageComplete": 0.0  
}
```

For more information, see [Step 5: Deploying a configuration](#) in the *AWS AppConfig User Guide*.

- For API details, see [StopDeployment](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To tag an application

The following `tag-resource` example tags an application resource.

```
aws appconfig tag-resource \  
  --resource-arn arn:aws:appconfig:us-east-1:682428703967:application/339ohji \  
  --tags '{"group1" : "1"}
```

This command produces no output.

For more information, see [Step 1: Creating an AWS AppConfig application](#) in the *AWS AppConfig User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag from an application

The following `untag-resource` example removes the `group1` tag from the specified application.

```
aws appconfig untag-resource \  
  --resource-arn arn:aws:appconfig:us-east-1:682428703967:application/339ohji \  
  --tag-key group1
```

```
--resource-arn arn:aws:appconfig:us-east-1:111122223333:application/339ohji \  
--tag-keys '["group1"]'
```

This command produces no output.

For more information, see [Step 1: Creating an AWS AppConfig application](#) in the *AWS AppConfig User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-application

The following code example shows how to use `update-application`.

AWS CLI

To update an application

The following `update-application` example updates the name of the specified application.

```
aws appconfig update-application \  
--application-id 339ohji \  
--name "Example-Application"
```

Output:

```
{  
  "Id": "339ohji",  
  "Name": "Example-Application",  
  "Description": "An application used for creating an example."  
}
```

For more information, see [Step 1: Creating an AWS AppConfig application](#) in the *AWS AppConfig User Guide*.

- For API details, see [UpdateApplication](#) in *AWS CLI Command Reference*.

update-configuration-profile

The following code example shows how to use `update-configuration-profile`.

AWS CLI

To update a configuration profile

The following `update-configuration-profile` example updates the description of the specified configuration profile.

```
aws appconfig update-configuration-profile \  
  --application-id 339ohji \  
  --configuration-profile-id ur8hx2f \  
  --description "Configuration profile used for examples."
```

Output:

```
{  
  "ApplicationId": "339ohji",  
  "Id": "ur8hx2f",  
  "Name": "Example-Configuration-Profile",  
  "Description": "Configuration profile used for examples.",  
  "LocationUri": "ssm-parameter://Example-Parameter",  
  "RetrievalRoleArn": "arn:aws:iam::111122223333:role/Example-App-Config-Role"  
}
```

For more information, see [Step 3: Creating a configuration and a configuration profile](#) in the *AWS AppConfig User Guide*.

- For API details, see [UpdateConfigurationProfile](#) in *AWS CLI Command Reference*.

update-deployment-strategy

The following code example shows how to use `update-deployment-strategy`.

AWS CLI

To update a deployment strategy

The following `update-deployment-strategy` example updates final bake time to 20 minutes in the specified deployment strategy.

```
aws appconfig update-deployment-strategy \  
  --application-id 339ohji \  
  --deployment-strategy-id ur8hx2f \  
  --final-bake-time 20
```

```
--deployment-strategy-id 1225qzk \  
--final-bake-time-in-minutes 20
```

Output:

```
{  
  "Id": "1225qzk",  
  "Name": "Example-Deployment",  
  "DeploymentDurationInMinutes": 15,  
  "GrowthType": "LINEAR",  
  "GrowthFactor": 25.0,  
  "FinalBakeTimeInMinutes": 20,  
  "ReplicateTo": "SSM_DOCUMENT"  
}
```

For more information, see [Step 4: Creating a deployment strategy](#) in the *AWS AppConfig User Guide*.

- For API details, see [UpdateDeploymentStrategy](#) in *AWS CLI Command Reference*.

update-environment

The following code example shows how to use `update-environment`.

AWS CLI

To update an environment

The following `update-environment` example updates an environment's description.

```
aws appconfig update-environment \  
  --application-id 339ohji \  
  --environment-id 54j1r29 \  
  --description "An environment for examples."
```

Output:

```
{  
  "ApplicationId": "339ohji",  
  "Id": "54j1r29",  
  "Name": "Example-Environment",
```

```
"Description": "An environment for examples.",
"State": "RolledBack"
}
```

For more information, see [Step 2: Creating an environment](#) in the *AWS AppConfig User Guide*.

- For API details, see [UpdateEnvironment](#) in *AWS CLI Command Reference*.

update-extension-association

The following code example shows how to use `update-extension-association`.

AWS CLI

To update an AWS AppConfig extension association

The following `update-extension-association` example adds a new parameter value to an extension association in AWS AppConfig.

```
aws appconfig update-extension-association \
  --region us-west-2 \
  --extension-association-id a1b2c3d4 \
  --parameters S3bucket=FinanceMobileApp
```

Output:

```
{
  "Id": "a1b2c3d4",
  "ExtensionArn": "arn:aws:appconfig:us-west-2:123456789012:extension/S3-backup-
extension/1",
  "ResourceArn": "arn:aws:appconfig:us-west-2:123456789012:application/Finance",
  "Parameters": {
    "S3bucket": "FinanceMobileApp"
  },
  "ExtensionVersionNumber": 1
}
```

For more information, see [Working with AWS AppConfig extensions](#) in the *AWS AppConfig User Guide*.

- For API details, see [UpdateExtensionAssociation](#) in *AWS CLI Command Reference*.

update-extension

The following code example shows how to use update-extension.

AWS CLI

To update an AWS AppConfig extension

The following update-extension example adds an additional parameter Key to an extension in AWS AppConfig.

```
aws appconfig update-extension \  
  --region us-west-2 \  
  --extension-identifier S3-backup-extension \  
  --parameters S3bucket={Required=true},CampaignID={Required=false}
```

Output:

```
{  
  "Id": "1A2B3C4D",  
  "Name": "S3-backup-extension",  
  "VersionNumber": 1,  
  "Arn": "arn:aws:appconfig:us-west-2:123456789012:extension/1A2B3C4D/1",  
  "Actions": {  
    "PRE_CREATE_HOSTED_CONFIGURATION_VERSION": [  
      {  
        "Name": "S3backup",  
        "Uri": "arn:aws:lambda:us-  
west-2:123456789012:function:S3backupfunction",  
        "RoleArn": "arn:aws:iam::123456789012:role/appconfigextensionrole"  
      }  
    ]  
  },  
  "Parameters": {  
    "CampaignID": {  
      "Required": false  
    },  
    "S3bucket": {  
      "Required": true  
    }  
  }  
}
```


For more information, see [Working with AWS AppConfig extensions](#) in the *AWS AppConfig User Guide*.

- For API details, see [UpdateExtension](#) in *AWS CLI Command Reference*.

validate-configuration

The following code example shows how to use `validate-configuration`.

AWS CLI

To validate a configuration

The following `validate-configuration` example uses the validators in a configuration profile to validate a configuration.

```
aws appconfig validate-configuration \  
  --application-id abc1234 \  
  --configuration-profile-id ur8hx2f \  
  --configuration-version 1
```

The command produces no output.

For more information, see [Step 3: Creating a configuration and a configuration profile](#) in the *AWS AppConfig User Guide*.

- For API details, see [ValidateConfiguration](#) in *AWS CLI Command Reference*.

Application Auto Scaling examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Application Auto Scaling.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

delete-scaling-policy

The following code example shows how to use `delete-scaling-policy`.

AWS CLI

To delete a scaling policy

This example deletes a scaling policy for the Amazon ECS service `web-app` running in the default cluster.

Command:

```
aws application-autoscaling delete-scaling-policy --policy-name web-app-cpu-1t-25 --scalable-dimension ecs:service:DesiredCount --resource-id service/default/web-app --service-namespace ecs
```

- For API details, see [DeleteScalingPolicy](#) in *AWS CLI Command Reference*.

delete-scheduled-action

The following code example shows how to use `delete-scheduled-action`.

AWS CLI

To delete a scheduled action

The following `delete-scheduled-action` example deletes the specified scheduled action from the specified Amazon AppStream 2.0 fleet:

```
aws application-autoscaling delete-scheduled-action \
  --service-namespace appstream \
  --scalable-dimension appstream:fleet:DesiredCapacity \
  --resource-id fleet/sample-fleet \
```

```
--scheduled-action-name my-recurring-action
```

This command produces no output.

For more information, see [Scheduled Scaling](#) in the *Application Auto Scaling User Guide*.

- For API details, see [DeleteScheduledAction](#) in *AWS CLI Command Reference*.

deregister-scalable-target

The following code example shows how to use `deregister-scalable-target`.

AWS CLI

To deregister a scalable target

This example deregisters a scalable target for an Amazon ECS service called `web-app` that is running in the default cluster.

Command:

```
aws application-autoscaling deregister-scalable-target --service-namespace ecs --scalable-dimension ecs:service:DesiredCount --resource-id service/default/web-app
```

This example deregisters a scalable target for a custom resource. The `custom-resource-id.txt` file contains a string that identifies the Resource ID, which, for a custom resource, is the path to the custom resource through your Amazon API Gateway endpoint.

Command:

```
aws application-autoscaling deregister-scalable-target --service-namespace custom-resource --scalable-dimension custom-resource:ResourceType:Property --resource-id file://~/custom-resource-id.txt
```

Contents of `custom-resource-id.txt` file:

```
https://example.execute-api.us-west-2.amazonaws.com/prod/scalableTargetDimensions/1-23456789
```

- For API details, see [DeregisterScalableTarget](#) in *AWS CLI Command Reference*.

describe-scalable-targets

The following code example shows how to use `describe-scalable-targets`.

AWS CLI

To describe scalable targets

The following `describe-scalable-targets` example describes the scalable targets for the `ecs` service namespace.

```
aws application-autoscaling describe-scalable-targets \  
  --service-namespace ecs
```

Output:

```
{  
  "ScalableTargets": [  
    {  
      "ServiceNamespace": "ecs",  
      "ScalableDimension": "ecs:service:DesiredCount",  
      "ResourceId": "service/default/web-app",  
      "MinCapacity": 1,  
      "MaxCapacity": 10,  
      "RoleARN": "arn:aws:iam::123456789012:role/  
aws-service-role/ecs.application-autoscaling.amazonaws.com/  
AWSServiceRoleForApplicationAutoScaling_ECSService",  
      "CreationTime": 1462558906.199,  
      "SuspendedState": {  
        "DynamicScalingOutSuspended": false,  
        "ScheduledScalingSuspended": false,  
        "DynamicScalingInSuspended": false  
      },  
      "ScalableTargetARN": "arn:aws:application-autoscaling:us-  
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"  
    }  
  ]  
}
```

For more information, see [AWS services that you can use with Application Auto Scaling](#) in the *Application Auto Scaling User Guide*.

- For API details, see [DescribeScalableTargets](#) in *AWS CLI Command Reference*.

describe-scaling-activities

The following code example shows how to use `describe-scaling-activities`.

AWS CLI

Example 1: To describe scaling activities for the specified Amazon ECS service

The following `describe-scaling-activities` example describes the scaling activities for an Amazon ECS service called `web-app` that is running in the default cluster. The output shows a scaling activity initiated by a scaling policy.

```
aws application-autoscaling describe-scaling-activities \
  --service-namespace ecs \
  --resource-id service/default/web-app
```

Output:

```
{
  "ScalingActivities": [
    {
      "ScalableDimension": "ecs:service:DesiredCount",
      "Description": "Setting desired count to 1.",
      "ResourceId": "service/default/web-app",
      "ActivityId": "e6c5f7d1-dbbb-4a3f-89b2-51f33e766399",
      "StartTime": 1462575838.171,
      "ServiceNamespace": "ecs",
      "EndTime": 1462575872.111,
      "Cause": "monitor alarm web-app-cpu-lt-25 in state ALARM triggered
policy web-app-cpu-lt-25",
      "StatusMessage": "Successfully set desired count to 1. Change
successfully fulfilled by ecs.",
      "StatusCode": "Successful"
    }
  ]
}
```

For more information, see [Scaling activities for Application Auto Scaling](#) in the *Application Auto Scaling User Guide*.

Example 2: To describe scaling activities for the specified DynamoDB table

The following `describe-scaling-activities` example describes the scaling activities for a DynamoDB table called `TestTable`. The output shows scaling activities initiated by two different scheduled actions.

```
aws application-autoscaling describe-scaling-activities \  
  --service-namespace dynamodb \  
  --resource-id table/TestTable
```

Output:

```
{  
  "ScalingActivities": [  
    {  
      "ScalableDimension": "dynamodb:table:WriteCapacityUnits",  
      "Description": "Setting write capacity units to 10.",  
      "ResourceId": "table/my-table",  
      "ActivityId": "4d1308c0-bbcf-4514-a673-b0220ae38547",  
      "StartTime": 1561574415.086,  
      "ServiceNamespace": "dynamodb",  
      "EndTime": 1561574449.51,  
      "Cause": "maximum capacity was set to 10",  
      "StatusMessage": "Successfully set write capacity units to 10. Change  
successfully fulfilled by dynamodb.",  
      "StatusCode": "Successful"  
    },  
    {  
      "ScalableDimension": "dynamodb:table:WriteCapacityUnits",  
      "Description": "Setting min capacity to 5 and max capacity to 10",  
      "ResourceId": "table/my-table",  
      "ActivityId": "f2b7847b-721d-4e01-8ef0-0c8d3bacc1c7",  
      "StartTime": 1561574414.644,  
      "ServiceNamespace": "dynamodb",  
      "Cause": "scheduled action name my-second-scheduled-action was  
triggered",  
      "StatusMessage": "Successfully set min capacity to 5 and max capacity to  
10",  
      "StatusCode": "Successful"  
    },  
    {  
      "ScalableDimension": "dynamodb:table:WriteCapacityUnits",  
      "Description": "Setting write capacity units to 15.",  
      "ResourceId": "table/my-table",  
      "ActivityId": "d8ea4de6-9eaa-499f-b466-2cc5e681ba8b",
```

```

        "StartTime": 1561574108.904,
        "ServiceNamespace": "dynamodb",
        "EndTime": 1561574140.255,
        "Cause": "minimum capacity was set to 15",
        "StatusMessage": "Successfully set write capacity units to 15. Change
successfully fulfilled by dynamodb.",
        "StatusCode": "Successful"
    },
    {
        "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
        "Description": "Setting min capacity to 15 and max capacity to 20",
        "ResourceId": "table/my-table",
        "ActivityId": "3250fd06-6940-4e8e-bb1f-d494db7554d2",
        "StartTime": 1561574108.512,
        "ServiceNamespace": "dynamodb",
        "Cause": "scheduled action name my-first-scheduled-action was
triggered",
        "StatusMessage": "Successfully set min capacity to 15 and max capacity
to 20",
        "StatusCode": "Successful"
    }
]
}

```

For more information, see [Scaling activities for Application Auto Scaling](#) in the *Application Auto Scaling User Guide*.

- For API details, see [DescribeScalingActivities](#) in *AWS CLI Command Reference*.

describe-scaling-policies

The following code example shows how to use `describe-scaling-policies`.

AWS CLI

To describe scaling policies

This example command describes the scaling policies for the `ecs` service namespace.

Command:

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs
```

Output:

```
{
  "ScalingPolicies": [
    {
      "PolicyName": "web-app-cpu-gt-75",
      "ScalableDimension": "ecs:service:DesiredCount",
      "ResourceId": "service/default/web-app",
      "CreationTime": 1462561899.23,
      "StepScalingPolicyConfiguration": {
        "Cooldown": 60,
        "StepAdjustments": [
          {
            "ScalingAdjustment": 200,
            "MetricIntervalLowerBound": 0.0
          }
        ],
        "AdjustmentType": "PercentChangeInCapacity"
      },
      "PolicyARN": "arn:aws:autoscaling:us-
west-2:012345678910:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/ecs/
service/default/web-app:policyName/web-app-cpu-gt-75",
      "PolicyType": "StepScaling",
      "Alarms": [
        {
          "AlarmName": "web-app-cpu-gt-75",
          "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:web-app-cpu-gt-75"
        }
      ],
      "ServiceNamespace": "ecs"
    },
    {
      "PolicyName": "web-app-cpu-lt-25",
      "ScalableDimension": "ecs:service:DesiredCount",
      "ResourceId": "service/default/web-app",
      "CreationTime": 1462562575.099,
      "StepScalingPolicyConfiguration": {
        "Cooldown": 1,
        "StepAdjustments": [
          {
            "ScalingAdjustment": -50,
            "MetricIntervalUpperBound": 0.0
          }
        ]
      }
    }
  ]
}
```



```

        ],
        "AdjustmentType": "PercentChangeInCapacity"
    },
    "PolicyARN": "arn:aws:autoscaling:us-
west-2:012345678910:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/ecs/
service/default/web-app:policyName/web-app-cpu-1t-25",
    "PolicyType": "StepScaling",
    "Alarms": [
        {
            "AlarmName": "web-app-cpu-1t-25",
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:web-app-cpu-1t-25"
        }
    ],
    "ServiceNamespace": "ecs"
}
]
}

```

- For API details, see [DescribeScalingPolicies](#) in *AWS CLI Command Reference*.

describe-scheduled-actions

The following code example shows how to use `describe-scheduled-actions`.

AWS CLI

To describe scheduled actions

The following `describe-scheduled-actions` example displays details for the scheduled actions for the specified service namespace:

```
aws application-autoscaling describe-scheduled-actions \
  --service-namespace dynamodb
```

Output:

```
{
  "ScheduledActions": [
    {
      "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
```

```

    "Schedule": "at(2019-05-20T18:35:00)",
    "ResourceId": "table/my-table",
    "CreationTime": 1561571888.361,
    "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:2d36aa3b-cdf9-4565-b290-81db519b227d:resource/
dynamodb/table/my-table:scheduledActionName/my-first-scheduled-action",
    "ScalableTargetAction": {
        "MinCapacity": 15,
        "MaxCapacity": 20
    },
    "ScheduledActionName": "my-first-scheduled-action",
    "ServiceNamespace": "dynamodb"
},
{
    "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
    "Schedule": "at(2019-05-20T18:40:00)",
    "ResourceId": "table/my-table",
    "CreationTime": 1561571946.021,
    "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:2d36aa3b-cdf9-4565-b290-81db519b227d:resource/
dynamodb/table/my-table:scheduledActionName/my-second-scheduled-action",
    "ScalableTargetAction": {
        "MinCapacity": 5,
        "MaxCapacity": 10
    },
    "ScheduledActionName": "my-second-scheduled-action",
    "ServiceNamespace": "dynamodb"
}
]
}

```

For more information, see [Scheduled Scaling](#) in the *Application Auto Scaling User Guide*.

- For API details, see [DescribeScheduledActions](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags for a scalable target

The following `list-tags-for-resource` example lists the tag key names and values that are attached to the scalable target specified by its ARN.

```
aws application-autoscaling list-tags-for-resource \
  --resource-arn arn:aws:application-autoscaling:us-west-2:123456789012:scalable-
  target/1234abcd56ab78cd901ef1234567890ab123
```

Output:

```
{
  "Tags": {
    "environment": "production"
  }
}
```

For more information, see [Tagging support for Application Auto Scaling](#) in the *Application Auto Scaling User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

put-scaling-policy

The following code example shows how to use `put-scaling-policy`.

AWS CLI

Example 1: To apply a target tracking scaling policy with a predefined metric specification

The following `put-scaling-policy` example applies a target tracking scaling policy with a predefined metric specification to an Amazon ECS service called `web-app` in the default cluster. The policy keeps the average CPU utilization of the service at 75 percent, with scale-out and scale-in cooldown periods of 60 seconds. The output contains the ARNs and names of the two CloudWatch alarms created on your behalf.

```
aws application-autoscaling put-scaling-policy --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/default/web-app \
  --policy-name cpu75-target-tracking-scaling-policy --policy-type
  TargetTrackingScaling \
  --target-tracking-scaling-policy-configuration file://config.json
```

This example assumes that you have a `config.json` file in the current directory with the following contents:

```
{
  "TargetValue": 75.0,
  "PredefinedMetricSpecification": {
    "PredefinedMetricType": "ECSServiceAverageCPUUtilization"
  },
  "ScaleOutCooldown": 60,
  "ScaleInCooldown": 60
}
```

Output:

```
{
  "PolicyARN": "arn:aws:autoscaling:us-west-2:012345678910:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/ecs/service/default/web-app:policyName/cpu75-target-tracking-scaling-policy",
  "Alarms": [
    {
      "AlarmARN": "arn:aws:cloudwatch:us-west-2:012345678910:alarm:TargetTracking-service/default/web-app-AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca",
      "AlarmName": "TargetTracking-service/default/web-app-AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca"
    },
    {
      "AlarmARN": "arn:aws:cloudwatch:us-west-2:012345678910:alarm:TargetTracking-service/default/web-app-AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d",
      "AlarmName": "TargetTracking-service/default/web-app-AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d"
    }
  ]
}
```

Example 2: To apply a target tracking scaling policy with a customized metric specification

The following `put-scaling-policy` example applies a target tracking scaling policy with a customized metric specification to an Amazon ECS service called `web-app` in the default cluster. The policy keeps the average utilization of the service at 75 percent, with scale-out and

scale-in cooldown periods of 60 seconds. The output contains the ARNs and names of the two CloudWatch alarms created on your behalf.

```
aws application-autoscaling put-scaling-policy --service-namespace ecs \  
--scalable-dimension ecs:service:DesiredCount \  
--resource-id service/default/web-app \  
--policy-name cms75-target-tracking-scaling-policy \  
--policy-type TargetTrackingScaling \  
--target-tracking-scaling-policy-configuration file://config.json
```

This example assumes that you have a config.json file in the current directory with the following contents:

```
{  
  "TargetValue":75.0,  
  "CustomizedMetricSpecification":{  
    "MetricName":"MyUtilizationMetric",  
    "Namespace":"MyNamespace",  
    "Dimensions": [  
      {  
        "Name":"MyOptionalMetricDimensionName",  
        "Value":"MyOptionalMetricDimensionValue"  
      }  
    ],  
    "Statistic":"Average",  
    "Unit":"Percent"  
  },  
  "ScaleOutCooldown": 60,  
  "ScaleInCooldown": 60  
}
```

Output:

```
{  
  "PolicyARN": "arn:aws:autoscaling:us-west-2:012345678910:scalingPolicy:  
8784a896-b2ba-47a1-b08c-27301cc499a1:resource/ecs/service/default/web-  
app:policyName/cms75-target-tracking-scaling-policy",  
  "Alarms": [  
    {  
      "AlarmARN": "arn:aws:cloudwatch:us-  
west-2:012345678910:alarm:TargetTracking-service/default/web-app-  
AlarmHigh-9bc77b56-0571-4276-ba0f-d4178882e0a0",
```

```

        "AlarmName": "TargetTracking-service/default/web-app-
AlarmHigh-9bc77b56-0571-4276-ba0f-d4178882e0a0"
    },
    {
        "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/default/web-app-
AlarmLow-9b6ad934-6d37-438e-9e05-02836ddcbdc4",
        "AlarmName": "TargetTracking-service/default/web-app-
AlarmLow-9b6ad934-6d37-438e-9e05-02836ddcbdc4"
    }
]
}

```

Example 3: To apply a target tracking scaling policy for scale out only

The following `put-scaling-policy` example applies a target tracking scaling policy to an Amazon ECS service called `web-app` in the default cluster. The policy is used to scale out the ECS service when the `RequestCountPerTarget` metric from the Application Load Balancer exceeds the threshold. The output contains the ARN and name of the CloudWatch alarm created on your behalf.

```

aws application-autoscaling put-scaling-policy \
  --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/default/web-app \
  --policy-name alb-scale-out-target-tracking-scaling-policy \
  --policy-type TargetTrackingScaling \
  --target-tracking-scaling-policy-configuration file://config.json

```

Contents of `config.json`:

```

{
  "TargetValue": 1000.0,
  "PredefinedMetricSpecification": {
    "PredefinedMetricType": "ALBRequestCountPerTarget",
    "ResourceLabel": "app/EC2Co-EcsE1-1TKLTMITMM0E0/f37c06a68c1748aa/
targetgroup/EC2Co-Defau-LDNM7Q3ZH1ZN/6d4ea56ca2d6a18d"
  },
  "ScaleOutCooldown": 60,
  "ScaleInCooldown": 60,
  "DisableScaleIn": true
}

```

Output:

```
{
  "PolicyARN": "arn:aws:autoscaling:us-west-2:123456789012:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/ecs/service/default/web-app:policyName/alb-scale-out-target-tracking-scaling-policy",
  "Alarms": [
    {
      "AlarmName": "TargetTracking-service/default/web-app-AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca",
      "AlarmARN": "arn:aws:cloudwatch:us-west-2:123456789012:alarm:TargetTracking-service/default/web-app-AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca"
    }
  ]
}
```

For more information, see [Target Tracking Scaling Policies for Application Auto Scaling](#) in the *AWS Application Auto Scaling User Guide*.

- For API details, see [PutScalingPolicy](#) in *AWS CLI Command Reference*.

put-scheduled-action

The following code example shows how to use `put-scheduled-action`.

AWS CLI**To add a scheduled action to a DynamoDB table**

This example adds a scheduled action to a DynamoDB table called `TestTable` to scale out on a recurring schedule. On the specified schedule (every day at 12:15pm UTC), if the current capacity is below the value specified for `MinCapacity`, Application Auto Scaling scales out to the value specified by `MinCapacity`.

Command:

```
aws application-autoscaling put-scheduled-action --service-namespace dynamodb
--scheduled-action-name my-recurring-action --schedule "cron(15 12 * * ? *)" --
resource-id table/TestTable --scalable-dimension dynamodb:table:WriteCapacityUnits
--scalable-target-action MinCapacity=6
```

For more information, see Scheduled Scaling in the *Application Auto Scaling User Guide*.

- For API details, see [PutScheduledAction](#) in *AWS CLI Command Reference*.

register-scalable-target

The following code example shows how to use `register-scalable-target`.

AWS CLI

Example 1: To register an ECS service as a scalable target

The following `register-scalable-target` example registers an Amazon ECS service with Application Auto Scaling. It also adds a tag with the key name `environment` and the value `production` to the scalable target.

```
aws application-autoscaling register-scalable-target \  
  --service-namespace ecs \  
  --scalable-dimension ecs:service:DesiredCount \  
  --resource-id service/default/web-app \  
  --min-capacity 1 --max-capacity 10 \  
  --tags environment=production
```

Output:

```
{  
  "ScalableTargetARN": "arn:aws:application-autoscaling:us-  
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"  
}
```

For examples for other AWS services and custom resources, see the topics in [AWS services that you can use with Application Auto Scaling](#) in the *Application Auto Scaling User Guide*.

Example 2: To suspend scaling activities for a scalable target

The following `register-scalable-target` example suspends scaling activities for an existing scalable target.

```
aws application-autoscaling register-scalable-target \  
  --service-namespace dynamodb \  
  --scalable-dimension dynamodb:table:ReadCapacityUnits \  
  --resource-id dynamodb:table:my-table
```



```
--resource-id table/my-table \  
--suspended-state  
DynamicScalingInSuspended=true,DynamicScalingOutSuspended=true,ScheduledScalingSuspended=tr
```

Output:

```
{  
  "ScalableTargetARN": "arn:aws:application-autoscaling:us-  
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"  
}
```

For more information, see [Suspending and resuming scaling for Application Auto Scaling](#) in the *Application Auto Scaling User Guide*.

Example 3: To resume scaling activities for a scalable target

The following `register-scalable-target` example resumes scaling activities for an existing scalable target.

```
aws application-autoscaling register-scalable-target \  
--service-namespace dynamodb \  
--scalable-dimension dynamodb:table:ReadCapacityUnits \  
--resource-id table/my-table \  
--suspended-state  
DynamicScalingInSuspended=false,DynamicScalingOutSuspended=false,ScheduledScalingSuspended=
```

Output:

```
{  
  "ScalableTargetARN": "arn:aws:application-autoscaling:us-  
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"  
}
```

For more information, see [Suspending and resuming scaling for Application Auto Scaling](#) in the *Application Auto Scaling User Guide*.

- For API details, see [RegisterScalableTarget](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add a tag to a scalable target

The following `tag-resource` example adds a tag with the key name `environment` and the value `production` to the scalable target specified by its ARN.

```
aws application-autoscaling tag-resource \  
  --resource-arn arn:aws:application-autoscaling:us-west-2:123456789012:scalable-  
target/1234abcd56ab78cd901ef1234567890ab123 \  
  --tags environment=production
```

This command produces no output.

For more information, see [Tagging support for Application Auto Scaling](#) in the *Application Auto Scaling User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag from a scalable target

The following `untag-resource` example removes the tag pair with the key name `environment` from the scalable target specified by its ARN.

```
aws application-autoscaling untag-resource \  
  --resource-arn arn:aws:application-autoscaling:us-west-2:123456789012:scalable-  
target/1234abcd56ab78cd901ef1234567890ab123 \  
  --tag-keys "environment"
```

This command produces no output.

For more information, see [Tagging support for Application Auto Scaling](#) in the *Application Auto Scaling User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

Application Discovery Service examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Application Discovery Service.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

describe-agents

The following code example shows how to use `describe-agents`.

AWS CLI

Describe agents with specified `collectionStatus` states

This example command describes collection agents with collection status of "STARTED" or "STOPPED".

Command:

```
aws discovery describe-agents --filters
name="collectionStatus",values="STARTED","STOPPED",condition="EQUALS" --max-results
3
```

Output:

```
{
  "Snapshots": [
    {
```

```
    "version": "1.0.40.0",
    "agentType": "EC2",
    "hostName": "ip-172-31-40-234",
    "collectionStatus": "STOPPED",
    "agentNetworkInfoList": [
      {
        "macAddress": "06:b5:97:14:fc:0d",
        "ipAddress": "172.31.40.234"
      }
    ],
    "health": "UNKNOWN",
    "agentId": "i-003305c02a776e883",
    "registeredTime": "2016-12-09T19:05:06Z",
    "lastHealthPingTime": "2016-12-09T19:05:10Z"
  },
  {
    "version": "1.0.40.0",
    "agentType": "EC2",
    "hostName": "ip-172-31-39-64",
    "collectionStatus": "STARTED",
    "agentNetworkInfoList": [
      {
        "macAddress": "06:a1:0e:c7:b2:73",
        "ipAddress": "172.31.39.64"
      }
    ],
    "health": "SHUTDOWN",
    "agentId": "i-003a5e5e2b36cf8bd",
    "registeredTime": "2016-11-16T16:36:25Z",
    "lastHealthPingTime": "2016-11-16T16:47:37Z"
  }
]
```

- For API details, see [DescribeAgents](#) in *AWS CLI Command Reference*.

describe-configurations

The following code example shows how to use describe-configurations.

AWS CLI

Describe selected asset configurations

This example command describes the configurations of two specified servers. The action detects the type of asset from the configuration ID. Only one type of asset is allowed per command.

Command:

```
aws discovery describe-configurations --configuration-ids "d-server-099385097ef9fbcbf" "d-server-0c4f2dd1fee22c6c1"
```

Output:

```
{
  "configurations": [
    {
      "server.performance.maxCpuUsagePct": "0.0",
      "server.performance.maxDiskReadIOPS": "0.0",
      "server.performance.avgCpuUsagePct": "0.0",
      "server.type": "EC2",
      "server.performance.maxNetworkReadsPerSecondInKB": "0.19140625",
      "server.hostName": "ip-172-31-35-152",
      "server.configurationId": "d-server-0c4f2dd1fee22c6c1",
      "server.tags.hasMoreValues": "false",
      "server.performance.minFreeRAMInKB": "1543496.0",
      "server.osVersion": "3.14.48-33.39.amzn1.x86_64",
      "server.performance.maxDiskReadsPerSecondInKB": "0.0",
      "server.applications": "[]",
      "server.performance.numDisks": "1",
      "server.performance.numCpus": "1",
      "server.performance.numCores": "1",
      "server.performance.maxDiskWriteIOPS": "0.0",
      "server.performance.maxNetworkWritesPerSecondInKB": "0.82421875",
      "server.performance.avgDiskWritesPerSecondInKB": "0.0",
      "server.networkInterfaceInfo": "[{\"name\":\"eth0\",
\\\"macAddress\\\":\\\"06:A7:7D:3F:54:57\\\",\\\"ipAddress\\\":\\\"172.31.35.152\\\",\\\"netMask\\\":
\\\"255.255.240.0\\\"},{\"name\":\"lo\",\\\"macAddress\\\":\\\"00:00:00:00:00:00\\\",\\\"ipAddress
\\\":\\\"127.0.0.1\\\",\\\"netMask\\\":\\\"255.0.0.0\\\"},{\"name\":\"eth0\",\\\"macAddress\\\":
\\\"06:A7:7D:3F:54:57\\\",\\\"ipAddress\\\":\\\"fe80::4a7:7dff:fe3f:5457\\\"},{\"name\":\"lo\",
\\\"macAddress\\\":\\\"00:00:00:00:00:00\\\",\\\"ipAddress\\\":\\\"::1\\\"}]",
      "server.performance.avgNetworkReadsPerSecondInKB":
"0.049153645833333333",
      "server.tags": "[]",
      "server.applications.hasMoreValues": "false",
      "server.timeOfCreation": "2016-10-28 23:44:00.0",
    }
  ]
}
```

```

"server.agentId": "i-4447bc1b",
"server.performance.maxDiskWritesPerSecondInKB": "0.0",
"server.performance.avgDiskReadIOPS": "0.0",
"server.performance.avgFreeRAMInKB": "1547210.1333333333",
"server.performance.avgDiskReadsPerSecondInKB": "0.0",
"server.performance.avgDiskWriteIOPS": "0.0",
"server.performance.numNetworkCards": "2",
"server.hypervisor": "xen",
"server.networkInterfaceInfo.hasMoreValues": "false",
"server.performance.avgNetworkWritesPerSecondInKB": "0.1380859375",
"server.osName": "Linux - Amazon Linux AMI release 2015.03",
"server.performance.totalRAMInKB": "1694732.0",
"server.cpuType": "x64"
},
{
"server.performance.maxCpuUsagePct": "100.0",
"server.performance.maxDiskReadIOPS": "0.0",
"server.performance.avgCpuUsagePct": "14.733333333333333",
"server.type": "EC2",
"server.performance.maxNetworkReadsPerSecondInKB": "13.400390625",
"server.hostName": "ip-172-31-42-208",
"server.configurationId": "d-server-099385097ef9fbcfb",
"server.tags.hasMoreValues": "false",
"server.performance.minFreeRAMInKB": "1531104.0",
"server.osVersion": "3.14.48-33.39.amzn1.x86_64",
"server.performance.maxDiskReadsPerSecondInKB": "0.0",
"server.applications": "[]",
"server.performance.numDisks": "1",
"server.performance.numCpus": "1",
"server.performance.numCores": "1",
"server.performance.maxDiskWriteIOPS": "1.0",
"server.performance.maxNetworkWritesPerSecondInKB": "12.271484375",
"server.performance.avgDiskWritesPerSecondInKB":
"0.5333333333333334",
"server.networkInterfaceInfo": "[{"name":"eth0",
\"macAddress\":\"06:4A:79:60:75:61\", \"ipAddress\":\"172.31.42.208\", \"netMask
\": \"255.255.240.0\"}, {"name\":\"eth0\", \"macAddress\":\"06:4A:79:60:75:61\",
\"ipAddress\":\"fe80::44a:79ff:fe60:7561\"}, {"name\":\"lo\", \"macAddress\":
\"00:00:00:00:00:00\", \"ipAddress\":\"::1\"}, {"name\":\"lo\", \"macAddress\":
\"00:00:00:00:00:00\", \"ipAddress\":\"127.0.0.1\", \"netMask\":\"255.0.0.0\"}]",
"server.performance.avgNetworkReadsPerSecondInKB":
"2.8720052083333334",
"server.tags": "[]",
"server.applications.hasMoreValues": "false",

```

```

        "server.timeOfCreation": "2016-10-28 23:44:30.0",
        "server.agentId": "i-c142b99e",
        "server.performance.maxDiskWritesPerSecondInKB": "4.0",
        "server.performance.avgDiskReadIOPS": "0.0",
        "server.performance.avgFreeRAMInKB": "1534946.4",
        "server.performance.avgDiskReadsPerSecondInKB": "0.0",
        "server.performance.avgDiskWriteIOPS": "0.13333333333333336",
        "server.performance.numNetworkCards": "2",
        "server.hypervisor": "xen",
        "server.networkInterfaceInfo.hasMoreValues": "false",
        "server.performance.avgNetworkWritesPerSecondInKB":
"1.7977864583333332",
        "server.osName": "Linux - Amazon Linux AMI release 2015.03",
        "server.performance.totalRAMInKB": "1694732.0",
        "server.cpuType": "x64"
    }
]
}

```

Describe selected asset configurations

This example command describes the configurations of two specified applications. The action detects the type of asset from the configuration ID. Only one type of asset is allowed per command.

Command:

```
aws discovery describe-configurations --configuration-ids "d-
application-0ac39bc0e4fad0e42" "d-application-02444a45288013764q"
```

Output:

```

{
  "configurations": [
    {
      "application.serverCount": "0",
      "application.name": "Application-12345",
      "application.lastModifiedTime": "2016-12-13 23:53:27.0",
      "application.description": "",
      "application.timeOfCreation": "2016-12-13 23:53:27.0",
      "application.configurationId": "d-application-0ac39bc0e4fad0e42"
    },
    {

```

```

        "application.serverCount": "0",
        "application.name": "Application-67890",
        "application.lastModifiedTime": "2016-12-13 23:53:33.0",
        "application.description": "",
        "application.timeOfCreation": "2016-12-13 23:53:33.0",
        "application.configurationId": "d-application-02444a45288013764"
    }
]
}

```

- For API details, see [DescribeConfigurations](#) in *AWS CLI Command Reference*.

list-configurations

The following code example shows how to use `list-configurations`.

AWS CLI

To list all of the discovered servers meeting a set of filter conditions

This example command lists discovered servers matching either of two hostname patterns and not running Ubuntu.

Command:

```

aws discovery list-configurations --configuration-type SERVER --filters
name="server.hostName",values="172-31-35","172-31-42",condition="CONTAINS"
name="server.osName",values="Ubuntu",condition="NOT_CONTAINS"

```

Output:

```

{
  "configurations": [
    {
      "server.osVersion": "3.14.48-33.39.amzn1.x86_64",
      "server.type": "EC2",
      "server.hostName": "ip-172-31-42-208",
      "server.timeOfCreation": "2016-10-28 23:44:30.0",
      "server.configurationId": "d-server-099385097ef9fbcfb",
      "server.osName": "Linux - Amazon Linux AMI release 2015.03",
      "server.agentId": "i-c142b99e"
    },

```



```
    {
      "server.osVersion": "3.14.48-33.39.amzn1.x86_64",
      "server.type": "EC2",
      "server.hostName": "ip-172-31-35-152",
      "server.timeOfCreation": "2016-10-28 23:44:00.0",
      "server.configurationId": "d-server-0c4f2dd1fee22c6c1",
      "server.osName": "Linux - Amazon Linux AMI release 2015.03",
      "server.agentId": "i-4447bc1b"
    }
  ]
}
```

- For API details, see [ListConfigurations](#) in *AWS CLI Command Reference*.

AppRegistry examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AppRegistry.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-attribute-group

The following code example shows how to use `associate-attribute-group`.

AWS CLI

To associate an attribute group

The following `associate-attribute-group` example associates a specific attribute group in your AWS account to a specific application in your AWS account.

```
aws servicecatalog-appregistry associate-attribute-group \  
  --application "ExampleApplication" \  
  --attribute-group "ExampleAttributeGroup"
```

Output:

```
{  
  "applicationArn": "arn:aws:servicecatalog:us-west-2:813737243517:/  
applications/0ars38r6btoohvpvd9gqrptt91",  
  "attributeGroupArn": "arn:aws:servicecatalog:us-west-2:813737243517:/attribute-  
groups/01sj5xdwhbw54kejwnt09fnpc1"  
}
```

For more information, see [Associating and disassociating attribute groups](#) in the *AWS Service Catalog AppRegistry Administrator Guide*.

- For API details, see [AssociateAttributeGroup](#) in *AWS CLI Command Reference*.

create-application

The following code example shows how to use `create-application`.

AWS CLI

To create an application

The following `create-application` example creates a new application in your AWS account.

```
aws servicecatalog-appregistry create-application \  
  --name "ExampleApplication"
```

Output:

```
{  
  "application": {  
    "id": "0ars38r6btoohvpvd9gqrptt91",  
    "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/  
applications/0ars38r6btoohvpvd9gqrptt91",  
    "name": "ExampleApplication",
```

```
    "creationTime": "2023-02-28T21:10:10.820000+00:00",
    "lastUpdateTime": "2023-02-28T21:10:10.820000+00:00",
    "tags": {}
  }
}
```

For more information, see [Creating applications](#) in the *AWS Service Catalog AppRegistry Administrator Guide*.

- For API details, see [CreateApplication](#) in *AWS CLI Command Reference*.

create-attribute-group

The following code example shows how to use `create-attribute-group`.

AWS CLI

To create an attribute group

The following `create-attribute-group` example creates a new attribute group in your AWS account.

```
aws servicecatalog-appregistry create-attribute-group \
  --name "ExampleAttributeGroup" \
  --attributes '{"SomeKey1":"SomeValue1","SomeKey2":"SomeValue2"}'
```

Output:

```
{
  "attributeGroup": {
    "id": "01sj5xdwhbw54kejwnt09fnpc1",
    "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/attribute-
groups/01sj5xdwhbw54kejwnt09fnpc1",
    "name": "ExampleAttributeGroup",
    "creationTime": "2023-02-28T20:38:01.389000+00:00",
    "lastUpdateTime": "2023-02-28T20:38:01.389000+00:00",
    "tags": {}
  }
}
```

For more information, see [Creating attribute groups](#) in the *AWS Service Catalog AppRegistry Administrator Guide*.

- For API details, see [CreateAttributeGroup](#) in *AWS CLI Command Reference*.

delete-application

The following code example shows how to use delete-application.

AWS CLI

To delete an application

The following delete-application example deletes a specific application in your AWS account.

```
aws servicecatalog-appregistry delete-application \  
  --application "ExampleApplication3"
```

Output:

```
{  
  "application": {  
    "id": "055gw7aynr1i5mbv7kjwzx5945",  
    "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/  
applications/055gw7aynr1i5mbv7kjwzx5945",  
    "name": "ExampleApplication3",  
    "creationTime": "2023-02-28T22:06:28.228000+00:00",  
    "lastUpdateTime": "2023-02-28T22:06:28.228000+00:00"  
  }  
}
```

For more information, see [Deleting applications](#) in the *AWS Service Catalog AppRegistry Administrator Guide*.

- For API details, see [DeleteApplication](#) in *AWS CLI Command Reference*.

delete-attribute-group

The following code example shows how to use delete-attribute-group.

AWS CLI

Example 8: To delete an attribute group

The following `delete-attribute-group` example deletes a specific attribute group in your AWS account.

```
aws servicecatalog-appregistry delete-attribute-group \  
  --attribute-group "ExampleAttributeGroup3"
```

Output:

```
{  
  "attributeGroup": {  
    "id": "011ge6y3emyjijt8dw8jn6r0hv",  
    "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/attribute-  
groups/011ge6y3emyjijt8dw8jn6r0hv",  
    "name": "ExampleAttributeGroup3",  
    "creationTime": "2023-02-28T22:05:35.224000+00:00",  
    "lastUpdateTime": "2023-02-28T22:05:35.224000+00:00"  
  }  
}
```

For more information, see [Deleting attribute groups](#) in the *AWS Service Catalog AppRegistry Administrator Guide*.

- For API details, see [DeleteAttributeGroup](#) in *AWS CLI Command Reference*.

get-application

The following code example shows how to use `get-application`.

AWS CLI

To get an application

The following `get-application` example retrieves metadata information about a specific application in your AWS account.

```
aws servicecatalog-appregistry get-application \  
  --application "ExampleApplication"
```

Output:

```
{  
  "id": "0ars38r6btoohvpvd9gqrptt91",
```

```

    "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/
applications/0ars38r6btoohvpvd9gqrpvt91",
    "name": "ExampleApplication",
    "creationTime": "2023-02-28T21:10:10.820000+00:00",
    "lastUpdateTime": "2023-02-28T21:10:10.820000+00:00",
    "associatedResourceCount": 0,
    "tags": {
      "aws:servicecatalog:applicationName": "ExampleApplication"
    },
    "integrations": {
      "resourceGroup": {
        "state": "CREATE_COMPLETE",
        "arn": "arn:aws:resource-groups:us-west-2:813737243517:group/
AWS_AppRegistry_Application-ExampleApplication"
      }
    }
  }
}

```

For more information, see [Using Application details](#) in the *AWS Service Catalog AppRegistry Administrator Guide*.

- For API details, see [GetApplication](#) in *AWS CLI Command Reference*.

get-attribute-group

The following code example shows how to use `get-attribute-group`.

AWS CLI

To get an attribute group

The following `get-attribute-group` example retrieves a specific attribute group in your AWS account.

```

aws servicecatalog-appregistry get-attribute-group \
  --attribute-group "ExampleAttributeGroup"

```

Output:

```

{
  "id": "01sj5xdwhbw54kejwnt09fnpc1",
  "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/attribute-
groups/01sj5xdwhbw54kejwnt09fnpc1",

```

```
"name": "ExampleAttributeGroup",
"attributes": "{\"SomeKey1\":\"SomeValue1\", \"SomeKey2\":\"SomeValue2\"}",
"creationTime": "2023-02-28T20:38:01.389000+00:00",
"lastUpdateTime": "2023-02-28T20:38:01.389000+00:00",
"tags": {
  "aws:servicecatalog:attributeGroupName": "ExampleAttributeGroup"
}
}
```

For more information, see [Managing metadata for attribute groups](#) in the *AWS Service Catalog AppRegistry Administrator Guide*.

- For API details, see [GetAttributeGroup](#) in *AWS CLI Command Reference*.

list-applications

The following code example shows how to use `list-applications`.

AWS CLI

To list applications

The following `list-applications` example retrieves a list of all the applications in your AWS account.

```
aws servicecatalog-appregistry list-applications
```

Output:

```
{
  "applications": [
    {
      "id": "03axw94pjfj3uan00tcgbrxnkw",
      "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/
applications/03axw94pjfj3uan00tcgbrxnkw",
      "name": "ExampleApplication2",
      "creationTime": "2023-02-28T21:59:34.094000+00:00",
      "lastUpdateTime": "2023-02-28T21:59:34.094000+00:00"
    },
    {
      "id": "055gw7aynr1i5mbv7kjwzx5945",
      "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/
applications/055gw7aynr1i5mbv7kjwzx5945",
```

```

        "name": "ExampleApplication3",
        "creationTime": "2023-02-28T22:06:28.228000+00:00",
        "lastUpdateTime": "2023-02-28T22:06:28.228000+00:00"
    },
    {
        "id": "0ars38r6btoohvpvd9gqrptt91",
        "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/
applications/0ars38r6btoohvpvd9gqrptt91",
        "name": "ExampleApplication",
        "description": "This is an example application",
        "creationTime": "2023-02-28T21:10:10.820000+00:00",
        "lastUpdateTime": "2023-02-28T21:24:19.729000+00:00"
    }
]
}

```

For more information, see [Viewing application details](#) in the *AWS Service Catalog AppRegistry Administrator Guide*.

- For API details, see [ListApplications](#) in *AWS CLI Command Reference*.

list-associated-attribute-groups

The following code example shows how to use `list-associated-attribute-groups`.

AWS CLI

To list associated attribute groups

The following `list-associated-attribute-groups` example retrieves a list of all attribute groups in your AWS account that are associated with a specific application in your AWS account.

```
aws servicecatalog-appregistry list-associated-attribute-groups \
  --application "ExampleApplication"
```

Output:

```
{
  "attributeGroups": [
    "01sj5xdwhbw54kejwnt09fnpc1"
  ]
}
```


For more information, see [Associating and disassociating attribute groups](#) in the *AWS Service Catalog AppRegistry Administrator Guide*.

- For API details, see [ListAssociatedAttributeGroups](#) in *AWS CLI Command Reference*.

list-attribute-groups-for-application

The following code example shows how to use `list-attribute-groups-for-application`.

AWS CLI

To list attribute groups for an application

The following `list-attribute-groups-for-application` example lists the details of all attribute groups in your AWS account that are associated with a specific application in your AWS account.

```
aws servicecatalog-appregistry list-attribute-groups-for-application \  
  --application "ExampleApplication"
```

Output:

```
{  
  "attributeGroupsDetails": [  
    {  
      "id": "01sj5xdwhbw54kejwnt09fnpc1",  
      "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/attribute-  
groups/01sj5xdwhbw54kejwnt09fnpc1",  
      "name": "ExampleAttributeGroup"  
    }  
  ]  
}
```

For more information, see [Viewing attribute group details](#) in the *AWS Service Catalog AppRegistry Administrator Guide*.

- For API details, see [ListAttributeGroupsForApplication](#) in *AWS CLI Command Reference*.

list-attribute-groups

The following code example shows how to use `list-attribute-groups`.

AWS CLI

To list attribute groups

The following `list-attribute-groups` example retrieves a list of all attribute groups in your AWS account.

```
aws servicecatalog-appregistry list-attribute-groups
```

Output:

```
{
  "attributeGroups": [
    {
      "id": "011ge6y3emyjijt8dw8jn6r0hv",
      "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/attribute-
groups/011ge6y3emyjijt8dw8jn6r0hv",
      "name": "ExampleAttributeGroup3",
      "creationTime": "2023-02-28T22:05:35.224000+00:00",
      "lastUpdateTime": "2023-02-28T22:05:35.224000+00:00"
    },
    {
      "id": "01sj5xdwhbw54kejwnt09fnpc1",
      "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/attribute-
groups/01sj5xdwhbw54kejwnt09fnpc1",
      "name": "ExampleAttributeGroup",
      "description": "This is an example attribute group",
      "creationTime": "2023-02-28T20:38:01.389000+00:00",
      "lastUpdateTime": "2023-02-28T21:02:04.559000+00:00"
    },
    {
      "id": "03n1yffgq6d18vwrzxf0c70nm3",
      "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/attribute-
groups/03n1yffgq6d18vwrzxf0c70nm3",
      "name": "ExampleAttributeGroup2",
      "creationTime": "2023-02-28T21:57:30.687000+00:00",
      "lastUpdateTime": "2023-02-28T21:57:30.687000+00:00"
    }
  ]
}
```

For more information, see [Viewing attribute group details](#) in the *AWS Service Catalog AppRegistry Administrator Guide*.

- For API details, see [ListAttributeGroups](#) in *AWS CLI Command Reference*.

update-application

The following code example shows how to use update-application.

AWS CLI

To update an application

The following update-application example updates a specific application in your AWS account to include a description.

```
aws servicecatalog-appregistry update-application \  
  --application "ExampleApplication" \  
  --description "This is an example application"
```

Output:

```
{  
  "application": {  
    "id": "0ars38r6btoohvpvd9gqrptt91",  
    "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/  
applications/0ars38r6btoohvpvd9gqrptt91",  
    "name": "ExampleApplication",  
    "description": "This is an example application",  
    "creationTime": "2023-02-28T21:10:10.820000+00:00",  
    "lastUpdateTime": "2023-02-28T21:24:19.729000+00:00",  
    "tags": {  
      "aws:servicecatalog:applicationName": "ExampleApplication"  
    }  
  }  
}
```

For more information, see [Editing applications](#) in the *AWS Service Catalog AppRegistry Administrator Guide*.

- For API details, see [UpdateApplication](#) in *AWS CLI Command Reference*.

update-attribute-group

The following code example shows how to use update-attribute-group.

AWS CLI

To update an attribute group

The following `update-attribute-group` example updates a specific attribute group in your AWS account to include a description.

```
aws servicecatalog-appregistry update-attribute-group \  
  --attribute-group "ExampleAttributeGroup" \  
  --description "This is an example attribute group"
```

Output:

```
{  
  "attributeGroup": {  
    "id": "01sj5xdwhbw54kejwnt09fnpc1",  
    "arn": "arn:aws:servicecatalog:us-west-2:813737243517:/attribute-  
groups/01sj5xdwhbw54kejwnt09fnpc1",  
    "name": "ExampleAttributeGroup",  
    "description": "This is an example attribute group",  
    "creationTime": "2023-02-28T20:38:01.389000+00:00",  
    "lastUpdateTime": "2023-02-28T21:02:04.559000+00:00",  
    "tags": {  
      "aws:servicecatalog:attributeGroupName": "ExampleAttributeGroup"  
    }  
  }  
}
```

For more information, see [Editing attribute groups](#) in the *AWS Service Catalog AppRegistry Administrator Guide*.

- For API details, see [UpdateAttributeGroup](#) in *AWS CLI Command Reference*.

Athena examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Athena.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

batch-get-named-query

The following code example shows how to use `batch-get-named-query`.

AWS CLI

To return information about more than one query

The following `batch-get-named-query` example returns information about the named queries that have the specified IDs.

```
aws athena batch-get-named-query \
  --named-query-ids a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 a1b2c3d4-5678-90ab-cdef-
  EXAMPLE22222 a1b2c3d4-5678-90ab-cdef-EXAMPLE33333
```

Output:

```
{
  "NamedQueries": [
    {
      "Name": "Flights Select Query",
      "Description": "Sample query to get the top 10 airports with the most
number of departures since 2000",
      "Database": "sampledb",
      "QueryString": "SELECT origin, count(*) AS total_departures\nFROM
\nflights_parquet\nWHERE year >= '2000'\nGROUP BY origin\nORDER BY total_departures
DESC\nLIMIT 10;",
      "NamedQueryId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "WorkGroup": "primary"
    },
  ],
}
```

```

    {
      "Name": "Load flights table partitions",
      "Description": "Sample query to load flights table partitions using MSCK
REPAIR TABLE statement",
      "Database": "sampledb",
      "QueryString": "MSCK REPAIR TABLE flights_parquet;",
      "NamedQueryId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
      "WorkGroup": "primary"
    },
    {
      "Name": "CloudFront Select Query",
      "Description": "Sample query to view requests per operating system
during a particular time frame",
      "Database": "sampledb",
      "QueryString": "SELECT os, COUNT(*) count FROM cloudfront_logs WHERE
date BETWEEN date '2014-07-05' AND date '2014-08-05' GROUP BY os;",
      "NamedQueryId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
      "WorkGroup": "primary"
    }
  ],
  "UnprocessedNamedQueryIds": []
}

```

For more information, see [Running SQL Queries Using Amazon Athena](#) in the *Amazon Athena User Guide*.

- For API details, see [BatchGetNamedQuery](#) in *AWS CLI Command Reference*.

batch-get-query-execution

The following code example shows how to use `batch-get-query-execution`.

AWS CLI

To return information about one or more query executions

The following `batch-get-query-execution` example returns query execution information for the queries that have the specified query IDs.

```

aws athena batch-get-query-execution \
  --query-execution-ids a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 a1b2c3d4-5678-90ab-
cdef-EXAMPLE22222

```

Output:

```
{
  "QueryExecutions": [
    {
      "QueryExecutionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "Query": "create database if not exists webdata",
      "StatementType": "DDL",
      "ResultConfiguration": {
        "OutputLocation": "s3://awsdoc-example-bucket/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111.txt"
      },
      "QueryExecutionContext": {},
      "Status": {
        "State": "SUCCEEDED",
        "SubmissionDateTime": 1593470720.592,
        "CompletionDateTime": 1593470720.902
      },
      "Statistics": {
        "EngineExecutionTimeInMillis": 232,
        "DataScannedInBytes": 0,
        "TotalExecutionTimeInMillis": 310,
        "ResultConfiguration": {
          "QueryQueueTimeInMillis": 50,
          "ServiceProcessingTimeInMillis": 28
        },
        "WorkGroup": "AthenaAdmin"
      },
    },
    {
      "QueryExecutionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
      "Query": "select date, location, browser, uri, status from cloudfront_logs where method = 'GET' and status = 200 and location like 'SF0%' limit 10",
      "StatementType": "DML",
      "ResultConfiguration": {
        "OutputLocation": "s3://awsdoc-example-bucket/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222.csv"
      },
      "QueryExecutionContext": {
        "Database": "mydatabase",
        "Catalog": "awsdatacatalog"
      },
      "Status": {
```

```
        "State": "SUCCEEDED",
        "SubmissionDateTime": 1593469842.665,
        "CompletionDateTime": 1593469846.486
    },
    "Statistics": {
        "EngineExecutionTimeInMillis": 3600,
        "DataScannedInBytes": 203089,
        "TotalExecutionTimeInMillis": 3821,
        "QueryQueueTimeInMillis": 267,
        "QueryPlanningTimeInMillis": 1175
    },
    "WorkGroup": "AthenaAdmin"
}
],
"UnprocessedQueryExecutionIds": []
}
```

For more information, see [Running SQL Queries Using Amazon Athena](#) in the *Amazon Athena User Guide*.

- For API details, see [BatchGetQueryExecution](#) in *AWS CLI Command Reference*.

create-data-catalog

The following code example shows how to use create-data-catalog.

AWS CLI

To create a data catalog

The following create-data-catalog example creates the dynamo_db_catalog data catalog.

```
aws athena create-data-catalog \
  --name dynamo_db_catalog \
  --type LAMBDA \
  --description "DynamoDB Catalog" \
  --parameters function=arn:aws:lambda:us-
west-2:111122223333:function:dynamo_db_lambda
```

This command produces no output. To see the result, use `aws athena get-data-catalog --name dynamo_db_catalog`.

For more information, see [Registering a Catalog: create-data-catalog](#) in the *Amazon Athena User Guide*.

- For API details, see [CreateDataCatalog](#) in *AWS CLI Command Reference*.

create-named-query

The following code example shows how to use create-named-query.

AWS CLI

To create a named query

The following create-named-query example creates a saved query in the AthenaAdmin workgroup that queries the flights_parquet table for flights from Seattle to New York in January, 2016 whose departure and arrival were both delayed by more than ten minutes. Because the airport code values in the table are strings that include double quotes (for example, "SEA"), they are escaped by backslashes and surrounded by single quotes.

```
aws athena create-named-query \  
  --name "SEA to JFK delayed flights Jan 2016" \  
  --description "Both arrival and departure delayed more than 10 minutes." \  
  --database sampledb \  
  --query-string "SELECT flightdate, carrier, flightnum, origin, dest,  
  depdelayminutes, arrdelayminutes FROM sampledb.flights_parquet WHERE yr = 2016 AND  
  month = 1 AND origin = '\"SEA\"' AND dest = '\"JFK\"' AND depdelayminutes > 10 AND  
  arrdelayminutes > 10" \  
  --work-group AthenaAdmin
```

Output:

```
{  
  "NamedQueryId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"  
}
```

For more information, see [Running SQL Queries Using Amazon Athena](#) in the *Amazon Athena User Guide*.

- For API details, see [CreateNamedQuery](#) in *AWS CLI Command Reference*.

create-work-group

The following code example shows how to use `create-work-group`.

AWS CLI

To create a workgroup

The following `create-work-group` example creates a workgroup called `Data_Analyst_Group` that has the query results output location `s3://awsdoc-example-bucket`. The command creates a workgroup that overrides client configuration settings, which includes the query results output location. The command also enables CloudWatch metrics and adds three key-value tag pairs to the workgroup to distinguish it from other workgroups. Note that the `--configuration` argument has no spaces before the commas that separate its options.

```
aws athena create-work-group \  
  --name Data_Analyst_Group \  
  --configuration ResultConfiguration={OutputLocation="s3://awsdoc-example-  
bucket"},EnforceWorkGroupConfiguration="true",PublishCloudWatchMetricsEnabled="true"  
 \  
  --description "Workgroup for data analysts" \  
  --tags Key=Division,Value=West Key=Location,Value=Seattle Key=Team,Value="Big  
Data"
```

This command produces no output. To see the results, use `aws athena get-work-group --work-group Data_Analyst_Group`.

For more information, see [Managing Workgroups](#) in the *Amazon Athena User Guide*.

- For API details, see [CreateWorkGroup](#) in *AWS CLI Command Reference*.

delete-data-catalog

The following code example shows how to use `delete-data-catalog`.

AWS CLI

To delete a data catalog

The following `delete-data-catalog` example deletes the `UnusedDataCatalog` data catalog.

```
aws athena delete-data-catalog \  
  --name UnusedDataCatalog
```

This command produces no output.

For more information, see [Deleting a Catalog: delete-data-catalog](#) in the *Amazon Athena User Guide*.

- For API details, see [DeleteDataCatalog](#) in *AWS CLI Command Reference*.

delete-named-query

The following code example shows how to use delete-named-query.

AWS CLI

To delete a named query

The following delete-named-query example deletes the named query that has the specified ID.

```
aws athena delete-named-query \  
  --named-query-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

This command produces no output.

For more information, see [Running SQL Queries Using Amazon Athena](#) in the *Amazon Athena User Guide*.

- For API details, see [DeleteNamedQuery](#) in *AWS CLI Command Reference*.

delete-work-group

The following code example shows how to use delete-work-group.

AWS CLI

To delete a workgroup

The following delete-work-group example deletes the TeamB workgroup.

```
aws athena delete-work-group \  
  --work-group TeamB
```

```
--work-group TeamB
```

This command produces no output. To confirm the deletion, use `aws athena list-work-groups`.

For more information, see [Managing Workgroups](#) in the *Amazon Athena User Guide*.

- For API details, see [DeleteWorkGroup](#) in *AWS CLI Command Reference*.

get-data-catalog

The following code example shows how to use `get-data-catalog`.

AWS CLI

To return information about a data catalog

The following `get-data-catalog` example returns information about the `dynamo_db_catalog` data catalog.

```
aws athena get-data-catalog \  
  --name dynamo_db_catalog
```

Output:

```
{  
  "DataCatalog": {  
    "Name": "dynamo_db_catalog",  
    "Description": "DynamoDB Catalog",  
    "Type": "LAMBDA",  
    "Parameters": {  
      "catalog": "dynamo_db_catalog",  
      "metadata-function": "arn:aws:lambda:us-  
west-2:111122223333:function:dynamo_db_lambda",  
      "record-function": "arn:aws:lambda:us-  
west-2:111122223333:function:dynamo_db_lambda"  
    }  
  }  
}
```

For more information, see [Showing Catalog Details: get-data-catalog](#) in the *Amazon Athena User Guide*.

- For API details, see [GetDataCatalog](#) in *AWS CLI Command Reference*.

get-database

The following code example shows how to use get-database.

AWS CLI

To return information about a database in a data catalog

The following get-database example returns information about the sampledb database in the AwsDataCatalog data catalog.

```
aws athena get-database \  
  --catalog-name AwsDataCatalog \  
  --database-name sampledb
```

Output:

```
{  
  "Database": {  
    "Name": "sampledb",  
    "Description": "Sample database",  
    "Parameters": {  
      "CreatedBy": "Athena",  
      "EXTERNAL": "TRUE"  
    }  
  }  
}
```

For more information, see [Showing Database Details: get-database](#) in the *Amazon Athena User Guide*.

- For API details, see [GetDatabase](#) in *AWS CLI Command Reference*.

get-named-query

The following code example shows how to use get-named-query.

AWS CLI

To return a named query

The following `get-named-query` example returns information about the query that has the specified ID.

```
aws athena get-named-query \  
  --named-query-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{  
  "NamedQuery": {  
    "Name": "CloudFront Logs - SF0",  
    "Description": "Shows successful GET request data for SF0",  
    "Database": "default",  
    "QueryString": "select date, location, browser, uri, status from  
cloudfront_logs where method = 'GET' and status = 200 and location like 'SF0%'  
limit 10",  
    "NamedQueryId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
    "WorkGroup": "AthenaAdmin"  
  }  
}
```

For more information, see [Running SQL Queries Using Amazon Athena](#) in the *Amazon Athena User Guide*.

- For API details, see [GetNamedQuery](#) in *AWS CLI Command Reference*.

get-query-execution

The following code example shows how to use `get-query-execution`.

AWS CLI

To return information about a query execution

The following `get-query-execution` example returns information about the query that has the specified query ID.

```
aws athena get-query-execution \  
  --query-execution-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```

{
  "QueryExecution": {
    "QueryExecutionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "Query": "select date, location, browser, uri, status from cloudfront_logs
where method = 'GET
' and status = 200 and location like 'SF0%' limit 10",
    "StatementType": "DML",
    "ResultConfiguration": {
      "OutputLocation": "s3://awsdoc-example-bucket/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111.csv"
    },
    "QueryExecutionContext": {
      "Database": "mydatabase",
      "Catalog": "awsdatacatalog"
    },
    "Status": {
      "State": "SUCCEEDED",
      "SubmissionDateTime": 1593469842.665,
      "CompletionDateTime": 1593469846.486
    },
    "Statistics": {
      "EngineExecutionTimeInMillis": 3600,
      "DataScannedInBytes": 203089,
      "TotalExecutionTimeInMillis": 3821,
      "QueryQueueTimeInMillis": 267,
      "QueryPlanningTimeInMillis": 1175
    },
    "WorkGroup": "AthenaAdmin"
  }
}

```

For more information, see [Running SQL Queries Using Amazon Athena](#) in the *Amazon Athena User Guide*.

- For API details, see [GetQueryExecution](#) in *AWS CLI Command Reference*.

get-query-results

The following code example shows how to use `get-query-results`.

AWS CLI

To return the results of a query

The following `get-query-results` example returns the results of the query that has the specified query ID.

```
aws athena get-query-results \  
  --query-execution-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{  
  "ResultSet": {  
    "Rows": [  
      {  
        "Data": [  
          {  
            "VarCharValue": "date"  
          },  
          {  
            "VarCharValue": "location"  
          },  
          {  
            "VarCharValue": "browser"  
          },  
          {  
            "VarCharValue": "uri"  
          },  
          {  
            "VarCharValue": "status"  
          }  
        ]  
      },  
      {  
        "Data": [  
          {  
            "VarCharValue": "2014-07-05"  
          },  
          {  
            "VarCharValue": "SF04"  
          },  
          {  
            "VarCharValue": "Safari"  
          },  
          {  
            "VarCharValue": "/test-image-2.jpeg"  
          }  
        ]  
      }  
    ]  
  }  
}
```



```
    },
    {
      "VarCharValue": "200"
    }
  ]
},
{
  "Data": [
    {
      "VarCharValue": "2014-07-05"
    },
    {
      "VarCharValue": "SF04"
    },
    {
      "VarCharValue": "Opera"
    },
    {
      "VarCharValue": "/test-image-2.jpeg"
    },
    {
      "VarCharValue": "200"
    }
  ]
},
{
  "Data": [
    {
      "VarCharValue": "2014-07-05"
    },
    {
      "VarCharValue": "SF04"
    },
    {
      "VarCharValue": "Firefox"
    },
    {
      "VarCharValue": "/test-image-3.jpeg"
    },
    {
      "VarCharValue": "200"
    }
  ]
},
}
```

```
{
  "Data": [
    {
      "VarCharValue": "2014-07-05"
    },
    {
      "VarCharValue": "SF04"
    },
    {
      "VarCharValue": "Lynx"
    },
    {
      "VarCharValue": "/test-image-3.jpeg"
    },
    {
      "VarCharValue": "200"
    }
  ]
},
{
  "Data": [
    {
      "VarCharValue": "2014-07-05"
    },
    {
      "VarCharValue": "SF04"
    },
    {
      "VarCharValue": "IE"
    },
    {
      "VarCharValue": "/test-image-2.jpeg"
    },
    {
      "VarCharValue": "200"
    }
  ]
},
{
  "Data": [
    {
      "VarCharValue": "2014-07-05"
    },
    {
```

```
        "VarCharValue": "SF04"
      },
      {
        "VarCharValue": "Opera"
      },
      {
        "VarCharValue": "/test-image-1.jpeg"
      },
      {
        "VarCharValue": "200"
      }
    ]
  },
  {
    "Data": [
      {
        "VarCharValue": "2014-07-05"
      },
      {
        "VarCharValue": "SF04"
      },
      {
        "VarCharValue": "Chrome"
      },
      {
        "VarCharValue": "/test-image-3.jpeg"
      },
      {
        "VarCharValue": "200"
      }
    ]
  },
  {
    "Data": [
      {
        "VarCharValue": "2014-07-05"
      },
      {
        "VarCharValue": "SF04"
      },
      {
        "VarCharValue": "Firefox"
      }
    ]
  }
}
```

```
        "VarCharValue": "/test-image-2.jpeg"
      },
      {
        "VarCharValue": "200"
      }
    ]
  },
  {
    "Data": [
      {
        "VarCharValue": "2014-07-05"
      },
      {
        "VarCharValue": "SF04"
      },
      {
        "VarCharValue": "Chrome"
      },
      {
        "VarCharValue": "/test-image-3.jpeg"
      },
      {
        "VarCharValue": "200"
      }
    ]
  },
  {
    "Data": [
      {
        "VarCharValue": "2014-07-05"
      },
      {
        "VarCharValue": "SF04"
      },
      {
        "VarCharValue": "IE"
      },
      {
        "VarCharValue": "/test-image-2.jpeg"
      },
      {
        "VarCharValue": "200"
      }
    ]
  }
]
```

```
    }
  ],
  "ResultSetMetadata": {
    "ColumnInfo": [
      {
        "CatalogName": "hive",
        "SchemaName": "",
        "TableName": "",
        "Name": "date",
        "Label": "date",
        "Type": "date",
        "Precision": 0,
        "Scale": 0,
        "Nullable": "UNKNOWN",
        "CaseSensitive": false
      },
      {
        "CatalogName": "hive",
        "SchemaName": "",
        "TableName": "",
        "Name": "location",
        "Label": "location",
        "Type": "varchar",
        "Precision": 2147483647,
        "Data": [
          {
            "Scale": 0,
            "Nullable": "UNKNOWN",
            "CaseSensitive": true
          }
        ],
        "Data": [
          {
            "CatalogName": "hive",
            "SchemaName": "",
            "TableName": "",
            "Name": "browser",
            "Label": "browser",
            "Type": "varchar",
            "Precision": 2147483647,
            "Scale": 0,
            "Nullable": "UNKNOWN",
            "CaseSensitive": true
          }
        ],
        "Data": [
          {
            "CatalogName": "hive",
```

```
        "SchemaName": "",
        "TableName": "",
        "Name": "uri",
        "Label": "uri",
        "Type": "varchar",
        "Precision": 2147483647,
        "Scale": 0,
        "Nullable": "UNKNOWN",
        "CaseSensitive": true
    },
    {
        "CatalogName": "hive",
        "SchemaName": "",
        "TableName": "",
        "Name": "status",
        "Label": "status",
        "Type": "integer",
        "Precision": 10,
        "Scale": 0,
        "Nullable": "UNKNOWN",
        "CaseSensitive": false
    }
]
}
},
"UpdateCount": 0
}
```

For more information, see [Working with Query Results, Output Files, and Query History](#) in the *Amazon Athena User Guide*.

- For API details, see [GetQueryResults](#) in *AWS CLI Command Reference*.

get-table-metadata

The following code example shows how to use `get-table-metadata`.

AWS CLI

To return metadata information about a table

The following `get-table-metadata` example returns metadata information about the `counties` table, including including column names and their datatypes, from the `sampledb` database of the `AwsDataCatalog` data catalog.

```
aws athena get-table-metadata \  
  --catalog-name AwsDataCatalog \  
  --database-name sampledb \  
  --table-name counties
```

Output:

```
{  
  "TableMetadata": {  
    "Name": "counties",  
    "CreateTime": 1593559968.0,  
    "LastAccessTime": 0.0,  
    "TableType": "EXTERNAL_TABLE",  
    "Columns": [  
      {  
        "Name": "name",  
        "Type": "string",  
        "Comment": "from deserializer"  
      },  
      {  
        "Name": "boundaryshape",  
        "Type": "binary",  
        "Comment": "from deserializer"  
      },  
      {  
        "Name": "motto",  
        "Type": "string",  
        "Comment": "from deserializer"  
      },  
      {  
        "Name": "population",  
        "Type": "int",  
        "Comment": "from deserializer"  
      }  
    ],  
    "PartitionKeys": [],  
    "Parameters": {  
      "EXTERNAL": "TRUE",  
      "inputformat": "com.esri.json.hadoop.EnclosedJsonInputFormat",
```

```

        "location": "s3://awsdoc-example-bucket/json",
        "outputformat":
"org.apache.hadoop.hive.q1.io.HiveIgnoreKeyTextOutputFormat",
        "serde.param.serialization.format": "1",
        "serde.serialization.lib": "com.esri.hadoop.hive.serde.JsonSerde",
        "transient_lastDdlTime": "1593559968"
    }
}
}

```

For more information, see [Showing Table Details: get-table-metadata](#) in the *Amazon Athena User Guide*.

- For API details, see [GetTableMetadata](#) in *AWS CLI Command Reference*.

get-work-group

The following code example shows how to use get-work-group.

AWS CLI

To return information about a workgroup

The following get-work-group example returns information about the AthenaAdmin workgroup.

```
aws athena get-work-group \
  --work-group AthenaAdmin
```

Output:

```

{
  "WorkGroup": {
    "Name": "AthenaAdmin",
    "State": "ENABLED",
    "Configuration": {
      "ResultConfiguration": {
        "OutputLocation": "s3://awsdoc-example-bucket/"
      },
      "EnforceWorkGroupConfiguration": false,
      "PublishCloudWatchMetricsEnabled": true,
      "RequesterPaysEnabled": false
    }
  },
}

```



```
    "Description": "Workgroup for Athena administrators",
    "CreationTime": 1573677174.105
  }
}
```

For more information, see [Managing Workgroups](#) in the *Amazon Athena User Guide*.

- For API details, see [GetWorkGroup](#) in *AWS CLI Command Reference*.

list-data-catalogs

The following code example shows how to use `list-data-catalogs`.

AWS CLI

To list the data catalogs registered with Athena

The following `list-data-catalogs` example lists the data catalogs registered with Athena.

```
aws athena list-data-catalogs
```

Output:

```
{
  "DataCatalogsSummary": [
    {
      "CatalogName": "AwsDataCatalog",
      "Type": "GLUE"
    },
    {
      "CatalogName": "cw_logs_catalog",
      "Type": "LAMBDA"
    },
    {
      "CatalogName": "cw_metrics_catalog",
      "Type": "LAMBDA"
    }
  ]
}
```

For more information, see [Listing Registered Catalogs: list-data-catalogs](#) in the *Amazon Athena User Guide*.

- For API details, see [ListDataCatalogs](#) in *AWS CLI Command Reference*.

list-databases

The following code example shows how to use `list-databases`.

AWS CLI

To list the databases in a data catalog

The following `list-databases` example lists the databases in the `AwsDataCatalog` data catalog.

```
aws athena list-databases \  
  --catalog-name AwsDataCatalog
```

Output:

```
{  
  "DatabaseList": [  
    {  
      "Name": "default"  
    },  
    {  
      "Name": "mydatabase"  
    },  
    {  
      "Name": "newdb"  
    },  
    {  
      "Name": "sampledb",  
      "Description": "Sample database",  
      "Parameters": {  
        "CreatedBy": "Athena",  
        "EXTERNAL": "TRUE"  
      }  
    },  
    {  
      "Name": "webdata"  
    }  
  ]  
}
```

For more information, see [Listing Databases in a Catalog: list-databases](#) in the *Amazon Athena User Guide*.

- For API details, see [ListDatabases](#) in *AWS CLI Command Reference*.

list-named-queries

The following code example shows how to use `list-named-queries`.

AWS CLI

To list the named queries for a workgroup

The following `list-named-queries` example lists the named queries for the `AthenaAdmin` workgroup.

```
aws athena list-named-queries \  
  --work-group AthenaAdmin
```

Output:

```
{  
  "NamedQueryIds": [  
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",  
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333"  
  ]  
}
```

For more information, see [Running SQL Queries Using Amazon Athena](#) in the *Amazon Athena User Guide*.

- For API details, see [ListNamedQueries](#) in *AWS CLI Command Reference*.

list-query-executions

The following code example shows how to use `list-query-executions`.

AWS CLI

To list the query IDs of the queries in a specified workgroup

The following `list-query-executions` example lists a maximum of ten of the query IDs in the AthenaAdmin workgroup.

```
aws athena list-query-executions \  
  --work-group AthenaAdmin \  
  --max-items 10
```

Output:

```
{  
  "QueryExecutionIds": [  
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE11110",  
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",  
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",  
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE11114",  
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE11115",  
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE11116",  
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE11117",  
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE11118",  
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE11119"  
  ],  
  "NextToken": "eyJ0ZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAxMH0="
```

For more information, see [Working with Query Results, Output Files, and Query History](#) in the *Amazon Athena User Guide*.

- For API details, see [ListQueryExecutions](#) in *AWS CLI Command Reference*.

list-table-metadata

The following code example shows how to use `list-table-metadata`.

AWS CLI

To list the metadata for tables in the specified database of a data catalog

The following `list-table-metadata` example returns metadata information for a maximum of two tables in the geography database of the AwsDataCatalog data catalog.

```
aws athena list-table-metadata \  
  --work-group AthenaAdmin \  
  --max-items 2
```

```
--catalog-name AwsDataCatalog \  
--database-name geography \  
--max-items 2
```

Output:

```
{  
  "TableMetadataList": [  
    {  
      "Name": "country_codes",  
      "CreateTime": 1586553454.0,  
      "TableType": "EXTERNAL_TABLE",  
      "Columns": [  
        {  
          "Name": "country",  
          "Type": "string",  
          "Comment": "geo id"  
        },  
        {  
          "Name": "alpha-2 code",  
          "Type": "string",  
          "Comment": "geo id2"  
        },  
        {  
          "Name": "alpha-3 code",  
          "Type": "string",  
          "Comment": "state name"  
        },  
        {  
          "Name": "numeric code",  
          "Type": "bigint",  
          "Comment": ""  
        },  
        {  
          "Name": "latitude",  
          "Type": "bigint",  
          "Comment": "location (latitude)"  
        },  
        {  
          "Name": "longitude",  
          "Type": "bigint",  
          "Comment": "location (longitude)"  
        }  
      ]  
    }  
  ]  
}
```

```
    ],
    "Parameters": {
      "areColumnsQuoted": "false",
      "classification": "csv",
      "columnsOrdered": "true",
      "delimiter": ",",
      "has_encrypted_data": "false",
      "inputformat": "org.apache.hadoop.mapred.TextInputFormat",
      "location": "s3://awsdoc-example-bucket/csv/countrycode",
      "outputformat":
"org.apache.hadoop.hive ql.io.HiveIgnoreKeyTextOutputFormat",
      "serde.param.field.delim": ",",
      "serde.serialization.lib":
"org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe",
      "skip.header.line.count": "1",
      "typeOfData": "file"
    }
  },
  {
    "Name": "county_populations",
    "CreateTime": 1586553446.0,
    "TableType": "EXTERNAL_TABLE",
    "Columns": [
      {
        "Name": "id",
        "Type": "string",
        "Comment": "geo id"
      },
      {
        "Name": "country",

        "Name": "id2",
        "Type": "string",
        "Comment": "geo id2"
      },
      {
        "Name": "county",
        "Type": "string",
        "Comment": "county name"
      },
      {
        "Name": "state",
        "Type": "string",
        "Comment": "state name"
      }
    ]
  }
}
```

```

    },
    {
      "Name": "population estimate 2018",
      "Type": "string",
      "Comment": ""
    }
  ],
  "Parameters": {
    "areColumnsQuoted": "false",
    "classification": "csv",
    "columnsOrdered": "true",
    "delimiter": ",",
    "has_encrypted_data": "false",
    "inputformat": "org.apache.hadoop.mapred.TextInputFormat",
    "location": "s3://awsdoc-example-bucket/csv/CountyPopulation",
    "outputformat":
"org.apache.hadoop.hive.q1.io.HiveIgnoreKeyTextOutputFormat",
    "serde.param.field.delim": ",",
    "serde.serialization.lib":
"org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe",
    "skip.header.line.count": "1",
    "typeOfData": "file"
  }
}
],
  "NextToken": "eyJ0ZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAyfQ=="
}

```

For more information, see [Showing Metadata for All Tables in a Database: list-table-metadata](#) in the *Amazon Athena User Guide*.

- For API details, see [ListTableMetadata](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

Example 1: To list the tags for a workgroup

The following `list-tags-for-resource` example lists the tags for the `Data_Analyst_Group` workgroup.

```
aws athena list-tags-for-resource \  
  --resource-arn arn:aws:athena:us-west-2:111122223333:workgroup/  
Data_Analyst_Group
```

Output:

```
{  
  "Tags": [  
    {  
      "Key": "Division",  
      "Value": "West"  
    },  
    {  
      "Key": "Team",  
      "Value": "Big Data"  
    },  
    {  
      "Key": "Location",  
      "Value": "Seattle"  
    }  
  ]  
}
```

Example 2: To list the tags for a data catalog

The following `list-tags-for-resource` example lists the tags for the `dynamo_db_catalog` data catalog.

```
aws athena list-tags-for-resource \  
  --resource-arn arn:aws:athena:us-west-2:111122223333:datacatalog/  
dynamo_db_catalog
```

Output:

```
{  
  "Tags": [  
    {  
      "Key": "Division",  
      "Value": "Mountain"  
    },  
    {
```



```
        "Key": "Organization",
        "Value": "Retail"
    },
    {
        "Key": "Product_Line",
        "Value": "Shoes"
    },
    {
        "Key": "Location",
        "Value": "Denver"
    }
]
}
```

For more information, see [Listing the tags for a resource: list-tags-for-resource](#) in the *Amazon Athena User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

list-work-groups

The following code example shows how to use `list-work-groups`.

AWS CLI

To list workgroups

The following `list-work-groups` example lists the workgroups in the current account.

```
aws athena list-work-groups
```

Output:

```
{
  "WorkGroups": [
    {
      "Name": "Data_Analyst_Group",
      "State": "ENABLED",
      "Description": "",
      "CreationTime": 1578006683.016
    },
    {
```

```

        "Name": "AthenaAdmin",
        "State": "ENABLED",
        "Description": "",
        "CreationTime": 1573677174.105
    },
    {
        "Name": "primary",
        "State": "ENABLED",
        "Description": "",
        "CreationTime": 1567465222.723
    }
]
}

```

For more information, see [Managing Workgroups](#) in the *Amazon Athena User Guide*.

- For API details, see [ListWorkGroups](#) in *AWS CLI Command Reference*.

start-query-execution

The following code example shows how to use start-query-execution.

AWS CLI

Example 1: To run a query in a workgroup on the specified table in the specified database and data catalog

The following start-query-execution example uses the AthenaAdmin workgroup to run a query on the cloudfront_logs table in the cflogsdatabase in the AwsDataCatalog data catalog.

```

aws athena start-query-execution \
  --query-string "select date, location, browser, uri, status from cloudfront_logs
  where method = 'GET' and status = 200 and location like 'SF0%' limit 10" \
  --work-group "AthenaAdmin" \
  --query-execution-context Database=cflogsdatabase,Catalog=AwsDataCatalog

```

Output:

```

{
  "QueryExecutionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}

```

```
}
```

For more information, see [Running SQL Queries Using Amazon Athena](#) in the *Amazon Athena User Guide*.

Example 2: To run a query that uses a specified workgroup to create a database in the specified data catalog

The following `start-query-execution` example uses the `AthenaAdmin` workgroup to create the database `newdb` in the default data catalog `AwsDataCatalog`.

```
aws athena start-query-execution \  
  --query-string "create database if not exists newdb" \  
  --work-group "AthenaAdmin"
```

Output:

```
{  
  "QueryExecutionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11112"  
}
```

For more information, see [Running SQL Queries Using Amazon Athena](#) in the *Amazon Athena User Guide*.

Example 3: To run a query that creates a view on a table in the specified database and data catalog

The following `start-query-execution` example uses a `SELECT` statement on the `cloudfront_logs` table in the `cflogsdatabase` to create the view `cf10`.

```
aws athena start-query-execution \  
  --query-string "CREATE OR REPLACE VIEW cf10 AS SELECT * FROM cloudfront_logs  
  limit 10" \  
  --query-execution-context Database=cflogsdatabase
```

Output:

```
{  
  "QueryExecutionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11113"  
}
```

```
}
```

For more information, see [Running SQL Queries Using Amazon Athena](#) in the *Amazon Athena User Guide*.

- For API details, see [StartQueryExecution](#) in *AWS CLI Command Reference*.

stop-query-execution

The following code example shows how to use stop-query-execution.

AWS CLI

To stop a running query

The following stop-query-execution example stops the query that has the specified query ID.

```
aws athena stop-query-execution \  
  --query-execution-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

This command produces no output.

For more information, see [Running SQL Queries Using Amazon Athena](#) in the *Amazon Athena User Guide*.

- For API details, see [StopQueryExecution](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To add a tag to a resource

The following tag-resource example adds three tags to the dynamo_db_catalog data catalog.

```
aws athena tag-resource \  
  --resource-id dynamo_db_catalog
```

```
--resource-arn arn:aws:athena:us-west-2:111122223333:datacatalog/  
dynamo_db_catalog \  
--tags Key=Organization,Value=Retail Key=Division,Value=Mountain  
Key=Product_Line,Value=Shoes Key=Location,Value=Denver
```

This command produces no output. To see the result, use `aws athena list-tags-for-resource --resource-arn arn:aws:athena:us-west-2:111122223333:datacatalog/dynamo_db_catalog`.

For more information, see [Adding tags to a resource: tag-resource](#) in the *Amazon Athena User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag from a resource

The following `untag-resource` example removes the `Specialization` and `Focus` keys and their associated values from the `dynamo_db_catalog` data catalog resource.

```
aws athena untag-resource \  
--resource-arn arn:aws:athena:us-west-2:111122223333:datacatalog/  
dynamo_db_catalog \  
--tag-keys Specialization Focus
```

This command produces no output. To see the results, use the `list-tags-for-resource` command.

For more information, see [Removing tags from a resource: untag-resource](#) in the *Amazon Athena User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-data-catalog

The following code example shows how to use `update-data-catalog`.

AWS CLI

To update a data catalog

The following `update-data-catalog` example updates the Lambda function and description of the `cw_logs_catalog` data catalog.

```
aws athena update-data-catalog \  
  --name cw_logs_catalog \  
  --type LAMBDA \  
  --description "New CloudWatch Logs Catalog" \  
  --function=arn:aws:lambda:us-west-2:111122223333:function:new_cw_logs_lambda
```

This command produces no output. To see the result, use `aws athena get-data-catalog --name cw_logs_catalog`.

For more information, see [Updating a Catalog: update-data-catalog](#) in the *Amazon Athena User Guide*.

- For API details, see [UpdateDataCatalog](#) in *AWS CLI Command Reference*.

update-work-group

The following code example shows how to use `update-work-group`.

AWS CLI

To update a workgroup

The following `update-work-group` example disables the `Data_Analyst_Group` workgroup. Users cannot run or create queries in the disabled workgroup, but can still view metrics, data usage limit controls, workgroup settings, query history, and saved queries.

```
aws athena update-work-group \  
  --work-group Data_Analyst_Group \  
  --state DISABLED
```

This command produces no output. To verify the change in state, use `aws athena get-work-group --work-group Data_Analyst_Group` and check the `State` property in the output.

For more information, see [Managing Workgroups](#) in the *Amazon Athena User Guide*.

- For API details, see [UpdateWorkGroup](#) in *AWS CLI Command Reference*.

Auto Scaling examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Auto Scaling.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

attach-instances

The following code example shows how to use `attach-instances`.

AWS CLI

To attach an instance to an Auto Scaling group

This example attaches the specified instance to the specified Auto Scaling group.

```
aws autoscaling attach-instances \  
  --instance-ids i-061c63c5eb45f0416 \  
  --auto-scaling-group-name my-asg
```

This command produces no output.

- For API details, see [AttachInstances](#) in *AWS CLI Command Reference*.

attach-load-balancer-target-groups

The following code example shows how to use `attach-load-balancer-target-groups`.

AWS CLI

To attach a target group to an Auto Scaling group

This example attaches the specified target group to the specified Auto Scaling group.

```
aws autoscaling attach-load-balancer-target-groups \  
  --auto-scaling-group-name my-asg \  
  --target-group-arns arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

This command produces no output.

For more information, see [Elastic Load Balancing and Amazon EC2 Auto Scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [AttachLoadBalancerTargetGroups](#) in *AWS CLI Command Reference*.

attach-load-balancers

The following code example shows how to use `attach-load-balancers`.

AWS CLI

To attach a Classic Load Balancer to an Auto Scaling group

This example attaches the specified Classic Load Balancer to the specified Auto Scaling group.

```
aws autoscaling attach-load-balancers \  
  --load-balancer-names my-load-balancer \  
  --auto-scaling-group-name my-asg
```

This command produces no output.

For more information, see [Elastic Load Balancing and Amazon EC2 Auto Scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [AttachLoadBalancers](#) in *AWS CLI Command Reference*.

cancel-instance-refresh

The following code example shows how to use `cancel-instance-refresh`.

AWS CLI

To cancel an instance refresh

The following `cancel-instance-refresh` example cancels an in-progress instance refresh for the specified Auto Scaling group.

```
aws autoscaling cancel-instance-refresh \  
  --auto-scaling-group-name my-asg
```

Output:

```
{  
  "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"  
}
```

For more information, see [Cancel an instance refresh](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [CancelInstanceRefresh](#) in *AWS CLI Command Reference*.

complete-lifecycle-action

The following code example shows how to use `complete-lifecycle-action`.

AWS CLI

To complete the lifecycle action

This example notifies Amazon EC2 Auto Scaling that the specified lifecycle action is complete so that it can finish launching or terminating the instance.

```
aws autoscaling complete-lifecycle-action \  
  --lifecycle-hook-name my-launch-hook \  
  --auto-scaling-group-name my-asg \  
  --lifecycle-action-result CONTINUE \  
  --lifecycle-action-token bcd2f1b8-9a78-44d3-8a7a-4dd07d7cf635
```

This command produces no output.

For more information, see [Amazon EC2 Auto Scaling lifecycle hooks](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [CompleteLifecycleAction](#) in *AWS CLI Command Reference*.

create-auto-scaling-group

The following code example shows how to use `create-auto-scaling-group`.

AWS CLI

Example 1: To create an Auto Scaling group

The following `create-auto-scaling-group` example creates an Auto Scaling group in subnets in multiple Availability Zones within a Region. The instances launch with the default version of the specified launch template. Note that defaults are used for most other settings, such as the termination policies and health check configuration.

```
aws autoscaling create-auto-scaling-group \  
  --auto-scaling-group-name my-asg \  
  --launch-template LaunchTemplateId=lt-1234567890abcde12 \  
  --min-size 1 \  
  --max-size 5 \  
  --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

This command produces no output.

For more information, see [Auto Scaling groups](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 2: To attach an Application Load Balancer, Network Load Balancer, or Gateway Load Balancer

This example specifies the ARN of a target group for a load balancer that supports the expected traffic. The health check type specifies ELB so that when Elastic Load Balancing reports an instance as unhealthy, the Auto Scaling group replaces it. The command also defines a health check grace period of 600 seconds. The grace period helps prevent premature termination of newly launched instances.

```
aws autoscaling create-auto-scaling-group \  
  --auto-scaling-group-name my-asg \  
  --launch-template LaunchTemplateId=lt-1234567890abcde12 \  
  --target-group-arns arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-targets/943f017f100becff \  
  --health-check-type ELB \  

```

```
--health-check-grace-period 600 \  
--min-size 1 \  
--max-size 5 \  
--vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

This command produces no output.

For more information, see [Elastic Load Balancing and Amazon EC2 Auto Scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 3: To specify a placement group and use the latest version of the launch template

This example launches instances into a placement group within a single Availability Zone. This can be useful for low-latency groups with HPC workloads. This example also specifies the minimum size, maximum size, and desired capacity of the group.

```
aws autoscaling create-auto-scaling-group \  
  --auto-scaling-group-name my-asg \  
  --launch-template LaunchTemplateId=lt-1234567890abcde12,Version='$Latest' \  
  --min-size 1 \  
  --max-size 5 \  
  --desired-capacity 3 \  
  --placement-group my-placement-group \  
  --vpc-zone-identifier "subnet-6194ea3b"
```

This command produces no output.

For more information, see [Placement groups](#) in the *Amazon EC2 User Guide for Linux Instances*.

Example 4: To specify a single instance Auto Scaling group and use a specific version of the launch template

This example creates an Auto Scaling group with minimum and maximum capacity set to 1 to enforce that one instance will be running. The command also specifies v1 of a launch template in which the ID of an existing ENI is specified. When you use a launch template that specifies an existing ENI for eth0, you must specify an Availability Zone for the Auto Scaling group that matches the network interface, without also specifying a subnet ID in the request.

```
aws autoscaling create-auto-scaling-group \  
  --auto-scaling-group-name my-asg-single-instance \  
  --launch-template LaunchTemplateName=my-template-for-auto-scaling,Version='1' \  
  --availability-zone us-east-1a
```

```
--min-size 1 \  
--max-size 1 \  
--availability-zones us-west-2a
```

This command produces no output.

For more information, see [Auto Scaling groups](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 5: To specify a different termination policy

This example creates an Auto Scaling group using a launch configuration and sets the termination policy to terminate the oldest instances first. The command also applies a tag to the group and its instances, with a key of Role and a value of WebServer.

```
aws autoscaling create-auto-scaling-group \  
  --auto-scaling-group-name my-asg \  
  --launch-configuration-name my-lc \  
  --min-size 1 \  
  --max-size 5 \  
  --termination-policies "OldestInstance" \  
  --tags "ResourceId=my-asg,ResourceType=auto-scaling-  
group,Key=Role,Value=WebServer,PropagateAtLaunch=true" \  
  --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

This command produces no output.

For more information, see [Working with Amazon EC2 Auto Scaling termination policies](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 6: To specify a launch lifecycle hook

This example creates an Auto Scaling group with a lifecycle hook that supports a custom action at instance launch.

```
aws autoscaling create-auto-scaling-group \  
  --cli-input-json file://~/config.json
```

Contents of config.json file:

```
{  
  "AutoScalingGroupName": "my-asg",
```

```

"LaunchTemplate": {
  "LaunchTemplateId": "lt-1234567890abcde12"
},
"LifecycleHookSpecificationList": [{
  "LifecycleHookName": "my-launch-hook",
  "LifecycleTransition": "autoscaling:EC2_INSTANCE_LAUNCHING",
  "NotificationTargetARN": "arn:aws:sqs:us-west-2:123456789012:my-sqs-queue",
  "RoleARN": "arn:aws:iam::123456789012:role/my-notification-role",
  "NotificationMetadata": "SQS message metadata",
  "HeartbeatTimeout": 4800,
  "DefaultResult": "ABANDON"
}],
"MinSize": 1,
"MaxSize": 5,
"VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
"Tags": [{
  "ResourceType": "auto-scaling-group",
  "ResourceId": "my-asg",
  "PropagateAtLaunch": true,
  "Value": "test",
  "Key": "environment"
}]
}

```

This command produces no output.

For more information, see [Amazon EC2 Auto Scaling lifecycle hooks](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 7: To specify a termination lifecycle hook

This example creates an Auto Scaling group with a lifecycle hook that supports a custom action at instance termination.

```

aws autoscaling create-auto-scaling-group \
  --cli-input-json file://~/config.json

```

Contents of config.json:

```

{
  "AutoScalingGroupName": "my-asg",
  "LaunchTemplate": {

```

```
    "LaunchTemplateId": "lt-1234567890abcde12"
  },
  "LifecycleHookSpecificationList": [{
    "LifecycleHookName": "my-termination-hook",
    "LifecycleTransition": "autoscaling:EC2_INSTANCE_TERMINATING",
    "HeartbeatTimeout": 120,
    "DefaultResult": "CONTINUE"
  }],
  "MinSize": 1,
  "MaxSize": 5,
  "TargetGroupARNs": [
    "arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-
    targets/73e2d6bc24d8a067"
  ],
  "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
}
```

This command produces no output.

For more information, see [Amazon EC2 Auto Scaling lifecycle hooks](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 8: To specify a custom termination policy

This example creates an Auto Scaling group that specifies a custom Lambda function termination policy that tells Amazon EC2 Auto Scaling which instances are safe to terminate on scale in.

```
aws autoscaling create-auto-scaling-group \
  --auto-scaling-group-name my-asg-single-instance \
  --launch-template LaunchTemplateName=my-template-for-auto-scaling \
  --min-size 1 \
  --max-size 5 \
  --termination-policies "arn:aws:lambda:us-
  west-2:123456789012:function:HelloFunction:prod" \
  --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

This command produces no output.

For more information, see [Creating a custom termination policy with Lambda](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [CreateAutoScalingGroup](#) in *AWS CLI Command Reference*.

create-launch-configuration

The following code example shows how to use create-launch-configuration.

AWS CLI

Example 1: To create a launch configuration

This example creates a simple launch configuration.

```
aws autoscaling create-launch-configuration \  
  --launch-configuration-name my-lc \  
  --image-id ami-04d5cc9b88example \  
  --instance-type m5.large
```

This command produces no output.

For more information, see [Creating a launch configuration](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 2: To create a launch configuration with a security group, key pair, and bootstrapping script

This example creates a launch configuration with a security group, a key pair, and a bootstrapping script contained in the user data.

```
aws autoscaling create-launch-configuration \  
  --launch-configuration-name my-lc \  
  --image-id ami-04d5cc9b88example \  
  --instance-type m5.large \  
  --security-groups sg-eb2af88example \  
  --key-name my-key-pair \  
  --user-data file://myuserdata.txt
```

This command produces no output.

For more information, see [Creating a launch configuration](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 3: To create a launch configuration with an IAM role

This example creates a launch configuration with the instance profile name of an IAM role.

```
aws autoscaling create-launch-configuration \  
  --launch-configuration-name my-lc \  
  --image-id ami-04d5cc9b88example \  
  --instance-type m5.large \  
  --iam-instance-profile my-autoscaling-role
```

This command produces no output.

For more information, see [IAM role for applications that run on Amazon EC2 instances](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 4: To create a launch configuration with detailed monitoring enabled

This example creates a launch configuration with EC2 detailed monitoring enabled, which sends EC2 metrics to CloudWatch in 1-minute periods.

```
aws autoscaling create-launch-configuration \  
  --launch-configuration-name my-lc \  
  --image-id ami-04d5cc9b88example \  
  --instance-type m5.large \  
  --instance-monitoring Enabled=true
```

This command produces no output.

For more information, see [Configuring monitoring for Auto Scaling instances](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 5: To create a launch configuration that launches Spot Instances

This example creates a launch configuration that uses Spot Instances as the only purchase option.

```
aws autoscaling create-launch-configuration \  
  --launch-configuration-name my-lc \  
  --image-id ami-04d5cc9b88example \  
  --instance-type m5.large \  
  --spot-price "0.50"
```

This command produces no output.

For more information, see [Requesting Spot Instances](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 6: To create a launch configuration using an EC2 instance

This example creates a launch configuration based on the attributes of an existing instance. It overrides the placement tenancy and whether a public IP address is set by including the `--placement-tenancy` and `--no-associate-public-ip-address` options.

```
aws autoscaling create-launch-configuration \  
  --launch-configuration-name my-lc-from-instance \  
  --instance-id i-0123a456700123456 \  
  --instance-type m5.large \  
  --no-associate-public-ip-address \  
  --placement-tenancy dedicated
```

This command produces no output.

For more information, see [Creating a launch configuration using an EC2 instance](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 7: To create a launch configuration with a block device mapping for an Amazon EBS volume

This example creates a launch configuration with a block device mapping for an Amazon EBS gp3 volume with the device name `/dev/sdh` and a volume size of 20.

```
aws autoscaling create-launch-configuration \  
  --launch-configuration-name my-lc \  
  --image-id ami-04d5cc9b88example \  
  --instance-type m5.large \  
  --block-device-mappings '[{"DeviceName":"/dev/sdh","Ebs":  
{"VolumeSize":20,"VolumeType":"gp3"}]'
```

This command produces no output.

For more information, see [EBS](#) in the *Amazon EC2 Auto Scaling API Reference*.

For information about the syntax for quoting JSON-formatted parameter values, see [Using quotation marks with strings in the AWS CLI](#) in the *AWS Command Line Interface User Guide*.

Example 8: To create a launch configuration with a block device mapping for an instance store volume

This example creates a launch configuration with `ephemeral1` as an instance store volume with the device name `/dev/sdc`.

```
aws autoscaling create-launch-configuration \  
  --launch-configuration-name my-lc \  
  --image-id ami-04d5cc9b88example \  
  --instance-type m5.large \  
  --block-device-mappings '[{"DeviceName":"/dev/sdc","VirtualName":"ephemeral1"}]'
```

This command produces no output.

For more information, see [BlockDeviceMapping](#) in the *Amazon EC2 Auto Scaling API Reference*.

For information about the syntax for quoting JSON-formatted parameter values, see [Using quotation marks with strings in the AWS CLI](#) in the *AWS Command Line Interface User Guide*.

Example 9: To create a launch configuration and suppress a block device from attaching at launch time

This example creates a launch configuration that suppresses a block device specified by the block device mapping of the AMI (for example, `/dev/sdf`).

```
aws autoscaling create-launch-configuration \  
  --launch-configuration-name my-lc \  
  --image-id ami-04d5cc9b88example \  
  --instance-type m5.large \  
  --block-device-mappings '[{"DeviceName":"/dev/sdf","NoDevice":""}]'
```

This command produces no output.

For more information, see [BlockDeviceMapping](#) in the *Amazon EC2 Auto Scaling API Reference*.

For information about the syntax for quoting JSON-formatted parameter values, see [Using quotation marks with strings in the AWS CLI](#) in the *AWS Command Line Interface User Guide*.

- For API details, see [CreateLaunchConfiguration](#) in *AWS CLI Command Reference*.

create-or-update-tags

The following code example shows how to use `create-or-update-tags`.

AWS CLI

To create or update tags for an Auto Scaling group

This example adds two tags to the specified Auto Scaling group.

```
aws autoscaling create-or-update-tags \  
  --tags ResourceId=my-asg,ResourceType=auto-scaling-  
group,Key=Role,Value=WebServer,PropagateAtLaunch=true ResourceId=my-  
asg,ResourceType=auto-scaling-group,Key=Dept,Value=Research,PropagateAtLaunch=true
```

This command produces no output.

For more information, see [Tagging Auto Scaling groups and instances](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [CreateOrUpdateTags](#) in *AWS CLI Command Reference*.

delete-auto-scaling-group

The following code example shows how to use delete-auto-scaling-group.

AWS CLI

Example 1: To delete the specified Auto Scaling group

This example deletes the specified Auto Scaling group.

```
aws autoscaling delete-auto-scaling-group \  
  --auto-scaling-group-name my-asg
```

This command produces no output.

For more information, see [Deleting your Auto Scaling infrastructure](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 2: To force delete the specified Auto Scaling group

To delete the Auto Scaling group without waiting for the instances in the group to terminate, use the --force-delete option.

```
aws autoscaling delete-auto-scaling-group \  
  --force-delete
```

```
--auto-scaling-group-name my-asg \  
--force-delete
```

This command produces no output.

For more information, see [Deleting your Auto Scaling infrastructure](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DeleteAutoScalingGroup](#) in *AWS CLI Command Reference*.

delete-launch-configuration

The following code example shows how to use `delete-launch-configuration`.

AWS CLI

To delete a launch configuration

This example deletes the specified launch configuration.

```
aws autoscaling delete-launch-configuration \  
--launch-configuration-name my-launch-config
```

This command produces no output.

For more information, see [Deleting your Auto Scaling infrastructure](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DeleteLaunchConfiguration](#) in *AWS CLI Command Reference*.

delete-lifecycle-hook

The following code example shows how to use `delete-lifecycle-hook`.

AWS CLI

To delete a lifecycle hook

This example deletes the specified lifecycle hook.

```
aws autoscaling delete-lifecycle-hook \  

```

```
--lifecycle-hook-name my-lifecycle-hook \  
--auto-scaling-group-name my-asg
```

This command produces no output.

- For API details, see [DeleteLifecycleHook](#) in *AWS CLI Command Reference*.

delete-notification-configuration

The following code example shows how to use delete-notification-configuration.

AWS CLI

To delete an Auto Scaling notification

This example deletes the specified notification from the specified Auto Scaling group.

```
aws autoscaling delete-notification-configuration \  
  --auto-scaling-group-name my-asg \  
  --topic-arn arn:aws:sns:us-west-2:123456789012:my-sns-topic
```

This command produces no output.

For more information, see [Delete the notification configuration](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DeleteNotificationConfiguration](#) in *AWS CLI Command Reference*.

delete-policy

The following code example shows how to use delete-policy.

AWS CLI

To delete a scaling policy

This example deletes the specified scaling policy.

```
aws autoscaling delete-policy \  
  --auto-scaling-group-name my-asg \  
  --policy-name alb1000-target-tracking-scaling-policy
```

This command produces no output.

- For API details, see [DeletePolicy](#) in *AWS CLI Command Reference*.

delete-scheduled-action

The following code example shows how to use `delete-scheduled-action`.

AWS CLI

To delete a scheduled action from an Auto Scaling group

This example deletes the specified scheduled action from the specified Auto Scaling group.

```
aws autoscaling delete-scheduled-action \  
  --auto-scaling-group-name my-asg \  
  --scheduled-action-name my-scheduled-action
```

This command produces no output.

- For API details, see [DeleteScheduledAction](#) in *AWS CLI Command Reference*.

delete-tags

The following code example shows how to use `delete-tags`.

AWS CLI

To delete a tag from an Auto Scaling group

This example deletes the specified tag from the specified Auto Scaling group.

```
aws autoscaling delete-tags \  
  --tags ResourceId=my-asg,ResourceType=auto-scaling-group,Key=Dept,Value=Research
```

This command produces no output.

For more information, see [Tagging Auto Scaling groups and instances](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DeleteTags](#) in *AWS CLI Command Reference*.

delete-warm-pool

The following code example shows how to use `delete-warm-pool`.

AWS CLI

Example 1: To delete a warm pool

The following example deletes the warm pool for the specified Auto Scaling group.

```
aws autoscaling delete-warm-pool \  
  --auto-scaling-group-name my-asg
```

This command produces no output.

For more information, see [Warm pools for Amazon EC2 Auto Scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 2: To force delete a warm pool

To delete the warm pool without waiting for its instances to terminate, use the `--force-delete` option.

```
aws autoscaling delete-warm-pool \  
  --auto-scaling-group-name my-asg \  
  --force-delete
```

This command produces no output.

For more information, see [Warm pools for Amazon EC2 Auto Scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DeleteWarmPool](#) in *AWS CLI Command Reference*.

describe-account-limits

The following code example shows how to use `describe-account-limits`.

AWS CLI

To describe your Amazon EC2 Auto Scaling account limits

This example describes the Amazon EC2 Auto Scaling limits for your AWS account.

```
aws autoscaling describe-account-limits
```

Output:

```
{
  "NumberOfLaunchConfigurations": 5,
  "MaxNumberOfLaunchConfigurations": 100,
  "NumberOfAutoScalingGroups": 3,
  "MaxNumberOfAutoScalingGroups": 20
}
```

For more information, see [Amazon EC2 Auto Scaling service quotas](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DescribeAccountLimits](#) in *AWS CLI Command Reference*.

describe-adjustment-types

The following code example shows how to use `describe-adjustment-types`.

AWS CLI

To describe the available scaling adjustment types

This example describes the available adjustment types.

```
aws autoscaling describe-adjustment-types
```

Output:

```
{
  "AdjustmentTypes": [
    {
      "AdjustmentType": "ChangeInCapacity"
    },
    {
      "AdjustmentType": "ExactCapacity"
    },
    {
```



```

        "AdjustmentType": "PercentChangeInCapacity"
      }
    ]
  }

```

For more information, see [Scaling adjustment types](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DescribeAdjustmentTypes](#) in *AWS CLI Command Reference*.

describe-auto-scaling-groups

The following code example shows how to use describe-auto-scaling-groups.

AWS CLI

Example 1: To describe the specified Auto Scaling group

This example describes the specified Auto Scaling group.

```

aws autoscaling describe-auto-scaling-groups \
  --auto-scaling-group-name my-asg

```

Output:

```

{
  "AutoScalingGroups": [
    {
      "AutoScalingGroupName": "my-asg",
      "AutoScalingGroupARN": "arn:aws:autoscaling:us-west-2:123456789012:autoScalingGroup:930d940e-891e-4781-a11a-7b0acd480f03:autoScalingGroupName/my-asg",
      "LaunchTemplate": {
        "LaunchTemplateName": "my-launch-template",
        "Version": "1",
        "LaunchTemplateId": "lt-1234567890abcde12"
      },
      "MinSize": 0,
      "MaxSize": 1,
      "DesiredCapacity": 1,
      "DefaultCooldown": 300,
      "AvailabilityZones": [
        "us-west-2a",

```

```

        "us-west-2b",
        "us-west-2c"
    ],
    "LoadBalancerNames": [],
    "TargetGroupARNs": [],
    "HealthCheckType": "EC2",
    "HealthCheckGracePeriod": 0,
    "Instances": [
        {
            "InstanceId": "i-06905f55584de02da",
            "InstanceType": "t2.micro",
            "AvailabilityZone": "us-west-2a",
            "HealthStatus": "Healthy",
            "LifecycleState": "InService",
            "ProtectedFromScaleIn": false,
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-1234567890abcde12"
            }
        }
    ],
    "CreatedTime": "2023-10-28T02:39:22.152Z",
    "SuspendedProcesses": [],
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
    "EnabledMetrics": [],
    "Tags": [],
    "TerminationPolicies": [
        "Default"
    ],
    "NewInstancesProtectedFromScaleIn": false,
    "ServiceLinkedRoleARN": "arn",
    "TrafficSources": []
    }
]
}

```

Example 2: To describe the first 100 specified Auto Scaling group

This example describes the specified Auto Scaling groups. It allows you to specify up to 100 group names.

```
aws autoscaling describe-auto-scaling-groups \
```

```
--max-items 100 \  
--auto-scaling-group-name "group1" "group2" "group3" "group4"
```

See example 1 for sample output.

Example 3: To describe an Auto Scaling group in the specified region

This example describes the Auto Scaling groups in the specified region, up to a maximum of 75 groups.

```
aws autoscaling describe-auto-scaling-groups \  
  --max-items 75 \  
  --region us-east-1
```

See example 1 for sample output.

Example 4: To describe the specified number of Auto Scaling group

To return a specific number of Auto Scaling groups, use the `--max-items` option.

```
aws autoscaling describe-auto-scaling-groups \  
  --max-items 1
```

See example 1 for sample output.

If the output includes a `NextToken` field, there are more groups. To get the additional groups, use the value of this field with the `--starting-token` option in a subsequent call as follows.

```
aws autoscaling describe-auto-scaling-groups \  
  --starting-token Z3M3LMPEXAMPLE
```

See example 1 for sample output.

Example 5: To describe Auto Scaling groups that use launch configurations

This example uses the `--query` option to describe Auto Scaling groups that use launch configurations.

```
aws autoscaling describe-auto-scaling-groups \  
  --query 'AutoScalingGroups[?LaunchConfigurationName!=`null`]'
```

Output:

```
[
  {
    "AutoScalingGroupName": "my-asg",
    "AutoScalingGroupARN": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:930d940e-891e-4781-
a11a-7b0acd480f03:autoScalingGroupName/my-asg",
    "LaunchConfigurationName": "my-lc",
    "MinSize": 0,
    "MaxSize": 1,
    "DesiredCapacity": 1,
    "DefaultCooldown": 300,
    "AvailabilityZones": [
      "us-west-2a",
      "us-west-2b",
      "us-west-2c"
    ],
    "LoadBalancerNames": [],
    "TargetGroupARNs": [],
    "HealthCheckType": "EC2",
    "HealthCheckGracePeriod": 0,
    "Instances": [
      {
        "InstanceId": "i-088c57934a6449037",
        "InstanceType": "t2.micro",
        "AvailabilityZone": "us-west-2c",
        "HealthStatus": "Healthy",
        "LifecycleState": "InService",
        "LaunchConfigurationName": "my-lc",
        "ProtectedFromScaleIn": false
      }
    ],
    "CreatedTime": "2023-10-28T02:39:22.152Z",
    "SuspendedProcesses": [],
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
    "EnabledMetrics": [],
    "Tags": [],
    "TerminationPolicies": [
      "Default"
    ],
    "NewInstancesProtectedFromScaleIn": false,
    "ServiceLinkedRoleARN": "arn",
    "TrafficSources": []
  }
]
```

```
    }  
  ]
```

For more information, see [Filter AWS CLI output](#) in the *AWS Command Line Interface User Guide*.

- For API details, see [DescribeAutoScalingGroups](#) in *AWS CLI Command Reference*.

describe-auto-scaling-instances

The following code example shows how to use `describe-auto-scaling-instances`.

AWS CLI

Example 1: To describe one or more instances

This example describes the specified instance.

```
aws autoscaling describe-auto-scaling-instances \  
  --instance-ids i-06905f55584de02da
```

Output:

```
{  
  "AutoScalingInstances": [  
    {  
      "InstanceId": "i-06905f55584de02da",  
      "InstanceType": "t2.micro",  
      "AutoScalingGroupName": "my-asg",  
      "AvailabilityZone": "us-west-2b",  
      "LifecycleState": "InService",  
      "HealthStatus": "HEALTHY",  
      "ProtectedFromScaleIn": false,  
      "LaunchTemplate": {  
        "LaunchTemplateId": "lt-1234567890abcde12",  
        "LaunchTemplateName": "my-launch-template",  
        "Version": "1"  
      }  
    }  
  ]  
}
```

Example 2: To describe one or more instances

This example uses the `--max-items` option to specify how many instances to return with this call.

```
aws autoscaling describe-auto-scaling-instances \  
  --max-items 1
```

If the output includes a `NextToken` field, there are more instances. To get the additional instances, use the value of this field with the `--starting-token` option in a subsequent call as follows.

```
aws autoscaling describe-auto-scaling-instances \  
  --starting-token Z3M3LMPEXAMPLE
```

See example 1 for sample output.

Example 3: To describe instances that use launch configurations

This example uses the `--query` option to describe instances that use launch configurations.

```
aws autoscaling describe-auto-scaling-instances \  
  --query 'AutoScalingInstances[?LaunchConfigurationName!=`null`]'
```

Output:

```
[  
  {  
    "InstanceId": "i-088c57934a6449037",  
    "InstanceType": "t2.micro",  
    "AutoScalingGroupName": "my-asg",  
    "AvailabilityZone": "us-west-2c",  
    "LifecycleState": "InService",  
    "HealthStatus": "HEALTHY",  
    "LaunchConfigurationName": "my-lc",  
    "ProtectedFromScaleIn": false  
  }  
]
```

For more information, see [Filter AWS CLI output](#) in the *AWS Command Line Interface User Guide*.

- For API details, see [DescribeAutoScalingInstances](#) in *AWS CLI Command Reference*.

describe-auto-scaling-notification-types

The following code example shows how to use `describe-auto-scaling-notification-types`.

AWS CLI

To describe the available notification types

This example describes the available notification types.

```
aws autoscaling describe-auto-scaling-notification-types
```

Output:

```
{
  "AutoScalingNotificationTypes": [
    "autoscaling:EC2_INSTANCE_LAUNCH",
    "autoscaling:EC2_INSTANCE_LAUNCH_ERROR",
    "autoscaling:EC2_INSTANCE_TERMINATE",
    "autoscaling:EC2_INSTANCE_TERMINATE_ERROR",
    "autoscaling:TEST_NOTIFICATION"
  ]
}
```

For more information, see [Getting Amazon SNS notifications when your Auto Scaling group scales](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DescribeAutoScalingNotificationTypes](#) in *AWS CLI Command Reference*.

describe-instance-refreshes

The following code example shows how to use `describe-instance-refreshes`.

AWS CLI

To describe instance refreshes

The following `describe-instance-refreshes` example returns a description of all instance refresh requests for the specified Auto Scaling group, including the status message and (if available) the status reason.

```
aws autoscaling describe-instance-refreshes \
  --auto-scaling-group-name my-asg
```

Output:

```
{
  "InstanceRefreshes": [
    {
      "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b",
      "AutoScalingGroupName": "my-asg",
      "Status": "InProgress",
      "StatusReason": "Waiting for instances to warm up before continuing. For
example: 0e69cc3f05f825f4f is warming up.",
      "EndTime": "2023-03-23T16:42:55Z",
      "PercentageComplete": 0,
      "InstancesToUpdate": 0,
      "Preferences": {
        "MinHealthyPercentage": 100,
        "InstanceWarmup": 300,
        "CheckpointPercentages": [
          50
        ],
        "CheckpointDelay": 3600,
        "SkipMatching": false,
        "AutoRollback": true,
        "ScaleInProtectedInstances": "Ignore",
        "StandbyInstances": "Ignore"
      }
    },
    {
      "InstanceRefreshId": "dd7728d0-5bc4-4575-96a3-1b2c52bf8bb1",
      "AutoScalingGroupName": "my-asg",
      "Status": "Successful",
      "EndTime": "2022-06-02T16:53:37Z",
      "PercentageComplete": 100,
      "InstancesToUpdate": 0,
      "Preferences": {
        "MinHealthyPercentage": 90,
        "InstanceWarmup": 300,
        "SkipMatching": true,
        "AutoRollback": true,
        "ScaleInProtectedInstances": "Ignore",
        "StandbyInstances": "Ignore"
      }
    }
  ]
}
```



```

    }
  }
]
}

```

For more information, see [Check the status of an instance refresh](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DescribeInstanceRefreshes](#) in *AWS CLI Command Reference*.

describe-launch-configurations

The following code example shows how to use `describe-launch-configurations`.

AWS CLI

Example 1: To describe the specified launch configuration

This example describes the specified launch configuration.

```

aws autoscaling describe-launch-configurations \
  --launch-configuration-names my-launch-config

```

Output:

```

{
  "LaunchConfigurations": [
    {
      "LaunchConfigurationName": "my-launch-config",
      "LaunchConfigurationARN": "arn:aws:autoscaling:us-
west-2:123456789012:launchConfiguration:98d3b196-4cf9-4e88-8ca1-8547c24ced8b:launchConfigura
my-launch-config",
      "ImageId": "ami-0528a5175983e7f28",
      "KeyName": "my-key-pair-uswest2",
      "SecurityGroups": [
        "sg-05eaec502fcdadc2e"
      ],
      "ClassicLinkVPCSecurityGroups": [],
      "UserData": "",
      "InstanceType": "t2.micro",
      "KernelId": "",
      "RamdiskId": ""
    }
  ]
}

```

```

    "BlockDeviceMappings": [
      {
        "DeviceName": "/dev/xvda",
        "Ebs": {
          "SnapshotId": "snap-06c1606ba5ca274b1",
          "VolumeSize": 8,
          "VolumeType": "gp2",
          "DeleteOnTermination": true,
          "Encrypted": false
        }
      }
    ],
    "InstanceMonitoring": {
      "Enabled": true
    },
    "CreatedTime": "2020-10-28T02:39:22.321Z",
    "EbsOptimized": false,
    "AssociatePublicIpAddress": true,
    "MetadataOptions": {
      "HttpTokens": "required",
      "HttpPutResponseHopLimit": 1,
      "HttpEndpoint": "disabled"
    }
  }
]
}

```

Example 2: To describe a specified number of launch configurations

To return a specific number of launch configurations, use the `--max-items` option.

```

aws autoscaling describe-launch-configurations \
  --max-items 1

```

If the output includes a `NextToken` field, there are more launch configurations. To get the additional launch configurations, use the value of this field with the `--starting-token` option in a subsequent call as follows.

```

aws autoscaling describe-launch-configurations \
  --starting-token Z3M3LMPEXAMPLE

```

- For API details, see [DescribeLaunchConfigurations](#) in *AWS CLI Command Reference*.

describe-lifecycle-hook-types

The following code example shows how to use `describe-lifecycle-hook-types`.

AWS CLI

To describe the available lifecycle hook types

This example describes the available lifecycle hook types.

```
aws autoscaling describe-lifecycle-hook-types
```

Output:

```
{
  "LifecycleHookTypes": [
    "autoscaling:EC2_INSTANCE_LAUNCHING",
    "autoscaling:EC2_INSTANCE_TERMINATING"
  ]
}
```

- For API details, see [DescribeLifecycleHookTypes](#) in *AWS CLI Command Reference*.

describe-lifecycle-hooks

The following code example shows how to use `describe-lifecycle-hooks`.

AWS CLI

To describe your lifecycle hooks

This example describes the lifecycle hooks for the specified Auto Scaling group.

```
aws autoscaling describe-lifecycle-hooks \
  --auto-scaling-group-name my-asg
```

Output:

```
{
  "LifecycleHooks": [
    {
```

```

        "GlobalTimeout": 3000,
        "HeartbeatTimeout": 30,
        "AutoScalingGroupName": "my-asg",
        "LifecycleHookName": "my-launch-hook",
        "DefaultResult": "ABANDON",
        "LifecycleTransition": "autoscaling:EC2_INSTANCE_LAUNCHING"
    },
    {
        "GlobalTimeout": 6000,
        "HeartbeatTimeout": 60,
        "AutoScalingGroupName": "my-asg",
        "LifecycleHookName": "my-termination-hook",
        "DefaultResult": "CONTINUE",
        "LifecycleTransition": "autoscaling:EC2_INSTANCE_TERMINATING"
    }
]
}

```

- For API details, see [DescribeLifecycleHooks](#) in *AWS CLI Command Reference*.

describe-load-balancer-target-groups

The following code example shows how to use `describe-load-balancer-target-groups`.

AWS CLI

To describe the load balancer target groups for an Auto Scaling group

This example describes the load balancer target groups attached to the specified Auto Scaling group.

```
aws autoscaling describe-load-balancer-target-groups \
  --auto-scaling-group-name my-asg
```

Output:

```
{
  "LoadBalancerTargetGroups": [
    {
      "LoadBalancerTargetGroupARN": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067",
      "State": "Added"
    }
  ]
}
```

```
    }  
  ]  
}
```

- For API details, see [DescribeLoadBalancerTargetGroups](#) in *AWS CLI Command Reference*.

describe-load-balancers

The following code example shows how to use `describe-load-balancers`.

AWS CLI

To describe the Classic Load Balancers for an Auto Scaling group

This example describes the Classic Load Balancers for the specified Auto Scaling group.

```
aws autoscaling describe-load-balancers \  
  --auto-scaling-group-name my-asg
```

Output:

```
{  
  "LoadBalancers": [  
    {  
      "State": "Added",  
      "LoadBalancerName": "my-load-balancer"  
    }  
  ]  
}
```

- For API details, see [DescribeLoadBalancers](#) in *AWS CLI Command Reference*.

describe-metric-collection-types

The following code example shows how to use `describe-metric-collection-types`.

AWS CLI

To describe the available metric collection types

This example describes the available metric collection types.

```
aws autoscaling describe-metric-collection-types
```

Output:

```
{
  "Metrics": [
    {
      "Metric": "GroupMinSize"
    },
    {
      "Metric": "GroupMaxSize"
    },
    {
      "Metric": "GroupDesiredCapacity"
    },
    {
      "Metric": "GroupInServiceInstances"
    },
    {
      "Metric": "GroupInServiceCapacity"
    },
    {
      "Metric": "GroupPendingInstances"
    },
    {
      "Metric": "GroupPendingCapacity"
    },
    {
      "Metric": "GroupTerminatingInstances"
    },
    {
      "Metric": "GroupTerminatingCapacity"
    },
    {
      "Metric": "GroupStandbyInstances"
    },
    {
      "Metric": "GroupStandbyCapacity"
    },
    {
      "Metric": "GroupTotalInstances"
    },
    {
```

```

        "Metric": "GroupTotalCapacity"
      }
    ],
    "Granularities": [
      {
        "Granularity": "1Minute"
      }
    ]
  }
}

```

For more information, see [Auto Scaling group metrics](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DescribeMetricCollectionTypes](#) in *AWS CLI Command Reference*.

describe-notification-configurations

The following code example shows how to use `describe-notification-configurations`.

AWS CLI

Example 1: To describe the notification configurations of a specified group

This example describes the notification configurations for the specified Auto Scaling group.

```

aws autoscaling describe-notification-configurations \
  --auto-scaling-group-name my-asg

```

Output:

```

{
  "NotificationConfigurations": [
    {
      "AutoScalingGroupName": "my-asg",
      "NotificationType": "autoscaling:TEST_NOTIFICATION",
      "TopicARN": "arn:aws:sns:us-west-2:123456789012:my-sns-topic-2"
    },
    {
      "AutoScalingGroupName": "my-asg",
      "NotificationType": "autoscaling:TEST_NOTIFICATION",
      "TopicARN": "arn:aws:sns:us-west-2:123456789012:my-sns-topic"
    }
  ]
}

```

```
}
```

For more information, see [Getting Amazon SNS notifications when your Auto Scaling group scales](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 1: To describe a specified number of notification configurations

To return a specific number of notification configurations, use the `max-items` parameter.

```
aws autoscaling describe-notification-configurations \  
  --auto-scaling-group-name my-auto-scaling-group \  
  --max-items 1
```

Output:

```
{  
  "NotificationConfigurations": [  
    {  
      "AutoScalingGroupName": "my-asg",  
      "NotificationType": "autoscaling:TEST_NOTIFICATION",  
      "TopicARN": "arn:aws:sns:us-west-2:123456789012:my-sns-topic-2"  
    },  
    {  
      "AutoScalingGroupName": "my-asg",  
      "NotificationType": "autoscaling:TEST_NOTIFICATION",  
      "TopicARN": "arn:aws:sns:us-west-2:123456789012:my-sns-topic"  
    }  
  ]  
}
```

If the output includes a `NextToken` field, there are more notification configurations. To get the additional notification configurations, use the value of this field with the `starting-token` parameter in a subsequent call as follows.

```
aws autoscaling describe-notification-configurations \  
  --auto-scaling-group-name my-asg \  
  --starting-token Z3M3LMPEXAMPLE
```

For more information, see [Getting Amazon SNS notifications when your Auto Scaling group scales](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DescribeNotificationConfigurations](#) in *AWS CLI Command Reference*.

describe-policies

The following code example shows how to use describe-policies.

AWS CLI

Example 1: To describe the scaling policies of a specified group

This example describes the scaling policies for the specified Auto Scaling group.

```
aws autoscaling describe-policies \  
  --auto-scaling-group-name my-asg
```

Output:

```
{  
  "ScalingPolicies": [  
    {  
      "AutoScalingGroupName": "my-asg",  
      "PolicyName": "alb1000-target-tracking-scaling-policy",  
      "PolicyARN": "arn:aws:autoscaling:us-  
west-2:123456789012:scalingPolicy:3065d9c8-9969-4bec-  
bb6a-3fbe5550fde6:autoScalingGroupName/my-asg:policyName/alb1000-target-tracking-  
scaling-policy",  
      "PolicyType": "TargetTrackingScaling",  
      "StepAdjustments": [],  
      "Alarms": [  
        {  
          "AlarmName": "TargetTracking-my-asg-  
AlarmHigh-924887a9-12d7-4e01-8686-6f844d13a196",  
          "AlarmARN": "arn:aws:cloudwatch:us-  
west-2:123456789012:alarm:TargetTracking-my-asg-  
AlarmHigh-924887a9-12d7-4e01-8686-6f844d13a196"  
        },  
        {  
          "AlarmName": "TargetTracking-my-asg-AlarmLow-f96f899d-b8e7-4d09-  
a010-c1aaa35da296",  
          "AlarmARN": "arn:aws:cloudwatch:us-  
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmLow-f96f899d-b8e7-4d09-a010-  
c1aaa35da296"  
        }  
      ],  
      "TargetTrackingConfiguration": {  
        "PredefinedMetricSpecification": {
```

```

        "PredefinedMetricType": "ALBRequestCountPerTarget",
        "ResourceLabel": "app/my-alb/778d41231b141a0f/targetgroup/my-
alb-target-group/943f017f100becff"
    },
    "TargetValue": 1000.0,
    "DisableScaleIn": false
},
"Enabled": true
},
{
    "AutoScalingGroupName": "my-asg",
    "PolicyName": "cpu40-target-tracking-scaling-policy",
    "PolicyARN": "arn:aws:autoscaling:us-
west-2:123456789012:scalingPolicy:5fd26f71-39d4-4690-82a9-
b8515c45cdde:autoScalingGroupName/my-asg:policyName/cpu40-target-tracking-scaling-
policy",
    "PolicyType": "TargetTrackingScaling",
    "StepAdjustments": [],
    "Alarms": [
        {
            "AlarmName": "TargetTracking-my-asg-
AlarmHigh-139f9789-37b9-42ad-bea5-b5b147d7f473",
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmHigh-139f9789-37b9-42ad-bea5-
b5b147d7f473"
        },
        {
            "AlarmName": "TargetTracking-my-asg-AlarmLow-bd681c67-
fc18-4c56-8468-fb8e413009c9",
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmLow-bd681c67-fc18-4c56-8468-
fb8e413009c9"
        }
    ],
    "TargetTrackingConfiguration": {
        "PredefinedMetricSpecification": {
            "PredefinedMetricType": "ASGAverageCPUUtilization"
        },
        "TargetValue": 40.0,
        "DisableScaleIn": false
    },
    "Enabled": true
}
]

```

```
}
```

For more information, see [Dynamic scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 2: To describe the scaling policies of a specified name

To return specific scaling policies, use the `--policy-names` option.

```
aws autoscaling describe-policies \  
  --auto-scaling-group-name my-asg \  
  --policy-names cpu40-target-tracking-scaling-policy
```

See example 1 for sample output.

For more information, see [Dynamic scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 3: To describe a number of scaling policies

To return a specific number of policies, use the `--max-items` option.

```
aws autoscaling describe-policies \  
  --auto-scaling-group-name my-asg \  
  --max-items 1
```

See example 1 for sample output.

If the output includes a `NextToken` field, use the value of this field with the `--starting-token` option in a subsequent call to get the additional policies.

```
aws autoscaling describe-policies --auto-scaling-group-name my-asg --starting-token  
Z3M3LMPEXAMPLE
```

For more information, see [Dynamic scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DescribePolicies](#) in *AWS CLI Command Reference*.

describe-scaling-activities

The following code example shows how to use `describe-scaling-activities`.

AWS CLI

Example 1: To describe scaling activities for the specified group

This example describes the scaling activities for the specified Auto Scaling group.

```
aws autoscaling describe-scaling-activities \  
  --auto-scaling-group-name my-asg
```

Output:

```
{  
  "Activities": [  
    {  
      "ActivityId": "f9f2d65b-f1f2-43e7-b46d-d86756459699",  
      "Description": "Launching a new EC2 instance: i-0d44425630326060f",  
      "AutoScalingGroupName": "my-asg",  
      "Cause": "At 2020-10-30T19:35:51Z a user request update of  
AutoScalingGroup constraints to min: 0, max: 16, desired: 16 changing the desired  
capacity from 0 to 16. At 2020-10-30T19:36:07Z an instance was started in response  
to a difference between desired and actual capacity, increasing the capacity from 0  
to 16.",  
      "StartTime": "2020-10-30T19:36:09.766Z",  
      "EndTime": "2020-10-30T19:36:41Z",  
      "StatusCode": "Successful",  
      "Progress": 100,  
      "Details": "{\"Subnet ID\": \"subnet-5ea0c127\", \"Availability Zone\":  
\"us-west-2b\"}"  
    }  
  ]  
}
```

For more information, see [Verify a scaling activity for an Auto Scaling group](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 2: To describe the scaling activities for a deleted group

To describe scaling activities after the Auto Scaling group has been deleted, add the `--include-deleted-groups` option.

```
aws autoscaling describe-scaling-activities \  
  --auto-scaling-group-name my-asg \  
  --include-deleted-groups
```

```
--include-deleted-groups
```

Output:

```
{
  "Activities": [
    {
      "ActivityId": "e1f5de0e-f93e-1417-34ac-092a76fba220",
      "Description": "Launching a new EC2 instance. Status Reason: Your Spot
request price of 0.001 is lower than the minimum required Spot request fulfillment
price of 0.0031. Launching EC2 instance failed.",
      "AutoScalingGroupName": "my-asg",
      "Cause": "At 2021-01-13T20:47:24Z a user request update of
AutoScalingGroup constraints to min: 1, max: 5, desired: 3 changing the desired
capacity from 0 to 3. At 2021-01-13T20:47:27Z an instance was started in response
to a difference between desired and actual capacity, increasing the capacity from 0
to 3.",
      "StartTime": "2021-01-13T20:47:30.094Z",
      "EndTime": "2021-01-13T20:47:30Z",
      "StatusCode": "Failed",
      "StatusMessage": "Your Spot request price of 0.001 is lower than the
minimum required Spot request fulfillment price of 0.0031. Launching EC2 instance
failed.",
      "Progress": 100,
      "Details": "{\"Subnet ID\": \"subnet-5ea0c127\", \"Availability Zone\":
\\\"us-west-2b\\\"}\",
      "AutoScalingGroupState": "Deleted",
      "AutoScalingGroupARN": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:283179a2-
f3ce-423d-93f6-66bb518232f7:autoScalingGroupName/my-asg"
    }
  ]
}
```

For more information, see [Troubleshoot Amazon EC2 Auto Scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 3: To describe a specified number of scaling activities

To return a specific number of activities, use the `--max-items` option.

```
aws autoscaling describe-scaling-activities \
```

```
--max-items 1
```

Output:

```
{
  "Activities": [
    {
      "ActivityId": "f9f2d65b-f1f2-43e7-b46d-d86756459699",
      "Description": "Launching a new EC2 instance: i-0d44425630326060f",
      "AutoScalingGroupName": "my-asg",
      "Cause": "At 2020-10-30T19:35:51Z a user request update of
AutoScalingGroup constraints to min: 0, max: 16, desired: 16 changing the desired
capacity from 0 to 16. At 2020-10-30T19:36:07Z an instance was started in response
to a difference between desired and actual capacity, increasing the capacity from 0
to 16.",
      "StartTime": "2020-10-30T19:36:09.766Z",
      "EndTime": "2020-10-30T19:36:41Z",
      "StatusCode": "Successful",
      "Progress": 100,
      "Details": "{\"Subnet ID\": \"subnet-5ea0c127\", \"Availability Zone\":
\\\"us-west-2b\\\"}"
    }
  ]
}
```

If the output includes a `NextToken` field, there are more activities. To get the additional activities, use the value of this field with the `--starting-token` option in a subsequent call as follows.

```
aws autoscaling describe-scaling-activities \
  --starting-token Z3M3LMPEXAMPLE
```

For more information, see [Verify a scaling activity for an Auto Scaling group](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DescribeScalingActivities](#) in *AWS CLI Command Reference*.

describe-scaling-process-types

The following code example shows how to use `describe-scaling-process-types`.

AWS CLI

To describe the available process types

This example describes the available process types.

```
aws autoscaling describe-scaling-process-types
```

Output:

```
{
  "Processes": [
    {
      "ProcessName": "AZRebalance"
    },
    {
      "ProcessName": "AddToLoadBalancer"
    },
    {
      "ProcessName": "AlarmNotification"
    },
    {
      "ProcessName": "HealthCheck"
    },
    {
      "ProcessName": "InstanceRefresh"
    },
    {
      "ProcessName": "Launch"
    },
    {
      "ProcessName": "ReplaceUnhealthy"
    },
    {
      "ProcessName": "ScheduledActions"
    },
    {
      "ProcessName": "Terminate"
    }
  ]
}
```

For more information, see [Suspending and resuming scaling processes](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DescribeScalingProcessTypes](#) in *AWS CLI Command Reference*.

describe-scheduled-actions

The following code example shows how to use `describe-scheduled-actions`.

AWS CLI

Example 1: To describe all scheduled actions

This example describes all your scheduled actions.

```
aws autoscaling describe-scheduled-actions
```

Output:

```
{
  "ScheduledUpdateGroupActions": [
    {
      "AutoScalingGroupName": "my-asg",
      "ScheduledActionName": "my-recurring-action",
      "Recurrence": "30 0 1 1,6,12 *",
      "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-action",
      "StartTime": "2023-12-01T04:00:00Z",
      "Time": "2023-12-01T04:00:00Z",
      "MinSize": 1,
      "MaxSize": 6,
      "DesiredCapacity": 4,
      "TimeZone": "America/New_York"
    }
  ]
}
```

For more information, see [Scheduled scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 2: To describe scheduled actions for the specified group

To describe the scheduled actions for a specific Auto Scaling group, use the `--auto-scaling-group-name` option.

```
aws autoscaling describe-scheduled-actions \  
  --auto-scaling-group-name my-asg
```

Output:

```
{  
  "ScheduledUpdateGroupActions": [  
    {  
      "AutoScalingGroupName": "my-asg",  
      "ScheduledActionName": "my-recurring-action",  
      "Recurrence": "30 0 1 1,6,12 *",  
      "ScheduledActionARN": "arn:aws:autoscaling:us-  
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-  
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-action",  
      "StartTime": "2023-12-01T04:00:00Z",  
      "Time": "2023-12-01T04:00:00Z",  
      "MinSize": 1,  
      "MaxSize": 6,  
      "DesiredCapacity": 4,  
      "TimeZone": "America/New_York"  
    }  
  ]  
}
```

For more information, see [Scheduled scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 3: To describe the specified scheduled action

To describe a specific scheduled action, use the `--scheduled-action-names` option.

```
aws autoscaling describe-scheduled-actions \  
  --scheduled-action-names my-recurring-action
```

Output:

```
{  
  "ScheduledUpdateGroupActions": [  
    {
```

```

        "AutoScalingGroupName": "my-asg",
        "ScheduledActionName": "my-recurring-action",
        "Recurrence": "30 0 1 1,6,12 *",
        "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-action",
        "StartTime": "2023-12-01T04:00:00Z",
        "Time": "2023-12-01T04:00:00Z",
        "MinSize": 1,
        "MaxSize": 6,
        "DesiredCapacity": 4,
        "TimeZone": "America/New_York"
    }
]
}

```

For more information, see [Scheduled scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 4: To describe scheduled actions with a specified start time

To describe the scheduled actions that start at a specific time, use the `--start-time` option.

```

aws autoscaling describe-scheduled-actions \
  --start-time "2023-12-01T04:00:00Z"

```

Output:

```

{
  "ScheduledUpdateGroupActions": [
    {
      "AutoScalingGroupName": "my-asg",
      "ScheduledActionName": "my-recurring-action",
      "Recurrence": "30 0 1 1,6,12 *",
      "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-action",
      "StartTime": "2023-12-01T04:00:00Z",
      "Time": "2023-12-01T04:00:00Z",
      "MinSize": 1,
      "MaxSize": 6,
      "DesiredCapacity": 4,
      "TimeZone": "America/New_York"
    }
  ]
}

```

```
]
}
```

For more information, see [Scheduled scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 5: To describe scheduled actions that end at a specified time

To describe the scheduled actions that end at a specific time, use the `--end-time` option.

```
aws autoscaling describe-scheduled-actions \
  --end-time "2023-12-01T04:00:00Z"
```

Output:

```
{
  "ScheduledUpdateGroupActions": [
    {
      "AutoScalingGroupName": "my-asg",
      "ScheduledActionName": "my-recurring-action",
      "Recurrence": "30 0 1 1,6,12 *",
      "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-action",
      "StartTime": "2023-12-01T04:00:00Z",
      "Time": "2023-12-01T04:00:00Z",
      "MinSize": 1,
      "MaxSize": 6,
      "DesiredCapacity": 4,
      "TimeZone": "America/New_York"
    }
  ]
}
```

For more information, see [Scheduled scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 6: To describe a specified number of scheduled actions

To return a specific number of scheduled actions, use the `--max-items` option.

```
aws autoscaling describe-scheduled-actions \
  --auto-scaling-group-name my-asg \
```

```
--max-items 1
```

Output:

```
{
  "ScheduledUpdateGroupActions": [
    {
      "AutoScalingGroupName": "my-asg",
      "ScheduledActionName": "my-recurring-action",
      "Recurrence": "30 0 1 1,6,12 *",
      "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-action",
      "StartTime": "2023-12-01T04:00:00Z",
      "Time": "2023-12-01T04:00:00Z",
      "MinSize": 1,
      "MaxSize": 6,
      "DesiredCapacity": 4,
      "TimeZone": "America/New_York"
    }
  ]
}
```

If the output includes a `NextToken` field, there are more scheduled actions. To get the additional scheduled actions, use the value of this field with the `--starting-token` option in a subsequent call as follows.

```
aws autoscaling describe-scheduled-actions \
  --auto-scaling-group-name my-asg \
  --starting-token Z3M3LMPEXAMPLE
```

For more information, see [Scheduled scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DescribeScheduledActions](#) in *AWS CLI Command Reference*.

describe-tags

The following code example shows how to use `describe-tags`.

AWS CLI

To describe all tags

This example describes all your tags.

```
aws autoscaling describe-tags
```

Output:

```
{
  "Tags": [
    {
      "ResourceType": "auto-scaling-group",
      "ResourceId": "my-asg",
      "PropagateAtLaunch": true,
      "Value": "Research",
      "Key": "Dept"
    },
    {
      "ResourceType": "auto-scaling-group",
      "ResourceId": "my-asg",
      "PropagateAtLaunch": true,
      "Value": "WebServer",
      "Key": "Role"
    }
  ]
}
```

For more information, see [Tagging Auto Scaling groups and instances](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 2: To describe tags for a specified group

To describe tags for a specific Auto Scaling group, use the `--filters` option.

```
aws autoscaling describe-tags --filters Name=auto-scaling-group,Values=my-asg
```

For more information, see [Tagging Auto Scaling groups and instances](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 3: To describe the specified number of tags

To return a specific number of tags, use the `--max-items` option.

```
aws autoscaling describe-tags \  
  --max-items 1
```

If the output includes a `NextToken` field, there are more tags. To get the additional tags, use the value of this field with the `--starting-token` option in a subsequent call as follows.

```
aws autoscaling describe-tags \  
  --filters Name=auto-scaling-group,Values=my-asg \  
  --starting-token Z3M3LMPEXAMPLE
```

For more information, see [Tagging Auto Scaling groups and instances](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DescribeTags](#) in *AWS CLI Command Reference*.

describe-termination-policy-types

The following code example shows how to use `describe-termination-policy-types`.

AWS CLI

To describe available termination policy types

This example describes the available termination policy types.

```
aws autoscaling describe-termination-policy-types
```

Output:

```
{  
  "TerminationPolicyTypes": [  
    "AllocationStrategy",  
    "ClosestToNextInstanceHour",  
    "Default",  
    "NewestInstance",  
    "OldestInstance",  
    "OldestLaunchConfiguration",  
    "OldestLaunchTemplate"  
  ]  
}
```

For more information, see [Controlling which Auto Scaling instances terminate during scale in](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DescribeTerminationPolicyTypes](#) in *AWS CLI Command Reference*.

describe-warm-pool

The following code example shows how to use `describe-warm-pool`.

AWS CLI

To describe a warm pool

This example describes the warm pool for the specified Auto Scaling group.

```
aws autoscaling describe-warm-pool \  
  --auto-scaling-group-name my-asg
```

Output:

```
{  
  "WarmPoolConfiguration": {  
    "MinSize": 2,  
    "PoolState": "Stopped"  
  },  
  "Instances": [  
    {  
      "InstanceId": "i-070a5bbc7e7f40dc5",  
      "InstanceType": "t2.micro",  
      "AvailabilityZone": "us-west-2c",  
      "LifecycleState": "Warmed:Pending",  
      "HealthStatus": "Healthy",  
      "LaunchTemplate": {  
        "LaunchTemplateId": "lt-00a731f6e9fa48610",  
        "LaunchTemplateName": "my-template-for-auto-scaling",  
        "Version": "6"  
      }  
    },  
    {  
      "InstanceId": "i-0b52f061814d3bd2d",  
      "InstanceType": "t2.micro",  
      "AvailabilityZone": "us-west-2b",  
      "LifecycleState": "Warmed:Pending",
```

```

    "HealthStatus": "Healthy",
    "LaunchTemplate": {
      "LaunchTemplateId": "lt-00a731f6e9fa48610",
      "LaunchTemplateName": "my-template-for-auto-scaling",
      "Version": "6"
    }
  ]
}

```

For more information, see [Warm pools for Amazon EC2 Auto Scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DescribeWarmPool](#) in *AWS CLI Command Reference*.

detach-instances

The following code example shows how to use `detach-instances`.

AWS CLI

To detach an instance from an Auto Scaling group

This example detaches the specified instance from the specified Auto Scaling group.

```

aws autoscaling detach-instances \
  --instance-ids i-030017cfa84b20135 \
  --auto-scaling-group-name my-asg \
  --should-decrement-desired-capacity

```

Output:

```

{
  "Activities": [
    {
      "ActivityId": "5091cb52-547a-47ce-a236-c9ccbc2cb2c9",
      "AutoScalingGroupName": "my-asg",
      "Description": "Detaching EC2 instance: i-030017cfa84b20135",
      "Cause": "At 2020-10-31T17:35:04Z instance i-030017cfa84b20135 was detached in response to a user request, shrinking the capacity from 2 to 1.",
      "StartTime": "2020-04-12T15:02:16.179Z",
      "StatusCode": "InProgress",
    }
  ]
}

```



```
        "Progress": 50,  
        "Details": "{\\"Subnet ID\\":\\"subnet-6194ea3b\\",\\"Availability Zone\\":  
\\\"us-west-2c\\\"}"  
      }  
    ]  
  }  
}
```

- For API details, see [DetachInstances](#) in *AWS CLI Command Reference*.

detach-load-balancer-target-groups

The following code example shows how to use `detach-load-balancer-target-groups`.

AWS CLI

To detach a load balancer target group from an Auto Scaling group

This example detaches the specified load balancer target group from the specified Auto Scaling group.

```
aws autoscaling detach-load-balancer-target-groups \  
  --auto-scaling-group-name my-asg \  
  --target-group-arns arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

This command produces no output

For more information, see [Attaching a load balancer to your Auto Scaling group](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DetachLoadBalancerTargetGroups](#) in *AWS CLI Command Reference*.

detach-load-balancers

The following code example shows how to use `detach-load-balancers`.

AWS CLI

To detach a Classic Load Balancer from an Auto Scaling group

This example detaches the specified Classic Load Balancer from the specified Auto Scaling group.

```
aws autoscaling detach-load-balancers \  
  --load-balancer-names my-load-balancer \  
  --auto-scaling-group-name my-asg
```

This command produces no output.

For more information, see [Attaching a load balancer to your Auto Scaling group](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DetachLoadBalancers](#) in *AWS CLI Command Reference*.

disable-metrics-collection

The following code example shows how to use `disable-metrics-collection`.

AWS CLI

To disable metrics collection for an Auto Scaling group

This example disables collection of the `GroupDesiredCapacity` metric for the specified Auto Scaling group.

```
aws autoscaling disable-metrics-collection \  
  --auto-scaling-group-name my-asg \  
  --metrics GroupDesiredCapacity
```

This command produces no output.

For more information, see [Monitoring CloudWatch metrics for your Auto Scaling groups and instances](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [DisableMetricsCollection](#) in *AWS CLI Command Reference*.

enable-metrics-collection

The following code example shows how to use `enable-metrics-collection`.

AWS CLI

Example 1: To enable metrics collection for an Auto Scaling group

This example enables data collection for the specified Auto Scaling group.

```
aws autoscaling enable-metrics-collection \  
  --auto-scaling-group-name my-asg \  
  --granularity "1Minute"
```

This command produces no output.

For more information, see [Monitoring CloudWatch metrics for your Auto Scaling groups and instances](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 2: To collect data for the specified metric for an Auto Scaling group

To collect data for a specific metric, use the `--metrics` option.

```
aws autoscaling enable-metrics-collection \  
  --auto-scaling-group-name my-asg \  
  --metrics GroupDesiredCapacity --granularity "1Minute"
```

This command produces no output.

For more information, see [Monitoring CloudWatch metrics for your Auto Scaling groups and instances](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [EnableMetricsCollection](#) in *AWS CLI Command Reference*.

enter-standby

The following code example shows how to use `enter-standby`.

AWS CLI

To move instances into standby mode

This example puts the specified instance into standby mode. This is useful for updating or troubleshooting an instance that is currently in service.

```
aws autoscaling enter-standby \  
  --instance-ids i-061c63c5eb45f0416 \  
  --auto-scaling-group-name my-asg \  
  --granularity "1Minute"
```

```
--should-decrement-desired-capacity
```

Output:

```
{
  "Activities": [
    {
      "ActivityId": "ffa056b4-6ed3-41ba-ae7c-249dfae6eba1",
      "AutoScalingGroupName": "my-asg",
      "Description": "Moving EC2 instance to Standby: i-061c63c5eb45f0416",
      "Cause": "At 2020-10-31T20:31:00Z instance i-061c63c5eb45f0416 was moved
to standby in response to a user request, shrinking the capacity from 1 to 0.",
      "StartTime": "2020-10-31T20:31:00.949Z",
      "StatusCode": "InProgress",
      "Progress": 50,
      "Details": "{\"Subnet ID\":\"subnet-6194ea3b\",\"Availability Zone\":
\\\"us-west-2c\\\"}"
    }
  ]
}
```

For more information, see [Amazon EC2 Auto Scaling instance lifecycle](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [EnterStandby](#) in *AWS CLI Command Reference*.

execute-policy

The following code example shows how to use `execute-policy`.

AWS CLI

To execute a scaling policy

This example executes the scaling policy named `my-step-scale-out-policy` for the specified Auto Scaling group.

```
aws autoscaling execute-policy \
  --auto-scaling-group-name my-asg \
  --policy-name my-step-scale-out-policy \
  --metric-value 95 \
```

```
--breach-threshold 80
```

This command produces no output.

For more information, see [Step and simple scaling policies](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [ExecutePolicy](#) in *AWS CLI Command Reference*.

exit-standby

The following code example shows how to use `exit-standby`.

AWS CLI

To move instances out of standby mode

This example moves the specified instance out of standby mode.

```
aws autoscaling exit-standby \  
  --instance-ids i-061c63c5eb45f0416 \  
  --auto-scaling-group-name my-asg
```

Output:

```
{  
  "Activities": [  
    {  
      "ActivityId": "142928e1-a2dc-453a-9b24-b85ad6735928",  
      "AutoScalingGroupName": "my-asg",  
      "Description": "Moving EC2 instance out of Standby:  
i-061c63c5eb45f0416",  
      "Cause": "At 2020-10-31T20:32:50Z instance i-061c63c5eb45f0416 was moved  
out of standby in response to a user request, increasing the capacity from 0 to  
1.",  
      "StartTime": "2020-10-31T20:32:50.222Z",  
      "StatusCode": "PreInService",  
      "Progress": 30,  
      "Details": "{\"Subnet ID\": \"subnet-6194ea3b\", \"Availability Zone\":  
\"us-west-2c\"}"  
    }  
  ]  
}
```

```
}
```

For more information, see [Temporarily removing instances from your Auto Scaling group](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [ExitStandby](#) in *AWS CLI Command Reference*.

put-lifecycle-hook

The following code example shows how to use `put-lifecycle-hook`.

AWS CLI

Example 1: To create a lifecycle hook

This example creates a lifecycle hook that will invoke on any newly launched instances, with a timeout of 4800 seconds. This is useful for keeping the instances in a wait state until the user data scripts have finished, or for invoking an AWS Lambda function using EventBridge.

```
aws autoscaling put-lifecycle-hook \  
  --auto-scaling-group-name my-asg \  
  --lifecycle-hook-name my-launch-hook \  
  --lifecycle-transition autoscaling:EC2_INSTANCE_LAUNCHING \  
  --heartbeat-timeout 4800
```

This command produces no output. If a lifecycle hook with the same name already exists, it will be overwritten by the new lifecycle hook.

For more information, see [Amazon EC2 Auto Scaling lifecycle hooks](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 2: To send an Amazon SNS email message to notify you of instance state transitions

This example creates a lifecycle hook with the Amazon SNS topic and IAM role to use to receive notification at instance launch.

```
aws autoscaling put-lifecycle-hook \  
  --auto-scaling-group-name my-asg \  
  --lifecycle-hook-name my-launch-hook \  
  --lifecycle-transition autoscaling:EC2_INSTANCE_LAUNCHING
```

```
--lifecycle-transition autoscaling:EC2_INSTANCE_LAUNCHING \  
--notification-target-arn arn:aws:sns:us-west-2:123456789012:my-sns-topic \  
--role-arn arn:aws:iam::123456789012:role/my-auto-scaling-role
```

This command produces no output.

For more information, see [Amazon EC2 Auto Scaling lifecycle hooks](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 3: To publish a message to an Amazon SQS queue

This example creates a lifecycle hook that publishes a message with metadata to the specified Amazon SQS queue.

```
aws autoscaling put-lifecycle-hook \  
  --auto-scaling-group-name my-asg \  
  --lifecycle-hook-name my-launch-hook \  
  --lifecycle-transition autoscaling:EC2_INSTANCE_LAUNCHING \  
  --notification-target-arn arn:aws:sqs:us-west-2:123456789012:my-sqs-queue \  
  --role-arn arn:aws:iam::123456789012:role/my-notification-role \  
  --notification-metadata "SQS message metadata"
```

This command produces no output.

For more information, see [Amazon EC2 Auto Scaling lifecycle hooks](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [PutLifecycleHook](#) in *AWS CLI Command Reference*.

put-notification-configuration

The following code example shows how to use `put-notification-configuration`.

AWS CLI

To add a notification

This example adds the specified notification to the specified Auto Scaling group.

```
aws autoscaling put-notification-configuration \  
  --auto-scaling-group-name my-asg \  
  --notification-configuration
```

```
--topic-arn arn:aws:sns:us-west-2:123456789012:my-sns-topic \  
--notification-type autoscaling:TEST_NOTIFICATION
```

This command produces no output.

For more information, see [Getting Amazon SNS notifications when your Auto Scaling group scales](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [PutNotificationConfiguration](#) in *AWS CLI Command Reference*.

put-scaling-policy

The following code example shows how to use `put-scaling-policy`.

AWS CLI

To add a target tracking scaling policy to an Auto Scaling group

The following `put-scaling-policy` example applies a target tracking scaling policy to the specified Auto Scaling group. The output contains the ARNs and names of the two CloudWatch alarms created on your behalf. If a scaling policy with the same name already exists, it will be overwritten by the new scaling policy.

```
aws autoscaling put-scaling-policy --auto-scaling-group-name my-asg \  
--policy-name alb1000-target-tracking-scaling-policy \  
--policy-type TargetTrackingScaling \  
--target-tracking-configuration file://config.json
```

Contents of `config.json`:

```
{  
  "TargetValue": 1000.0,  
  "PredefinedMetricSpecification": {  
    "PredefinedMetricType": "ALBRequestCountPerTarget",  
    "ResourceLabel": "app/my-alb/778d41231b141a0f/targetgroup/my-alb-target-  
group/943f017f100becff"  
  }  
}
```

Output:


```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:228f02c2-
c665-4bfd-aaac-8b04080bea3c:autoScalingGroupName/my-asg:policyName/alb1000-target-
tracking-scaling-policy",
  "Alarms": [
    {
      "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
my-asg-AlarmHigh-fc0e4183-23ac-497e-9992-691c9980c38e",
      "AlarmName": "TargetTracking-my-asg-AlarmHigh-
fc0e4183-23ac-497e-9992-691c9980c38e"
    },
    {
      "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
my-asg-AlarmLow-61a39305-ed0c-47af-bd9e-471a352ee1a2",
      "AlarmName": "TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2"
    }
  ]
}
```

For more examples, see [Example scaling policies for the AWS Command Line Interface \(AWS CLI\)](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [PutScalingPolicy](#) in *AWS CLI Command Reference*.

put-scheduled-update-group-action

The following code example shows how to use `put-scheduled-update-group-action`.

AWS CLI

Example 1: To add a scheduled action to an Auto Scaling group

This example adds the specified scheduled action to the specified Auto Scaling group.

```
aws autoscaling put-scheduled-update-group-action \
  --auto-scaling-group-name my-asg \
  --scheduled-action-name my-scheduled-action \
  --start-time "2023-05-12T08:00:00Z" \
  --min-size 2 \
  --max-size 6 \
  --desired-capacity 4
```

This command produces no output. If a scheduled action with the same name already exists, it will be overwritten by the new scheduled action.

For more examples, see [Scheduled scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 2: To specify a recurring schedule

This example creates a scheduled action to scale on a recurring schedule that is scheduled to execute at 00:30 hours on the first of January, June, and December every year.

```
aws autoscaling put-scheduled-update-group-action \  
  --auto-scaling-group-name my-asg \  
  --scheduled-action-name my-recurring-action \  
  --recurrence "30 0 1 1,6,12 *" \  
  --min-size 2 \  
  --max-size 6 \  
  --desired-capacity 4
```

This command produces no output. If a scheduled action with the same name already exists, it will be overwritten by the new scheduled action.

For more examples, see [Scheduled scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [PutScheduledUpdateGroupAction](#) in *AWS CLI Command Reference*.

put-warm-pool

The following code example shows how to use `put-warm-pool`.

AWS CLI

To create a warm pool

The following example creates a warm pool for the specified Auto Scaling group.

```
aws autoscaling put-warm-pool \  
  --auto-scaling-group-name my-asg \  
  --min-size 2
```

This command produces no output. If a warm pool already exists, it will be updated.

For more information, see [Warm pools for Amazon EC2 Auto Scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [PutWarmPool](#) in *AWS CLI Command Reference*.

record-lifecycle-action-heartbeat

The following code example shows how to use `record-lifecycle-action-heartbeat`.

AWS CLI

To record a lifecycle action heartbeat

This example records a lifecycle action heartbeat to keep the instance in a pending state.

```
aws autoscaling record-lifecycle-action-heartbeat \  
  --lifecycle-hook-name my-launch-hook \  
  --auto-scaling-group-name my-asg \  
  --lifecycle-action-token bcd2f1b8-9a78-44d3-8a7a-4dd07d7cf635
```

This command produces no output.

For more information, see [Amazon EC2 Auto Scaling lifecycle hooks](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [RecordLifecycleActionHeartbeat](#) in *AWS CLI Command Reference*.

resume-processes

The following code example shows how to use `resume-processes`.

AWS CLI

To resume suspended processes

This example resumes the specified suspended scaling process for the specified Auto Scaling group.

```
aws autoscaling resume-processes \  
  --auto-scaling-group-name my-asg \  
  --scaling-processes AlarmNotification
```

This command produces no output.

For more information, see [Suspending and resuming scaling processes](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [ResumeProcesses](#) in *AWS CLI Command Reference*.

rollback-instance-refresh

The following code example shows how to use `rollback-instance-refresh`.

AWS CLI

To roll back an instance refresh

The following `rollback-instance-refresh` example rolls back an in-progress instance refresh for the specified Auto Scaling group.

```
aws autoscaling rollback-instance-refresh \  
  --auto-scaling-group-name my-asg
```

Output:

```
{  
  "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"  
}
```

For more information, see [Undo changes with a rollback](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [RollbackInstanceRefresh](#) in *AWS CLI Command Reference*.

set-desired-capacity

The following code example shows how to use `set-desired-capacity`.

AWS CLI

To set the desired capacity for an Auto Scaling group

This example sets the desired capacity for the specified Auto Scaling group.

```
aws autoscaling set-desired-capacity \  
  --auto-scaling-group-name my-asg \  
  --desired-capacity 2
```

```
--desired-capacity 2 \  
--honor-cooldown
```

This command returns to the prompt if successful.

- For API details, see [SetDesiredCapacity](#) in *AWS CLI Command Reference*.

set-instance-health

The following code example shows how to use `set-instance-health`.

AWS CLI

To set the health status of an instance

This example sets the health status of the specified instance to Unhealthy.

```
aws autoscaling set-instance-health \  
  --instance-id i-061c63c5eb45f0416 \  
  --health-status Unhealthy
```

This command produces no output.

- For API details, see [SetInstanceHealth](#) in *AWS CLI Command Reference*.

set-instance-protection

The following code example shows how to use `set-instance-protection`.

AWS CLI

Example 1: To enable the instance protection setting for an instance

This example enables instance protection for the specified instance.

```
aws autoscaling set-instance-protection \  
  --instance-ids i-061c63c5eb45f0416 \  
  --auto-scaling-group-name my-asg --protected-from-scale-in
```

This command produces no output.

Example 2: To disable the instance protection setting for an instance

This example disables instance protection for the specified instance.

```
aws autoscaling set-instance-protection \  
  --instance-ids i-061c63c5eb45f0416 \  
  --auto-scaling-group-name my-asg \  
  --no-protected-from-scale-in
```

This command produces no output.

- For API details, see [SetInstanceProtection](#) in *AWS CLI Command Reference*.

start-instance-refresh

The following code example shows how to use `start-instance-refresh`.

AWS CLI

Example 1: To start an instance refresh using command line parameters

The following `start-instance-refresh` example starts an instance refresh using command line arguments. The optional `preferences` parameter specifies an `InstanceWarmup` of 60 seconds and a `MinHealthyPercentage` of 50 percent.

```
aws autoscaling start-instance-refresh \  
  --auto-scaling-group-name my-asg \  
  --preferences '{"InstanceWarmup": 60, "MinHealthyPercentage": 50}'
```

Output:

```
{  
  "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"  
}
```

For more information, see [Start an instance refresh](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 2: To start an instance refresh using a JSON file

The following `start-instance-refresh` example starts an instance refresh using a JSON file. You can specify the Auto Scaling group and define your desired configuration and preferences in a JSON file, as shown in the following example.

```
aws autoscaling start-instance-refresh \  
  --cli-input-json file://config.json
```

Contents of config.json:

```
{  
  "AutoScalingGroupName": "my-asg",  
  "DesiredConfiguration": {  
    "LaunchTemplate": {  
      "LaunchTemplateId": "lt-068f72b729example",  
      "Version": "$Default"  
    }  
  },  
  "Preferences": {  
    "InstanceWarmup": 60,  
    "MinHealthyPercentage": 50,  
    "AutoRollback": true,  
    "ScaleInProtectedInstances": Ignore,  
    "StandbyInstances": Terminate  
  }  
}
```

Output:

```
{  
  "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"  
}
```

For more information, see [Start an instance refresh](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [StartInstanceRefresh](#) in *AWS CLI Command Reference*.

suspend-processes

The following code example shows how to use suspend-processes.

AWS CLI

To suspend Auto Scaling processes

This example suspends the specified scaling process for the specified Auto Scaling group.

```
aws autoscaling suspend-processes \  
  --auto-scaling-group-name my-asg \  
  --scaling-processes AlarmNotification
```

This command produces no output.

For more information, see [Suspending and resuming scaling processes](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [SuspendProcesses](#) in *AWS CLI Command Reference*.

terminate-instance-in-auto-scaling-group

The following code example shows how to use `terminate-instance-in-auto-scaling-group`.

AWS CLI

To terminate an instance in an Auto Scaling group

This example terminates the specified instance from the specified Auto Scaling group without updating the size of the group. Amazon EC2 Auto Scaling launches a replacement instance after the specified instance terminates.

```
aws autoscaling terminate-instance-in-auto-scaling-group \  
  --instance-id i-061c63c5eb45f0416 \  
  --no-should-decrement-desired-capacity
```

Output:

```
{  
  "Activities": [  
    {  
      "ActivityId": "8c35d601-793c-400c-fcd0-f64a27530df7",  
      "AutoScalingGroupName": "my-asg",  
      "Description": "Terminating EC2 instance: i-061c63c5eb45f0416",  
      "Cause": "",  
      "StartTime": "2020-10-31T20:34:25.680Z",  
      "StatusCode": "InProgress",  
      "Progress": 0,  
      "Details": "{\"Subnet ID\":\"subnet-6194ea3b\",\"Availability Zone\":  
        \"us-west-2c\"}"
```



```
    }  
  ]  
}
```

- For API details, see [TerminateInstanceInAutoScalingGroup](#) in *AWS CLI Command Reference*.

update-auto-scaling-group

The following code example shows how to use `update-auto-scaling-group`.

AWS CLI

Example 1: To update the size limits of an Auto Scaling group

This example updates the specified Auto Scaling group with a minimum size of 2 and a maximum size of 10.

```
aws autoscaling update-auto-scaling-group \  
  --auto-scaling-group-name my-asg \  
  --min-size 2 \  
  --max-size 10
```

This command produces no output.

For more information, see [Setting capacity limits for your Auto Scaling group](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 2: To add Elastic Load Balancing health checks and specify which Availability Zones and subnets to use

This example updates the specified Auto Scaling group to add Elastic Load Balancing health checks. This command also updates the value of `--vpc-zone-identifier` with a list of subnet IDs in multiple Availability Zones.

```
aws autoscaling update-auto-scaling-group \  
  --auto-scaling-group-name my-asg \  
  --health-check-type ELB \  
  --health-check-grace-period 600 \  
  --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

This command produces no output.

For more information, see [Elastic Load Balancing and Amazon EC2 Auto Scaling](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 3: To update the placement group and termination policy

This example updates the placement group and termination policy to use.

```
aws autoscaling update-auto-scaling-group \  
  --auto-scaling-group-name my-asg \  
  --placement-group my-placement-group \  
  --termination-policies "OldestInstance"
```

This command produces no output.

For more information, see [Auto Scaling groups](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 4: To use the latest version of the launch template

This example updates the specified Auto Scaling group to use the latest version of the specified launch template.

```
aws autoscaling update-auto-scaling-group \  
  --auto-scaling-group-name my-asg \  
  --launch-template LaunchTemplateId=lt-1234567890abcde12,Version='$Latest'
```

This command produces no output.

For more information, see [Launch templates](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 5: To use a specific version of the launch template

This example updates the specified Auto Scaling group to use a specific version of a launch template instead of the latest or default version.

```
aws autoscaling update-auto-scaling-group \  
  --auto-scaling-group-name my-asg \  
  --launch-template LaunchTemplateName=my-template-for-auto-scaling,Version='2'
```

This command produces no output.

For more information, see [Launch templates](#) in the *Amazon EC2 Auto Scaling User Guide*.

Example 6: To define a mixed instances policy and enable capacity rebalancing

This example updates the specified Auto Scaling group to use a mixed instances policy and enables capacity rebalancing. This structure lets you specify groups with Spot and On-Demand capacities and use different launch templates for different architectures.

```
aws autoscaling update-auto-scaling-group \  
  --cli-input-json file://~/config.json
```

Contents of config.json:

```
{  
  "AutoScalingGroupName": "my-asg",  
  "CapacityRebalance": true,  
  "MixedInstancesPolicy": {  
    "LaunchTemplate": {  
      "LaunchTemplateSpecification": {  
        "LaunchTemplateName": "my-launch-template-for-x86",  
        "Version": "$Latest"  
      },  
      "Overrides": [  
        {  
          "InstanceType": "c6g.large",  
          "LaunchTemplateSpecification": {  
            "LaunchTemplateName": "my-launch-template-for-arm",  
            "Version": "$Latest"  
          }  
        },  
        {  
          "InstanceType": "c5.large"  
        },  
        {  
          "InstanceType": "c5a.large"  
        }  
      ]  
    },  
    "InstancesDistribution": {  
      "OnDemandPercentageAboveBaseCapacity": 50,  
      "SpotAllocationStrategy": "capacity-optimized"  
    }  
  }  
}
```

This command produces no output.

For more information, see [Auto Scaling groups with multiple instance types and purchase options](#) in the *Amazon EC2 Auto Scaling User Guide*.

- For API details, see [UpdateAutoScalingGroup](#) in *AWS CLI Command Reference*.

Auto Scaling Plans examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Auto Scaling Plans.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-scaling-plan

The following code example shows how to use create-scaling-plan.

AWS CLI

To create a scaling plan

The following create-scaling-plan example creates a scaling plan named my-scaling-plan using an already-created JSON file (named config.json). The structure of the scaling plan includes a scaling instruction for an Auto Scaling group named my-asg. It specifies the TagFilters property as the application source and enables predictive scaling and dynamic scaling.

```
aws autoscaling-plans create-scaling-plan \  
  --scaling-plan-name my-scaling-plan \  
  --config-file config.json
```

```
--cli-input-json file://~/config.json
```

Contents of config.json file:

```
{
  "ApplicationSource": {
    "TagFilters": [
      {
        "Key": "purpose",
        "Values": [
          "my-application"
        ]
      }
    ]
  },
  "ScalingInstructions": [
    {
      "ServiceNamespace": "autoscaling",
      "ResourceId": "autoScalingGroup/my-asg",
      "ScalableDimension": "autoscaling:autoScalingGroup:DesiredCapacity",
      "ScheduledActionBufferTime": 300,
      "PredictiveScalingMaxCapacityBehavior":
"SetForecastCapacityToMaxCapacity",
      "PredictiveScalingMode": "ForecastAndScale",
      "PredefinedLoadMetricSpecification": {
        "PredefinedLoadMetricType": "ASGTotalCPUUtilization"
      },
      "ScalingPolicyUpdateBehavior": "ReplaceExternalPolicies",
      "MinCapacity": 1,
      "MaxCapacity": 4,
      "TargetTrackingConfigurations": [
        {
          "PredefinedScalingMetricSpecification": {
            "PredefinedScalingMetricType": "ASGAverageCPUUtilization"
          },
          "TargetValue": 50
        }
      ]
    }
  ]
}
```

Output:

```
{  
  "ScalingPlanVersion": 1  
}
```

For more information, see the [AWS Auto Scaling User Guide](#).

- For API details, see [CreateScalingPlan](#) in *AWS CLI Command Reference*.

delete-scaling-plan

The following code example shows how to use delete-scaling-plan.

AWS CLI

To delete a scaling plan

The following delete-scaling-plan example deletes the specified scaling plan.

```
aws autoscaling-plans delete-scaling-plan \  
  --scaling-plan-name my-scaling-plan \  
  --scaling-plan-version 1
```

This command produces no output.

For more information, see the [AWS Auto Scaling User Guide](#).

- For API details, see [DeleteScalingPlan](#) in *AWS CLI Command Reference*.

describe-scaling-plan-resources

The following code example shows how to use describe-scaling-plan-resources.

AWS CLI

To describe the scalable resources for a scaling plan

The following describe-scaling-plan-resources example displays details about the single scalable resource (an Auto Scaling group) that is associated with the specified scaling plan.

```
aws autoscaling-plans describe-scaling-plan-resources \  
  --scaling-plan-name my-scaling-plan \  
  --scaling-plan-version 1
```

```
--scaling-plan-name my-scaling-plan \
--scaling-plan-version 1
```

Output:

```
{
  "ScalingPlanResources": [
    {
      "ScalableDimension": "autoscaling:autoScalingGroup:DesiredCapacity",
      "ScalingPlanVersion": 1,
      "ResourceId": "autoScalingGroup/my-asg",
      "ScalingStatusCode": "Active",
      "ScalingStatusMessage": "Target tracking scaling policies have been
applied to the resource.",
      "ScalingPolicies": [
        {
          "PolicyName": "AutoScaling-my-asg-b1ab65ae-4be3-4634-bd64-
c7471662b251",
          "PolicyType": "TargetTrackingScaling",
          "TargetTrackingConfiguration": {
            "PredefinedScalingMetricSpecification": {
              "PredefinedScalingMetricType":
"ALBRequestCountPerTarget",
              "ResourceLabel": "app/my-alb/f37c06a68c1748aa/
targetgroup/my-target-group/6d4ea56ca2d6a18d"
            },
            "TargetValue": 40.0
          }
        }
      ],
      "ServiceNamespace": "autoscaling",
      "ScalingPlanName": "my-scaling-plan"
    }
  ]
}
```

For more information, see [What Is AWS Auto Scaling?](#) in the *AWS Auto Scaling User Guide*.

- For API details, see [DescribeScalingPlanResources](#) in *AWS CLI Command Reference*.

describe-scaling-plans

The following code example shows how to use describe-scaling-plans.

AWS CLI

To describe a scaling plan

The following `describe-scaling-plans` example displays the details of the specified scaling plan.

```
aws autoscaling-plans describe-scaling-plans \  
  --scaling-plan-names scaling-plan-with-asg-and-ddb
```

Output:

```
{  
  "ScalingPlans": [  
    {  
      "LastMutatingRequestTime": 1565388443.963,  
      "ScalingPlanVersion": 1,  
      "CreationTime": 1565388443.963,  
      "ScalingInstructions": [  
        {  
          "ScalingPolicyUpdateBehavior": "ReplaceExternalPolicies",  
          "ScalableDimension":  
"autoscaling:autoScalingGroup:DesiredCapacity",  
          "TargetTrackingConfigurations": [  
            {  
              "PredefinedScalingMetricSpecification": {  
                "PredefinedScalingMetricType":  
"ASGAverageCPUUtilization"  
              },  
              "TargetValue": 50.0,  
              "EstimatedInstanceWarmup": 300,  
              "DisableScaleIn": false  
            }  
          ],  
          "ResourceId": "autoScalingGroup/my-asg",  
          "DisableDynamicScaling": false,  
          "MinCapacity": 1,  
          "ServiceNamespace": "autoscaling",  
          "MaxCapacity": 10  
        },  
        {  
          "ScalingPolicyUpdateBehavior": "ReplaceExternalPolicies",  
          "ScalableDimension": "dynamodb:table:ReadCapacityUnits",
```



```

    "TargetTrackingConfigurations": [
      {
        "PredefinedScalingMetricSpecification": {
          "PredefinedScalingMetricType":
"DynamoDBReadCapacityUtilization"
          },
        "TargetValue": 50.0,
        "ScaleInCooldown": 60,
        "DisableScaleIn": false,
        "ScaleOutCooldown": 60
      }
    ],
    "ResourceId": "table/my-table",
    "DisableDynamicScaling": false,
    "MinCapacity": 5,
    "ServiceNamespace": "dynamodb",
    "MaxCapacity": 10000
  },
  {
    "ScalingPolicyUpdateBehavior": "ReplaceExternalPolicies",
    "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
    "TargetTrackingConfigurations": [
      {
        "PredefinedScalingMetricSpecification": {
          "PredefinedScalingMetricType":
"DynamoDBWriteCapacityUtilization"
          },
        "TargetValue": 50.0,
        "ScaleInCooldown": 60,
        "DisableScaleIn": false,
        "ScaleOutCooldown": 60
      }
    ],
    "ResourceId": "table/my-table",
    "DisableDynamicScaling": false,
    "MinCapacity": 5,
    "ServiceNamespace": "dynamodb",
    "MaxCapacity": 10000
  }
],
"ApplicationSource": {
  "TagFilters": [
    {
      "Values": [

```

```

        "my-application-id"
      ],
      "Key": "application"
    }
  ]
},
"StatusStartTime": 1565388455.836,
"ScalingPlanName": "scaling-plan-with-asg-and-ddb",
"StatusMessage": "Scaling plan has been created and applied to all
resources.",
"StatusCode": "Active"
}
]
}

```

For more information, see [What Is AWS Auto Scaling?](#) in the *AWS Auto Scaling User Guide*.

- For API details, see [DescribeScalingPlans](#) in *AWS CLI Command Reference*.

get-scaling-plan-resource-forecast-data

The following code example shows how to use `get-scaling-plan-resource-forecast-data`.

AWS CLI

To retrieve load forecast data

This example retrieves load forecast data for a scalable resource (an Auto Scaling group) that is associated with the specified scaling plan.

```

aws autoscaling-plans get-scaling-plan-resource-forecast-data \
  --scaling-plan-name my-scaling-plan \
  --scaling-plan-version 1 \
  --service-namespace "autoscaling" \
  --resource-id autoScalingGroup/my-asg \
  --scalable-dimension "autoscaling:autoScalingGroup:DesiredCapacity" \
  --forecast-data-type "LoadForecast" \
  --start-time "2019-08-30T00:00:00Z" \
  --end-time "2019-09-06T00:00:00Z"

```

Output:

```
{
```

```
"Datapoints": [...]  
}
```

For more information, see [What Is AWS Auto Scaling](#) in the *AWS Auto Scaling User Guide*.

- For API details, see [GetScalingPlanResourceForecastData](#) in *AWS CLI Command Reference*.

update-scaling-plan

The following code example shows how to use `update-scaling-plan`.

AWS CLI

To update a scaling plan

The following `update-scaling-plan` example modifies the scaling metric for an Auto Scaling group in the specified scaling plan.

```
aws autoscaling-plans update-scaling-plan \  
  --scaling-plan-name my-scaling-plan \  
  --scaling-plan-version 1 \  
  --scaling-instructions  
  '{"ScalableDimension":"autoscaling:autoScalingGroup:DesiredCapacity","ResourceId":"autoScal  
my-asg","ServiceNamespace":"autoscaling","TargetTrackingConfigurations":  
[{"PredefinedScalingMetricSpecification":  
  {"PredefinedScalingMetricType":"ALBRequestCountPerTarget","ResourceLabel":"app/my-  
alb/f37c06a68c1748aa/targetgroup/my-target-  
group/6d4ea56ca2d6a18d"},"TargetValue":40.0}],"MinCapacity": 1,"MaxCapacity": 10}'
```

This command produces no output.

For more information, see [What Is AWS Auto Scaling?](#) in the *AWS Auto Scaling User Guide*.

- For API details, see [UpdateScalingPlan](#) in *AWS CLI Command Reference*.

AWS Backup examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Backup.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-backup-plan

The following code example shows how to use `create-backup-plan`.

AWS CLI

To create a backup plan

The following `create-backup-plan` example creates the specified backup plan with a 35 day retention.

```
aws backup create-backup-plan \  
--backup-plan "{\"BackupPlanName\": \"Example-Backup-Plan\", \"Rules\": [{\"RuleName\": \  
\"DailyBackups\", \"ScheduleExpression\": \"cron(0 5 ? * * *)\", \"StartWindowMinutes \  
\":480, \"TargetBackupVaultName\": \"Default\", \"Lifecycle\": {\"DeleteAfterDays \  
\":35}}]}\"
```

Output:

```
{  
  "BackupPlanId": "1fa3895c-a7f5-484a-a371-2dd6a1a9f729",  
  "BackupPlanArn": "arn:aws:backup:us-west-2:123456789012:backup-plan:1fa3895c-  
a7f5-484a-a371-2dd6a1a9f729",  
  "CreationDate": 1568928754.747,  
  "VersionId": "ZjQ2ZTI5YWQtZDg5Yi00MzYzLWJmZTAzMDE1Mzh1MDhjYjEz"  
}
```

For more information, see [Creating a Backup Plan](#) in the *AWS Backup Developer Guide*.

- For API details, see [CreateBackupPlan](#) in *AWS CLI Command Reference*.

create-backup-vault

The following code example shows how to use `create-backup-vault`.

AWS CLI

To create a backup vault

The following `create-backup-vault` example creates a backup vault with the specified name.

```
aws backup create-backup-vault
  --backup-vault-name sample-vault
```

This command produces no output. Output:

```
{
  "BackupVaultName": "sample-vault",
  "BackupVaultArn": "arn:aws:backup:us-west-2:123456789012:backup-vault:sample-
vault",
  "CreationDate": 1568928338.385
}
```

For more information, see [Creating a Backup Vault](#) in the *AWS Backup Developer Guide*.

- For API details, see [CreateBackupVault](#) in *AWS CLI Command Reference*.

get-backup-plan-from-template

The following code example shows how to use `get-backup-plan-from-template`.

AWS CLI

To get an existing backup plan from a template

The following `get-backup-plan-from-template` example gets an existing backup plan from a template that specifies a daily backup with a 35 day retention.

```
aws backup get-backup-plan-from-template \  
  --backup-plan-template-id "87c0c1ef-254d-4180-8fef-2e76a2c38aaa"
```

Output:

```
{  
  "BackupPlanDocument": {  
    "Rules": [  
      {  
        "RuleName": "DailyBackups",  
        "ScheduleExpression": "cron(0 5 ? * * *)",  
        "StartWindowMinutes": 480,  
        "Lifecycle": {  
          "DeleteAfterDays": 35  
        }  
      }  
    ]  
  }  
}
```

For more information, see [Creating a Backup Plan](#) in the *AWS Backup Developer Guide*.

- For API details, see [GetBackupPlanFromTemplate](#) in *AWS CLI Command Reference*.

get-backup-plan

The following code example shows how to use `get-backup-plan`.

AWS CLI

To get the details of a backup plan

The following `get-backup-plan` example displays the details of the specified backup plan.

```
aws backup get-backup-plan \  
  --backup-plan-id "fcbf5d8f-bd77-4f3a-9c97-f24fb3d373a5"
```

Output:

```
{  
  "BackupPlan": {
```

```
"BackupPlanName": "Example-Backup-Plan",
"Rules": [
  {
    "RuleName": "DailyBackups",
    "TargetBackupVaultName": "Default",
    "ScheduleExpression": "cron(0 5 ? * * *)",
    "StartWindowMinutes": 480,
    "CompletionWindowMinutes": 10080,
    "Lifecycle": {
      "DeleteAfterDays": 35
    },
    "RuleId": "70e0ccdc-e9df-4e83-82ad-c1e5a9471cc3"
  }
],
"BackupPlanId": "fcbf5d8f-bd77-4f3a-9c97-f24fb3d373a5",
"BackupPlanArn": "arn:aws:backup:us-west-2:123456789012:backup-plan:fcbf5d8f-
bd77-4f3a-9c97-f24fb3d373a5",
"VersionId": "NjQ2ZTZkODktMGVhNy00MmQ0LWE4YjktZTkWNTQ3OTkyYTcw",
"CreationDate": 1568926091.57
}
```

For more information, see [Creating a Backup Plan](#) in the *AWS Backup Developer Guide*.

- For API details, see [GetBackupPlan](#) in *AWS CLI Command Reference*.

list-backup-jobs

The following code example shows how to use `list-backup-jobs`.

AWS CLI

Example 1: To list all backup jobs

The following `list-backup-jobs` example returns metadata about your backup jobs in your AWS account.

```
aws backup list-backup-jobs
```

Output:

```
{
```

```

"BackupJobs": [
  {
    "BackupJobId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "BackupVaultName": "Default",
    "BackupVaultArn": "arn:aws:backup:us-west-2:123456789012:backup-
vault:Default",
    "ResourceArn": "arn:aws:ec2:us-west-2:123456789012:instance/
i-12345678901234567",
    "CreationDate": 1600721892.929,
    "State": "CREATED",
    "PercentDone": "0.0",
    "IamRoleArn": "arn:aws:iam::123456789012:role/service-role/
AWSBackupDefaultServiceRole",
    "StartBy": 1600725492.929,
    "ResourceType": "EC2"
  },
  {
    "BackupJobId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "BackupVaultName": "Default",
    "BackupVaultArn": "arn:aws:backup:us-west-2:123456789012:backup-
vault:Default",
    "RecoveryPointArn": "arn:aws:backup:us-west-2:123456789012:recovery-
point:a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "ResourceArn": "arn:aws:elasticfilesystem:us-west-2:123456789012:file-
system/fs-12345678",
    "CreationDate": 1600721724.77,
    "CompletionDate": 1600721744.488,
    "State": "COMPLETED",
    "PercentDone": "100.0",
    "BackupSizeInBytes": 71,
    "IamRoleArn": "arn:aws:iam::123456789012:role/service-role/
AWSBackupDefaultServiceRole",
    "StartBy": 1600725324.77,
    "ResourceType": "EFS"
  }
]
}

```

For more information, see [Creating a Backup](#) in the *AWS Backup Developer Guide*.

Example 2: To list completed backup jobs

The following `list-backup-jobs` example returns metadata about your completed backup jobs in your AWS account.


```
aws backup list-backup-jobs \  
  --by-state COMPLETED
```

Output:

```
{  
  "BackupJobs": [  
    {  
      "BackupJobId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",  
      "BackupVaultName": "Default",  
      "BackupVaultArn": "arn:aws:backup:us-west-2:123456789012:backup-  
vault:Default",  
      "RecoveryPointArn": "arn:aws:backup:us-west-2:123456789012:recovery-  
point:a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",  
      "ResourceArn": "arn:aws:elasticfilesystem:us-west-2:123456789012:file-  
system/fs-12345678",  
      "CreationDate": 1600721724.77,  
      "CompletionDate": 1600721744.488,  
      "State": "COMPLETED",  
      "PercentDone": "100.0",  
      "BackupSizeInBytes": 71,  
      "IamRoleArn": "arn:aws:iam::123456789012:role/service-role/  
AWSBackupDefaultServiceRole",  
      "StartBy": 1600725324.77,  
      "ResourceType": "EFS"  
    }  
  ]  
}
```

For more information, see [Creating a Backup](#) in the *AWS Backup Developer Guide*.

- For API details, see [ListBackupJobs](#) in *AWS CLI Command Reference*.

AWS Batch examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Batch.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

cancel-job

The following code example shows how to use `cancel-job`.

AWS CLI

To cancel a job

This example cancels a job with the specified job ID.

Command:

```
aws batch cancel-job --job-id bcf0b186-a532-4122-842e-2ccab8d54efb --reason "Cancelling job."
```

- For API details, see [CancelJob](#) in *AWS CLI Command Reference*.

create-compute-environment

The following code example shows how to use `create-compute-environment`.

AWS CLI

To create a managed compute environment with On-Demand instances

This example creates a managed compute environment with specific C4 instance types that are launched on demand. The compute environment is called `C4OnDemand`.

Command:

```
aws batch create-compute-environment --cli-input-json file://<path_to_json_file>/C4OnDemand.json
```

JSON file format:

```
{
  "computeEnvironmentName": "C4OnDemand",
  "type": "MANAGED",
  "state": "ENABLED",
  "computeResources": {
    "type": "EC2",
    "minvCpus": 0,
    "maxvCpus": 128,
    "desiredvCpus": 48,
    "instanceTypes": [
      "c4.large",
      "c4.xlarge",
      "c4.2xlarge",
      "c4.4xlarge",
      "c4.8xlarge"
    ],
    "subnets": [
      "subnet-220c0e0a",
      "subnet-1a95556d",
      "subnet-978f6dce"
    ],
    "securityGroupIds": [
      "sg-cf5093b2"
    ],
    "ec2KeyPair": "id_rsa",
    "instanceRole": "ecsInstanceRole",
    "tags": {
      "Name": "Batch Instance - C4OnDemand"
    }
  },
  "serviceRole": "arn:aws:iam::012345678910:role/AWSBatchServiceRole"
}
```

Output:

```
{
  "computeEnvironmentName": "C4OnDemand",
```

```
"computeEnvironmentArn": "arn:aws:batch:us-east-1:012345678910:compute-
environment/C4OnDemand"
}
```

To create a managed compute environment with Spot Instances

This example creates a managed compute environment with the M4 instance type that is launched when the Spot bid price is at or below 20% of the On-Demand price for the instance type. The compute environment is called M4Spot.

Command:

```
aws batch create-compute-environment --cli-input-json file://<path_to_json_file>/
M4Spot.json
```

JSON file format:

```
{
  "computeEnvironmentName": "M4Spot",
  "type": "MANAGED",
  "state": "ENABLED",
  "computeResources": {
    "type": "SPOT",
    "spotIamFleetRole": "arn:aws:iam::012345678910:role/aws-ec2-spot-fleet-role",
    "minvCpus": 0,
    "maxvCpus": 128,
    "desiredvCpus": 4,
    "instanceTypes": [
      "m4"
    ],
    "bidPercentage": 20,
    "subnets": [
      "subnet-220c0e0a",
      "subnet-1a95556d",
      "subnet-978f6dce"
    ],
    "securityGroupIds": [
      "sg-cf5093b2"
    ],
    "ec2KeyPair": "id_rsa",
    "instanceRole": "ecsInstanceRole",
    "tags": {
```

```
    "Name": "Batch Instance - M4Spot"
  }
},
"serviceRole": "arn:aws:iam::012345678910:role/AWSBatchServiceRole"
}
```

Output:

```
{
  "computeEnvironmentName": "M4Spot",
  "computeEnvironmentArn": "arn:aws:batch:us-east-1:012345678910:compute-
environment/M4Spot"
}
```

- For API details, see [CreateComputeEnvironment](#) in *AWS CLI Command Reference*.

create-job-queue

The following code example shows how to use `create-job-queue`.

AWS CLI

To create a low priority job queue with a single compute environment

This example creates a job queue called `LowPriority` that uses the `M4Spot` compute environment.

Command:

```
aws batch create-job-queue --cli-input-json file://<path_to_json_file>/
LowPriority.json
```

JSON file format:

```
{
  "jobQueueName": "LowPriority",
  "state": "ENABLED",
  "priority": 10,
  "computeEnvironmentOrder": [
    {
```

```
    "order": 1,
    "computeEnvironment": "M4Spot"
  }
]
```

Output:

```
{
  "jobQueueArn": "arn:aws:batch:us-east-1:012345678910:job-queue/LowPriority",
  "jobQueueName": "LowPriority"
}
```

To create a high priority job queue with two compute environments

This example creates a job queue called HighPriority that uses the C4OnDemand compute environment with an order of 1 and the M4Spot compute environment with an order of 2. The scheduler will attempt to place jobs on the C4OnDemand compute environment first.

Command:

```
aws batch create-job-queue --cli-input-json file://<path_to_json_file>/
HighPriority.json
```

JSON file format:

```
{
  "jobQueueName": "HighPriority",
  "state": "ENABLED",
  "priority": 1,
  "computeEnvironmentOrder": [
    {
      "order": 1,
      "computeEnvironment": "C4OnDemand"
    },
    {
      "order": 2,
      "computeEnvironment": "M4Spot"
    }
  ]
}
```

Output:

```
{
  "jobQueueArn": "arn:aws:batch:us-east-1:012345678910:job-queue/HighPriority",
  "jobQueueName": "HighPriority"
}
```

- For API details, see [CreateJobQueue](#) in *AWS CLI Command Reference*.

delete-compute-environment

The following code example shows how to use `delete-compute-environment`.

AWS CLI**To delete a compute environment**

This example deletes the P2OnDemand compute environment.

Command:

```
aws batch delete-compute-environment --compute-environment P2OnDemand
```

- For API details, see [DeleteComputeEnvironment](#) in *AWS CLI Command Reference*.

delete-job-queue

The following code example shows how to use `delete-job-queue`.

AWS CLI**To delete a job queue**

This example deletes the GPGPU job queue.

Command:

```
aws batch delete-job-queue --job-queue GPGPU
```

- For API details, see [DeleteJobQueue](#) in *AWS CLI Command Reference*.

deregister-job-definition

The following code example shows how to use `deregister-job-definition`.

AWS CLI

To deregister a job definition

This example deregisters a job definition called `sleep10`.

Command:

```
aws batch deregister-job-definition --job-definition sleep10
```

- For API details, see [DeregisterJobDefinition](#) in *AWS CLI Command Reference*.

describe-compute-environments

The following code example shows how to use `describe-compute-environments`.

AWS CLI

To describe a compute environment

This example describes the `P2OnDemand` compute environment.

Command:

```
aws batch describe-compute-environments --compute-environments P2OnDemand
```

Output:

```
{
  "computeEnvironments": [
    {
      "status": "VALID",
      "serviceRole": "arn:aws:iam::012345678910:role/AWSBatchServiceRole",
      "computeEnvironmentArn": "arn:aws:batch:us-east-1:012345678910:compute-environment/P2OnDemand",
      "computeResources": {
        "subnets": [
          "subnet-220c0e0a",
          "subnet-1a95556d",
```



```

        "subnet-978f6dce"
      ],
      "tags": {
        "Name": "Batch Instance - P2OnDemand"
      },
      "desiredvCpus": 48,
      "minvCpus": 0,
      "instanceTypes": [
        "p2"
      ],
      "securityGroupIds": [
        "sg-cf5093b2"
      ],
      "instanceRole": "ecsInstanceRole",
      "maxvCpus": 128,
      "type": "EC2",
      "ec2KeyPair": "id_rsa"
    },
    "statusReason": "ComputeEnvironment Healthy",
    "ecsClusterArn": "arn:aws:ecs:us-east-1:012345678910:cluster/P2OnDemand_Batch_2c06f29d-d1fe-3a49-879d-42394c86effc",
    "state": "ENABLED",
    "computeEnvironmentName": "P2OnDemand",
    "type": "MANAGED"
  }
]
}

```

- For API details, see [DescribeComputeEnvironments](#) in *AWS CLI Command Reference*.

describe-job-definitions

The following code example shows how to use `describe-job-definitions`.

AWS CLI

To describe active job definitions

This example describes all of your active job definitions.

Command:

```
aws batch describe-job-definitions --status ACTIVE
```

Output:

```
{
  "jobDefinitions": [
    {
      "status": "ACTIVE",
      "jobDefinitionArn": "arn:aws:batch:us-east-1:012345678910:job-
definition/sleep60:1",
      "containerProperties": {
        "mountPoints": [],
        "parameters": {},
        "image": "busybox",
        "environment": {},
        "vcpus": 1,
        "command": [
          "sleep",
          "60"
        ],
        "volumes": [],
        "memory": 128,
        "ulimits": []
      },
      "type": "container",
      "jobDefinitionName": "sleep60",
      "revision": 1
    }
  ]
}
```

- For API details, see [DescribeJobDefinitions](#) in *AWS CLI Command Reference*.

describe-job-queues

The following code example shows how to use `describe-job-queues`.

AWS CLI**To describe a job queue**

This example describes the HighPriority job queue.

Command:

```
aws batch describe-job-queues --job-queues HighPriority
```

Output:

```
{
  "jobQueues": [
    {
      "status": "VALID",
      "jobQueueArn": "arn:aws:batch:us-east-1:012345678910:job-queue/HighPriority",
      "computeEnvironmentOrder": [
        {
          "computeEnvironment": "arn:aws:batch:us-east-1:012345678910:compute-environment/C4OnDemand",
          "order": 1
        }
      ],
      "statusReason": "JobQueue Healthy",
      "priority": 1,
      "state": "ENABLED",
      "jobQueueName": "HighPriority"
    }
  ]
}
```

- For API details, see [DescribeJobQueues](#) in *AWS CLI Command Reference*.

describe-jobs

The following code example shows how to use `describe-jobs`.

AWS CLI**To describe a job**

The following `describe-jobs` example describes a job with the specified job ID.

```
aws batch describe-jobs \
  --jobs bcf0b186-a532-4122-842e-2ccab8d54efb
```

Output:

```
{
  "jobs": [
    {
      "status": "SUBMITTED",
      "container": {
        "mountPoints": [],
        "image": "busybox",
        "environment": [],
        "vcpus": 1,
        "command": [
          "sleep",
          "60"
        ],
        "volumes": [],
        "memory": 128,
        "ulimits": []
      },
      "parameters": {},
      "jobDefinition": "arn:aws:batch:us-east-1:012345678910:job-definition/sleep60:1",
      "jobQueue": "arn:aws:batch:us-east-1:012345678910:job-queue/HighPriority",
      "jobId": "bcf0b186-a532-4122-842e-2ccab8d54efb",
      "dependsOn": [],
      "jobName": "example",
      "createdAt": 1480483387803
    }
  ]
}
```

- For API details, see [DescribeJobs](#) in *AWS CLI Command Reference*.

list-jobs

The following code example shows how to use `list-jobs`.

AWS CLI

To list running jobs

This example lists the running jobs in the HighPriority job queue.

Command:

```
aws batch list-jobs --job-queue HighPriority
```

Output:

```
{
  "jobSummaryList": [
    {
      "jobName": "example",
      "jobId": "e66ff5fd-a1ff-4640-b1a2-0b0a142f49bb"
    }
  ]
}
```

To list submitted jobs

This example lists jobs in the HighPriority job queue that are in the SUBMITTED job status.

Command:

```
aws batch list-jobs --job-queue HighPriority --job-status SUBMITTED
```

Output:

```
{
  "jobSummaryList": [
    {
      "jobName": "example",
      "jobId": "68f0c163-fbd4-44e6-9fd1-25b14a434786"
    }
  ]
}
```

- For API details, see [ListJobs](#) in *AWS CLI Command Reference*.

register-job-definition

The following code example shows how to use register-job-definition.

AWS CLI

To register a job definition

This example registers a job definition for a simple container job.

Command:

```
aws batch register-job-definition --job-definition-name sleep30 --type container --
container-properties '{ "image": "busybox", "vcpus": 1, "memory": 128, "command":
  [ "sleep", "30"]}'
```

Output:

```
{
  "jobDefinitionArn": "arn:aws:batch:us-east-1:012345678910:job-definition/
sleep30:1",
  "jobDefinitionName": "sleep30",
  "revision": 1
}
```

- For API details, see [RegisterJobDefinition](#) in *AWS CLI Command Reference*.

submit-job

The following code example shows how to use `submit-job`.

AWS CLI

To submit a job

This example submits a simple container job called `example` to the `HighPriority` job queue.

Command:

```
aws batch submit-job --job-name example --job-queue HighPriority --job-definition
sleep60
```

Output:

```
{
  "jobName": "example",
  "jobId": "876da822-4198-45f2-a252-6cea32512ea8"
}
```

- For API details, see [SubmitJob](#) in *AWS CLI Command Reference*.

terminate-job

The following code example shows how to use `terminate-job`.

AWS CLI

To terminate a job

This example terminates a job with the specified job ID.

Command:

```
aws batch terminate-job --job-id 61e743ed-35e4-48da-b2de-5c8333821c84 --reason "Terminating job."
```

- For API details, see [TerminateJob](#) in *AWS CLI Command Reference*.

update-compute-environment

The following code example shows how to use `update-compute-environment`.

AWS CLI

To update a compute environment

This example disables the P2OnDemand compute environment so it can be deleted.

Command:

```
aws batch update-compute-environment --compute-environment P2OnDemand --state DISABLED
```

Output:

```
{
  "computeEnvironmentName": "P2OnDemand",
  "computeEnvironmentArn": "arn:aws:batch:us-east-1:012345678910:compute-environment/P2OnDemand"
}
```

- For API details, see [UpdateComputeEnvironment](#) in *AWS CLI Command Reference*.

update-job-queue

The following code example shows how to use `update-job-queue`.

AWS CLI

To update a job queue

This example disables a job queue so that it can be deleted.

Command:

```
aws batch update-job-queue --job-queue GPGPU --state DISABLED
```

Output:

```
{
  "jobQueueArn": "arn:aws:batch:us-east-1:012345678910:job-queue/GPGPU",
  "jobQueueName": "GPGPU"
}
```

- For API details, see [UpdateJobQueue](#) in *AWS CLI Command Reference*.

AWS Budgets examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Budgets.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-budget

The following code example shows how to use create-budget.

AWS CLI

To create a Cost and Usage budget

The following create-budget command creates a Cost and Usage budget.

```
aws budgets create-budget \  
  --account-id 111122223333 \  
  --budget file://budget.json \  
  --notifications-with-subscribers file://notifications-with-subscribers.json
```

Contents of budget.json:

```
{  
  "BudgetLimit": {  
    "Amount": "100",  
    "Unit": "USD"  
  },  
  "BudgetName": "Example Tag Budget",  
  "BudgetType": "COST",  
  "CostFilters": {  
    "TagKeyValue": [  
      "user:Key$value1",  
      "user:Key$value2"  
    ]  
  },  
  "CostTypes": {  
    "IncludeCredit": true,  
    "IncludeDiscount": true,  
    "IncludeOtherSubscription": true,  
    "IncludeRecurring": true,  
    "IncludeRefund": true,  
    "IncludeSubscription": true,  
    "IncludeSupport": true,
```

```
    "IncludeTax": true,
    "IncludeUpfront": true,
    "UseBlended": false
  },
  "TimePeriod": {
    "Start": 1477958399,
    "End": 3706473600
  },
  "TimeUnit": "MONTHLY"
}
```

Contents of notifications-with-subscribers.json:

```
[
  {
    "Notification": {
      "ComparisonOperator": "GREATER_THAN",
      "NotificationType": "ACTUAL",
      "Threshold": 80,
      "ThresholdType": "PERCENTAGE"
    },
    "Subscribers": [
      {
        "Address": "example@example.com",
        "SubscriptionType": "EMAIL"
      }
    ]
  }
]
```

- For API details, see [CreateBudget](#) in *AWS CLI Command Reference*.

create-notification

The following code example shows how to use create-notification.

AWS CLI

To create a notification for the specified Cost and Usage budget

This example creates a notification for the specified Cost and Usage budget.

Command:

```
aws budgets create-notification --account-id 111122223333 --budget-name "Example Budget" --notification NotificationType=ACTUAL,ComparisonOperator=GREATER_THAN,Threshold=80,ThresholdType=PERCENTAGE --subscriber SubscriptionType=EMAIL,Address=example@example.com
```

- For API details, see [CreateNotification](#) in *AWS CLI Command Reference*.

create-subscriber

The following code example shows how to use `create-subscriber`.

AWS CLI

To create a subscriber for a notification associated with a Cost and Usage budget

This example creates a subscriber for the specified notification.

Command:

```
aws budgets create-subscriber --account-id 111122223333 --budget-name "Example Budget" --notification NotificationType=ACTUAL,ComparisonOperator=GREATER_THAN,Threshold=80,ThresholdType=PERCENTAGE --subscriber SubscriptionType=EMAIL,Address=example@example.com
```

- For API details, see [CreateSubscriber](#) in *AWS CLI Command Reference*.

delete-budget

The following code example shows how to use `delete-budget`.

AWS CLI

To delete a Cost and Usage budget

This example deletes the specified Cost and Usage budget.

Command:

```
aws budgets delete-budget --account-id 111122223333 --budget-name "Example Budget"
```

- For API details, see [DeleteBudget](#) in *AWS CLI Command Reference*.

delete-notification

The following code example shows how to use delete-notification.

AWS CLI

To delete a notification from a budget

This example deletes the specified notification from the specified budget.

Command:

```
aws budgets delete-notification --account-id 111122223333 --budget-name "Example Budget" --notification NotificationType=ACTUAL,ComparisonOperator=GREATER_THAN,Threshold=80,ThresholdType=PERCENTAGE
```

- For API details, see [DeleteNotification](#) in *AWS CLI Command Reference*.

delete-subscriber

The following code example shows how to use delete-subscriber.

AWS CLI

To delete a subscriber from a notification

This example deletes the specified subscriber from the specified notification.

Command:

```
aws budgets delete-subscriber --account-id 111122223333 --budget-name "Example Budget" --notification NotificationType=ACTUAL,ComparisonOperator=GREATER_THAN,Threshold=80,ThresholdType=PERCENTAGE --subscriber SubscriptionType=EMAIL,Address=example@example.com
```

- For API details, see [DeleteSubscriber](#) in *AWS CLI Command Reference*.

describe-budget

The following code example shows how to use describe-budget.

AWS CLI

To retrieve a budget associated with an account

This example retrieves the specified Cost and Usage budget.

Command:

```
aws budgets describe-budget --account-id 111122223333 --budget-name "Example Budget"
```

Output:

```
{
  "Budget": {
    "CalculatedSpend": {
      "ForecastedSpend": {
        "Amount": "2641.548000000000022919266484677791595458984375",
        "Unit": "USD"
      },
      "ActualSpend": {
        "Amount": "604.45600000000000172803993336856365203857421875",
        "Unit": "USD"
      }
    },
    "BudgetType": "COST",
    "BudgetLimit": {
      "Amount": "100",
      "Unit": "USD"
    },
    "BudgetName": "Example Budget",
    "CostTypes": {
      "IncludeOtherSubscription": true,
      "IncludeUpfront": true,
      "IncludeRefund": true,
      "UseBlended": false,
      "IncludeDiscount": true,
      "UseAmortized": false,
      "IncludeTax": true,
      "IncludeCredit": true,
      "IncludeSupport": true,
      "IncludeRecurring": true,
      "IncludeSubscription": true
    }
  },
}
```

```
    "TimeUnit": "MONTHLY",
    "TimePeriod": {
      "Start": 1477958399.0,
      "End": 3706473600.0
    },
    "CostFilters": {
      "AZ": [
        "us-east-1"
      ]
    }
  }
}
```

- For API details, see [DescribeBudget](#) in *AWS CLI Command Reference*.

describe-budgets

The following code example shows how to use describe-budgets.

AWS CLI

To retrieve the budgets associated with an account

This example retrieves the Cost and Usage budgets for an account.

Command:

```
aws budgets describe-budgets --account-id 111122223333 --max-results 20
```

Output:

```
{
  "Budgets": [
    {
      "CalculatedSpend": {
        "ForecastedSpend": {
          "Amount": "2641.548000000000022919266484677791595458984375",
          "Unit": "USD"
        },
      },
      "ActualSpend": {
        "Amount": "604.45600000000000172803993336856365203857421875",
        "Unit": "USD"
      }
    }
  ]
}
```

```
    },
    "BudgetType": "COST",
    "BudgetLimit": {
      "Amount": "100",
      "Unit": "USD"
    },
    },
    "BudgetName": "Example Budget",
    "CostTypes": {
      "IncludeOtherSubscription": true,
      "IncludeUpfront": true,
      "IncludeRefund": true,
      "UseBlended": false,
      "IncludeDiscount": true,
      "UseAmortized": false,
      "IncludeTax": true,
      "IncludeCredit": true,
      "IncludeSupport": true,
      "IncludeRecurring": true,
      "IncludeSubscription": true
    },
    },
    "TimeUnit": "MONTHLY",
    "TimePeriod": {
      "Start": 1477958399.0,
      "End": 3706473600.0
    },
    },
    "CostFilters": {
      "AZ": [
        "us-east-1"
      ]
    }
  }
]
}
```

- For API details, see [DescribeBudgets](#) in *AWS CLI Command Reference*.

describe-notifications-for-budget

The following code example shows how to use `describe-notifications-for-budget`.

AWS CLI

To retrieve the notifications for a budget

This example retrieves the notifications for a Cost and Usage budget.

Command:

```
aws budgets describe-notifications-for-budget --account-id 111122223333 --budget-name "Example Budget" --max-results 5
```

Output:

```
{
  "Notifications": [
    {
      "Threshold": 80.0,
      "ComparisonOperator": "GREATER_THAN",
      "NotificationType": "ACTUAL"
    }
  ]
}
```

- For API details, see [DescribeNotificationsForBudget](#) in *AWS CLI Command Reference*.

describe-subscribers-for-notification

The following code example shows how to use `describe-subscribers-for-notification`.

AWS CLI

To retrieve the subscribers for a budget notification

This example retrieves the subscribers for a Cost and Usage budget notification.

Command:

```
aws budgets describe-subscribers-for-notification --account-id 111122223333 --budget-name "Example Budget" --notification NotificationType=ACTUAL,ComparisonOperator=GREATER_THAN,Threshold=80,ThresholdType=PERCENTAGE --max-results 5
```

Output:

```
{
  "Subscribers": [
```



```
{
  "SubscriptionType": "EMAIL",
  "Address": "example2@example.com"
},
{
  "SubscriptionType": "EMAIL",
  "Address": "example@example.com"
}
]
```

- For API details, see [DescribeSubscribersForNotification](#) in *AWS CLI Command Reference*.

update-budget

The following code example shows how to use `update-budget`.

AWS CLI

To replace a budget for a Cost and Usage budget

This example replaces a Cost and Usage budget with a new budget.

Command:

```
aws budgets update-budget --account-id 111122223333 --new-budget file://new-budget.json
```

`new-budget.json`:

```
{
  "BudgetLimit": {
    "Amount": "100",
    "Unit": "USD"
  },
  "BudgetName": "Example Budget",
  "BudgetType": "COST",
  "CostFilters": {
    "AZ" : [ "us-east-1" ]
  },
  "CostTypes": {
    "IncludeCredit": false,
    "IncludeDiscount": true,
  }
}
```

```
    "IncludeOtherSubscription": true,  
    "IncludeRecurring": true,  
    "IncludeRefund": true,  
    "IncludeSubscription": true,  
    "IncludeSupport": true,  
    "IncludeTax": true,  
    "IncludeUpfront": true,  
    "UseBlended": false,  
    "UseAmortized": true  
  },  
  "TimePeriod": {  
    "Start": 1477958399,  
    "End": 3706473600  
  },  
  "TimeUnit": "MONTHLY"  
}
```

- For API details, see [UpdateBudget](#) in *AWS CLI Command Reference*.

update-notification

The following code example shows how to use update-notification.

AWS CLI

To replace a notification for a Cost and Usage budget

This example replaces an 80% notification for a Cost and Usage budget with a 90% notification.

Command:

```
aws budgets update-notification --account-id 111122223333 --budget-name "Example  
Budget" --old-notification  
NotificationType=ACTUAL,ComparisonOperator=GREATER_THAN,Threshold=80,ThresholdType=PERCENTA  
--new-notification  
NotificationType=ACTUAL,ComparisonOperator=GREATER_THAN,Threshold=90,ThresholdType=PERCENTA
```

- For API details, see [UpdateNotification](#) in *AWS CLI Command Reference*.

update-subscriber

The following code example shows how to use update-subscriber.

AWS CLI

To replace a subscriber for a Cost and Usage budget

This example replaces the subscriber for a Cost and Usage budget.

Command:

```
aws budgets update-subscriber --account-id 111122223333 --budget-name "Example Budget" --notification NotificationType=ACTUAL,ComparisonOperator=GREATER_THAN,Threshold=80,ThresholdType=PERCENTAGE --old-subscriber SubscriptionType=EMAIL,Address=example@example.com --new-subscriber SubscriptionType=EMAIL,Address=example2@example.com
```

- For API details, see [UpdateSubscriber](#) in *AWS CLI Command Reference*.

Amazon Chime examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon Chime.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-phone-number-with-user

The following code example shows how to use `associate-phone-number-with-user`.

AWS CLI

To associate a phone number with a user

The following `associate-phone-number-with-user` example associates the specified phone number with a user.

```
aws chime associate-phone-number-with-user \  
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \  
  --user-id 1ab2345c-67de-8901-f23g-45h678901j2k \  
  --e164-phone-number "+12065550100"
```

This command produces no output.

For more information, see [Managing User Phone Numbers](#) in the *Amazon Chime Administration Guide*.

- For API details, see [AssociatePhoneNumberWithUser](#) in *AWS CLI Command Reference*.

associate-phone-numbers-with-voice-connector-group

The following code example shows how to use `associate-phone-numbers-with-voice-connector-group`.

AWS CLI

To associate phone numbers with an Amazon Chime Voice Connector group

The following `associate-phone-numbers-with-voice-connector-group` example associates the specified phone numbers with an Amazon Chime Voice Connector group.

```
aws chime associate-phone-numbers-with-voice-connector-group \  
  --voice-connector-group-id 123a456b-c7d8-90e1-fg23-4h567jkl8901 \  
  --e164-phone-numbers "+12065550100" "+12065550101" \  
  --force-associate
```

Output:

```
{  
  "PhoneNumberErrors": []
```

```
}
```

For more information, see [Working with Amazon Chime Voice Connector groups](#) in the *Amazon Chime Administration Guide*.

- For API details, see [AssociatePhoneNumbersWithVoiceConnectorGroup](#) in *AWS CLI Command Reference*.

associate-phone-numbers-with-voice-connector

The following code example shows how to use `associate-phone-numbers-with-voice-connector`.

AWS CLI

To associate phone numbers with an Amazon Chime Voice Connector

The following `associate-phone-numbers-with-voice-connector` example associates the specified phone numbers with an Amazon Chime Voice Connector.

```
aws chime associate-phone-numbers-with-voice-connector \  
  --voice-connector-id abcdef1ghij2klmno3pqr4 \  
  --e164-phone-numbers "+12065550100" "+12065550101" \  
  --force-associate
```

Output:

```
{  
  "PhoneNumberErrors": []  
}
```

For more information, see [Working with Amazon Chime Voice Connectors](#) in the *Amazon Chime Administration Guide*.

- For API details, see [AssociatePhoneNumbersWithVoiceConnector](#) in *AWS CLI Command Reference*.

associate-signin-delegate-groups-with-account

The following code example shows how to use `associate-signin-delegate-groups-with-account`.

AWS CLI

To associate sign-in delegate groups

The following `associate-signin-delegate-groups-with-account` example associates the specified sign-in delegate group with the specified Amazon Chime account.

```
aws chime associate-signin-delegate-groups-with-account \  
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \  
  --signin-delegate-groups GroupName=my_users
```

This command produces no output.

For more information, see [Managing User Access and Permissions](#) in the *Amazon Chime Administration Guide*.

- For API details, see [AssociateSigninDelegateGroupsWithAccount](#) in *AWS CLI Command Reference*.

batch-create-room-membership

The following code example shows how to use `batch-create-room-membership`.

AWS CLI

To create multiple room memberships

The following `batch-create-room-membership` example adds multiple users to a chat room as chat room members. It also assigns administrator and member roles to the users.

```
aws chime batch-create-room-membership \  
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \  
  --room-id abcd1e2d-3e45-6789-01f2-3g45h67i890j \  
  --membership-item-list "MemberId=1ab2345c-67de-8901-  
f23g-45h678901j2k,Role=Administrator" "MemberId=2ab2345c-67de-8901-  
f23g-45h678901j2k,Role=Member"
```

Output:

```
{  
  "ResponseMetadata": {
```

```
"RequestId": "169ba401-d886-475f-8b3f-e01eac6fadfb",
"HTTPStatusCode": 201,
"HTTPHeaders": {
  "x-amzn-requestid": "169ba401-d886-475f-8b3f-e01eac6fadfb",
  "content-type": "application/json",
  "content-length": "13",
  "date": "Mon, 02 Dec 2019 22:46:58 GMT",
  "connection": "keep-alive"
},
"RetryAttempts": 0
},
"Errors": []
}
```

For more information, see [Creating a Chat Room](#) in the *Amazon Chime User Guide*.

- For API details, see [BatchCreateRoomMembership](#) in *AWS CLI Command Reference*.

batch-delete-phone-number

The following code example shows how to use `batch-delete-phone-number`.

AWS CLI

To delete multiple phone numbers

The following `batch-delete-phone-number` example deletes all of the specified phone numbers.

```
aws chime batch-delete-phone-number \
  --phone-number-ids "%2B12065550100" "%2B12065550101"
```

This command produces no output. Output:

```
{
  "PhoneNumberErrors": []
}
```

For more information, see [Working with Phone Numbers](#) in the *Amazon Chime Administration Guide*.

- For API details, see [BatchDeletePhoneNumber](#) in *AWS CLI Command Reference*.

batch-suspend-user

The following code example shows how to use `batch-suspend-user`.

AWS CLI

To suspend multiple users

The following `batch-suspend-user` example suspends the listed users from the specified Amazon Chime account.

```
aws chime batch-suspend-user \  
  --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \  
  --user-id-list "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE" "a1b2c3d4-5678-90ab-  
cdef-33333EXAMPLE" "a1b2c3d4-5678-90ab-cdef-44444EXAMPLE"
```

Output:

```
{  
  "UserErrors": []  
}
```

- For API details, see [BatchSuspendUser](#) in *AWS CLI Command Reference*.

batch-unsuspend-user

The following code example shows how to use `batch-unsuspend-user`.

AWS CLI

To unsuspend multiple users

The following `batch-unsuspend-user` example removes any previous suspension for the listed users on the specified Amazon Chime account.

```
aws chime batch-unsuspend-user \  
  --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \  
  --user-id-list "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE" "a1b2c3d4-5678-90ab-  
cdef-33333EXAMPLE" "a1b2c3d4-5678-90ab-cdef-44444EXAMPLE"
```

Output:


```
{
  "UserErrors": []
}
```

- For API details, see [BatchUnsuspendUser](#) in *AWS CLI Command Reference*.

batch-update-phone-number

The following code example shows how to use `batch-update-phone-number`.

AWS CLI

To update several phone number product types at the same time

The following `batch-update-phone-number` example updates the product types for all of the specified phone numbers.

```
aws chime batch-update-phone-number \
  --update-phone-number-request-items PhoneNumberId=
%2B12065550100,ProductType=BusinessCalling PhoneNumberId=
%2B12065550101,ProductType=BusinessCalling
```

Output:

```
{
  "PhoneNumberErrors": []
}
```

To update several phone number calling names at the same time

The following `batch-update-phone-number` example updates the calling names for all of the specified phone numbers.

```
aws chime batch-update-phone-number \
  --update-phone-number-request-items PhoneNumberId=
%2B14013143874,CallingName=phonenumber1 PhoneNumberId=
%2B14013144061,CallingName=phonenumber2
```

Output:

```
{
  "PhoneNumberErrors": []
}
```

For more information, see [Working with Phone Numbers](#) in the *Amazon Chime Administration Guide*.

- For API details, see [BatchUpdatePhoneNumber](#) in *AWS CLI Command Reference*.

batch-update-user

The following code example shows how to use `batch-update-user`.

AWS CLI

To update multiple users in a single command

The following `batch-update-user` example updates the `LicenseType` for each of the listed users in the specified Amazon Chime account.

```
aws chime batch-update-user \
  --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
  --update-user-request-items "UserId=a1b2c3d4-5678-90ab-
cdef-22222EXAMPLE,LicenseType=Basic" "UserId=a1b2c3d4-5678-90ab-
cdef-33333EXAMPLE,LicenseType=Basic"
```

Output:

```
{
  "UserErrors": []
}
```

- For API details, see [BatchUpdateUser](#) in *AWS CLI Command Reference*.

create-account

The following code example shows how to use `create-account`.

AWS CLI

To create an account

The following `create-account` example creates an Amazon Chime account under the administrator's AWS account.

```
aws chime create-account \  
  --name MyChimeAccount
```

Output:

```
{  
  "Account": {  
    "AwsAccountId": "111122223333",  
    "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
    "Name": "MyChimeAccount",  
    "AccountType": "Team",  
    "CreatedTimestamp": "2019-01-04T17:11:22.003Z",  
    "DefaultLicense": "Pro",  
    "SupportedLicenses": [  
      "Basic",  
      "Pro"  
    ],  
    "SigninDelegateGroups": [  
      {  
        "GroupName": "myGroup"  
      },  
    ]  
  }  
}
```

For more information, see [Getting Started](#) in the *Amazon Chime Administration Guide*.

- For API details, see [CreateAccount](#) in *AWS CLI Command Reference*.

create-bot

The following code example shows how to use `create-bot`.

AWS CLI

To create an Amazon Chime bot

The following `create-bot` example creates a bot for the specified Amazon Chime Enterprise account.

```
aws chime create-bot \  
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \  
  --display-name "myBot" \  
  --domain "example.com"
```

Output:

```
{  
  "Bot": {  
    "BotId": "123abcd4-5ef6-789g-0h12-34j56789012k",  
    "UserId": "123abcd4-5ef6-789g-0h12-34j56789012k",  
    "DisplayName": "myBot (Bot)",  
    "BotType": "ChatBot",  
    "Disabled": false,  
    "CreatedTimestamp": "2019-09-09T18:05:56.749Z",  
    "UpdatedTimestamp": "2019-09-09T18:05:56.749Z",  
    "BotEmail": "myBot-chimebot@example.com",  
    "SecurityToken": "wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY"  
  }  
}
```

For more information, see [Integrate a Chat Bot with Amazon Chime](#) in the *Amazon Chime Developer Guide*.

- For API details, see [CreateBot](#) in *AWS CLI Command Reference*.

create-phone-number-order

The following code example shows how to use `create-phone-number-order`.

AWS CLI

To create a phone number order

The following `create-phone-number-order` example creates a phone number order for the specified phone numbers.

```
aws chime create-phone-number-order \  
  --product-type VoiceConnector \  
  --e164-phone-numbers "+12065550100" "+12065550101" "+12065550102"
```

Output:

```
{
  "PhoneNumberOrder": {
    "PhoneNumberOrderId": "abc12345-de67-89f0-123g-h45i678j9012",
    "ProductType": "VoiceConnector",
    "Status": "Processing",
    "OrderedPhoneNumbers": [
      {
        "E164PhoneNumber": "+12065550100",
        "Status": "Processing"
      },
      {
        "E164PhoneNumber": "+12065550101",
        "Status": "Processing"
      },
      {
        "E164PhoneNumber": "+12065550102",
        "Status": "Processing"
      }
    ],
    "CreatedTimestamp": "2019-08-09T21:35:21.427Z",
    "UpdatedTimestamp": "2019-08-09T21:35:22.408Z"
  }
}
```

For more information, see [Working with Phone Numbers](#) in the *Amazon Chime Administration Guide*.

- For API details, see [CreatePhoneNumberOrder](#) in *AWS CLI Command Reference*.

create-proxy-session

The following code example shows how to use `create-proxy-session`.

AWS CLI

To create a proxy session

The following `create-proxy-session` example creates a proxy session with voice and SMS capabilities.

```
aws chime create-proxy-session \
  --voice-connector-id abcdef1ghij2klmno3pqr4 \
  --participant-phone-numbers "+14015550101" "+12065550100" \
```

```
--capabilities "Voice" "SMS"
```

Output:

```
{
  "ProxySession": {
    "VoiceConnectorId": "abcdef1ghij2klmno3pqr4",
    "ProxySessionId": "123a4bc5-67d8-901e-2f3g-h4ghjk567891",
    "Status": "Open",
    "ExpiryMinutes": 60,
    "Capabilities": [
      "SMS",
      "Voice"
    ],
    "CreatedTimestamp": "2020-04-15T16:10:10.288Z",
    "UpdatedTimestamp": "2020-04-15T16:10:10.288Z",
    "Participants": [
      {
        "PhoneNumber": "+12065550100",
        "ProxyPhoneNumber": "+19135550199"
      },
      {
        "PhoneNumber": "+14015550101",
        "ProxyPhoneNumber": "+19135550199"
      }
    ]
  }
}
```

For more information, see [Proxy Phone Sessions](#) in the *Amazon Chime Developer Guide*.

- For API details, see [CreateProxySession](#) in *AWS CLI Command Reference*.

create-room-membership

The following code example shows how to use `create-room-membership`.

AWS CLI

To create a room membership

The following `create-room-membership` example adds the specified user to the chat room as a chat room member.

```
aws chime create-room-membership \  
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \  
  --room-id abcd1e2d-3e45-6789-01f2-3g45h67i890j \  
  --member-id 1ab2345c-67de-8901-f23g-45h678901j2k
```

Output:

```
{  
  "RoomMembership": {  
    "RoomId": "abcd1e2d-3e45-6789-01f2-3g45h67i890j",  
    "Member": {  
      "MemberId": "1ab2345c-67de-8901-f23g-45h678901j2k",  
      "MemberType": "User",  
      "Email": "janed@example.com",  
      "FullName": "Jane Doe",  
      "AccountId": "12a3456b-7c89-012d-3456-78901e23fg45"  
    },  
    "Role": "Member",  
    "InvitedBy": "arn:aws:iam::111122223333:user/alejandro",  
    "UpdatedTimestamp": "2019-12-02T22:36:41.969Z"  
  }  
}
```

For more information, see [Creating a Chat Room](#) in the *Amazon Chime User Guide*.

- For API details, see [CreateRoomMembership](#) in *AWS CLI Command Reference*.

create-room

The following code example shows how to use `create-room`.

AWS CLI

To create a chat room

The following `create-room` example creates a chat room for the specified Amazon Chime account.

```
aws chime create-room \  
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \  
  --name chatRoom
```

Output:

```
{
  "Room": {
    "RoomId": "abcd1e2d-3e45-6789-01f2-3g45h67i890j",
    "Name": "chatRoom",
    "AccountId": "12a3456b-7c89-012d-3456-78901e23fg45",
    "CreatedBy": "arn:aws:iam::111122223333:user/alejandro",
    "CreatedTimestamp": "2019-12-02T22:29:31.549Z",
    "UpdatedTimestamp": "2019-12-02T22:29:31.549Z"
  }
}
```

For more information, see [Creating a Chat Room](#) in the *Amazon Chime User Guide*.

- For API details, see [CreateRoom](#) in *AWS CLI Command Reference*.

create-user

The following code example shows how to use `create-user`.

AWS CLI**To create a user profile for a shared device**

The following `create-user` example creates a shared device profile for the specified email address.

```
aws chime create-user \
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
  --email roomdevice@example.com \
  --user-type SharedDevice
```

Output:

```
{
  "User": {
    "UserId": "1ab2345c-67de-8901-f23g-45h678901j2k",
    "AccountId": "12a3456b-7c89-012d-3456-78901e23fg45",
    "PrimaryEmail": "roomdevice@example.com",
    "DisplayName": "Room Device",
    "LicenseType": "Pro",
  }
}
```



```
    "UserType": "SharedDevice",
    "UserRegistrationStatus": "Registered",
    "RegisteredOn": "2020-01-15T22:38:09.806Z",
    "AlexaForBusinessMetadata": {
      "IsAlexaForBusinessEnabled": false
    }
  }
}
```

For more information, see [Preparing for Setup](#) in the *Amazon Chime Administration Guide*.

- For API details, see [CreateUser](#) in *AWS CLI Command Reference*.

create-voice-connector-group

The following code example shows how to use `create-voice-connector-group`.

AWS CLI

To create an Amazon Chime Voice Connector group

The following `create-voice-connector-group` example creates an Amazon Chime Voice Connector group that includes the specified Amazon Chime Voice Connector.

```
aws chime create-voice-connector-group \
  --name myGroup \
  --voice-connector-items VoiceConnectorId=abcdefghijklmno3pqr4,Priority=2
```

Output:

```
{
  "VoiceConnectorGroup": {
    "VoiceConnectorGroupId": "123a456b-c7d8-90e1-fg23-4h567jk18901",
    "Name": "myGroup",
    "VoiceConnectorItems": [],
    "CreatedTimestamp": "2019-09-18T16:38:34.734Z",
    "UpdatedTimestamp": "2019-09-18T16:38:34.734Z"
  }
}
```

For more information, see [Working with Amazon Chime Voice Connector Groups](#) in the *Amazon Chime Administration Guide*.

- For API details, see [CreateVoiceConnectorGroup](#) in *AWS CLI Command Reference*.

create-voice-connector

The following code example shows how to use `create-voice-connector`.

AWS CLI

To create an Amazon Chime Voice Connector

The following `create-voice-connector` example creates an Amazon Chime Voice Connector in the specified AWS Region, with encryption enabled.

```
aws chime create-voice-connector \
  --name newVoiceConnector \
  --aws-region us-west-2 \
  --require-encryption
```

Output:

```
{
  "VoiceConnector": {
    "VoiceConnectorId": "abcdef1ghij2klmno3pqr4",
    "AwsRegion": "us-west-2",
    "Name": "newVoiceConnector",
    "OutboundHostName": "abcdef1ghij2klmno3pqr4.voiceconnector.chime.aws",
    "RequireEncryption": true,
    "CreatedTimestamp": "2019-09-18T20:34:01.352Z",
    "UpdatedTimestamp": "2019-09-18T20:34:01.352Z"
  }
}
```

For more information, see [Working with Amazon Chime Voice Connectors](#) in the *Amazon Chime Administration Guide*.

- For API details, see [CreateVoiceConnector](#) in *AWS CLI Command Reference*.

delete-account

The following code example shows how to use `delete-account`.

AWS CLI

To delete an account

The following `delete-account` example deletes the specified account.

```
aws chime delete-account --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
```

This command produces no output.

For more information, see [Deleting Your Account](#) in the *Amazon Chime Administration Guide*.

- For API details, see [DeleteAccount](#) in *AWS CLI Command Reference*.

`delete-phone-number`

The following code example shows how to use `delete-phone-number`.

AWS CLI

To delete a phone number

The following `delete-phone-number` example moves the specified phone number into the deletion queue.

```
aws chime delete-phone-number \  
  --phone-number-id "+12065550100"
```

This command produces no output.

For more information, see [Working with Phone Numbers](#) in the *Amazon Chime Administration Guide*.

- For API details, see [DeletePhoneNumber](#) in *AWS CLI Command Reference*.

`delete-proxy-session`

The following code example shows how to use `delete-proxy-session`.

AWS CLI

To delete a proxy session

The following `delete-proxy-session` example deletes the specified proxy session.

```
aws chime delete-proxy-session \  
  --voice-connector-id abcdef1ghij2klmno3pqr4 \  
  --proxy-session-id 123a4bc5-67d8-901e-2f3g-h4ghjk567891
```

This command produces no output.

For more information, see [Proxy Phone Sessions](#) in the *Amazon Chime Developer Guide*.

- For API details, see [DeleteProxySession](#) in *AWS CLI Command Reference*.

delete-room-membership

The following code example shows how to use `delete-room-membership`.

AWS CLI

To remove a user as a member of a chat room

The following `delete-room-membership` example removes the specified member from the specified chat room.

```
aws chime delete-room-membership \  
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \  
  --room-id abcd1e2d-3e45-6789-01f2-3g45h67i890j \  
  --member-id 1ab2345c-67de-8901-f23g-45h678901j2k
```

This command produces no output.

For more information, see [Creating a Chat Room](#) in the *Amazon Chime User Guide*.

- For API details, see [DeleteRoomMembership](#) in *AWS CLI Command Reference*.

delete-room

The following code example shows how to use `delete-room`.

AWS CLI

To delete a chat room

The following `delete-room` example deletes the specified chat room and removes the chat room memberships.

```
aws chime delete-room \  
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \  
  --room-id abcd1e2d-3e45-6789-01f2-3g45h67i890j
```

This command produces no output.

For more information, see [Creating a Chat Room](#) in the *Amazon Chime User Guide*.

- For API details, see [DeleteRoom](#) in *AWS CLI Command Reference*.

delete-voice-connector-group

The following code example shows how to use `delete-voice-connector-group`.

AWS CLI

title

The following `delete-voice-connector-group` example deletes the specified Amazon Chime Voice Connector group.

```
aws chime delete-voice-connector-group \  
  --voice-connector-group-id 123a456b-c7d8-90e1-fg23-4h567jkl18901
```

This command produces no output.

For more information, see [Working with Amazon Chime Voice Connector Groups](#) in the *Amazon Chime Administration Guide*.

- For API details, see [DeleteVoiceConnectorGroup](#) in *AWS CLI Command Reference*.

delete-voice-connector-origination

The following code example shows how to use `delete-voice-connector-origination`.

AWS CLI

To delete origination settings

The following `delete-voice-connector-origination` example deletes the origination host, port, protocol, priority, and weight from the specified Amazon Chime Voice Connector.

```
aws chime delete-voice-connector-origination \  
  --voice-connector-id abcdef1ghij2klmno3pqr4
```

This command produces no output.

For more information, see [Working with Amazon Chime Voice Connectors](#) in the *Amazon Chime Administration Guide*.

- For API details, see [DeleteVoiceConnectorOrigination](#) in *AWS CLI Command Reference*.

delete-voice-connector-proxy

The following code example shows how to use `delete-voice-connector-proxy`.

AWS CLI

To delete a proxy configuration

The following `delete-voice-connector-proxy` example deletes the proxy configuration from your Amazon Chime Voice Connector.

```
aws chime delete-voice-connector-proxy \  
  --voice-connector-id abcdef1ghij2klmno3pqr4
```

This command produces no output.

For more information, see [Proxy Phone Sessions](#) in the *Amazon Chime Developer Guide*.

- For API details, see [DeleteVoiceConnectorProxy](#) in *AWS CLI Command Reference*.

delete-voice-connector-streaming-configuration

The following code example shows how to use `delete-voice-connector-streaming-configuration`.

AWS CLI

To delete a streaming configuration

The following `delete-voice-connector-streaming-configuration` example deletes the streaming configuration for the specified Amazon Chime Voice Connector.

```
aws chime delete-voice-connector-streaming-configuration \  
  --voice-connector-id abcdef1ghij2klmno3pqr4
```

This command produces no output.

For more information, see [Streaming Amazon Chime Voice Connector Data to Kinesis](#) in the *Amazon Chime Administration Guide*.

- For API details, see [DeleteVoiceConnectorStreamingConfiguration](#) in *AWS CLI Command Reference*.

delete-voice-connector-termination-credentials

The following code example shows how to use `delete-voice-connector-termination-credentials`.

AWS CLI

To delete termination credentials

The following `delete-voice-connector-termination-credentials` example deletes the termination credentials for the specified user name and Amazon Chime Voice Connector.

```
aws chime delete-voice-connector-termination-credentials \  
  --voice-connector-id abcdef1ghij2klmno3pqr4 \  
  --usernames "jdoe"
```

This command produces no output.

For more information, see [Working with Amazon Chime Voice Connectors](#) in the *Amazon Chime Administration Guide*.

- For API details, see [DeleteVoiceConnectorTerminationCredentials](#) in *AWS CLI Command Reference*.

delete-voice-connector-termination

The following code example shows how to use `delete-voice-connector-termination`.

AWS CLI

To delete termination settings

The following `delete-voice-connector-termination` example deletes the termination settings for the specified Amazon Chime Voice Connector.

```
aws chime delete-voice-connector-termination \  
  --voice-connector-id abcdef1ghij2klmno3pqr4
```

This command produces no output.

For more information, see [Working with Amazon Chime Voice Connectors](#) in the *Amazon Chime Administration Guide*.

- For API details, see [DeleteVoiceConnectorTermination](#) in *AWS CLI Command Reference*.

`delete-voice-connector`

The following code example shows how to use `delete-voice-connector`.

AWS CLI

To delete an Amazon Chime Voice Connector

The following `delete-voice-connector` example does this

```
aws chime delete-voice-connector \  
  --voice-connector-id abcdef1ghij2klmno3pqr4
```

This command produces no output.

For more information, see [Working with Amazon Chime Voice Connectors](#) in the *Amazon Chime Administration Guide*.

- For API details, see [DeleteVoiceConnector](#) in *AWS CLI Command Reference*.

`disassociate-phone-number-from-user`

The following code example shows how to use `disassociate-phone-number-from-user`.

AWS CLI

To disassociate a phone number from a user

The following `disassociate-phone-number-from-user` example disassociates a phone number from the specified user.

```
aws chime disassociate-phone-number-from-user \  
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \  
  --user-id 1ab2345c-67de-8901-f23g-45h678901j2k
```

This command produces no output.

For more information, see [Managing User Phone Numbers](#) in the *Amazon Chime Administration Guide*.

- For API details, see [DisassociatePhoneNumberFromUser](#) in *AWS CLI Command Reference*.

`disassociate-phone-numbers-from-voice-connector-group`

The following code example shows how to use `disassociate-phone-numbers-from-voice-connector-group`.

AWS CLI

To disassociate phone numbers from an Amazon Chime Voice Connector group

The following `disassociate-phone-numbers-from-voice-connector-group` example disassociates the specified phone numbers from an Amazon Chime Voice Connector group.

```
aws chime disassociate-phone-numbers-from-voice-connector-group \  
  --voice-connector-group-id 123a456b-c7d8-90e1-fg23-4h567jkl8901 \  
  --e164-phone-numbers "+12065550100" "+12065550101"
```

Output:

```
{  
  "PhoneNumberErrors": []  
}
```

For more information, see [Working with Amazon Chime Voice Connector Groups](#) in the *Amazon Chime Administration Guide*.

- For API details, see [DisassociatePhoneNumbersFromVoiceConnectorGroup](#) in *AWS CLI Command Reference*.

disassociate-phone-numbers-from-voice-connector

The following code example shows how to use `disassociate-phone-numbers-from-voice-connector`.

AWS CLI

To disassociate phone numbers from an Amazon Chime Voice Connector

The following `disassociate-phone-numbers-from-voice-connector` example disassociates the specified phone numbers from an Amazon Chime Voice Connector.

```
aws chime disassociate-phone-numbers-from-voice-connector \  
  --voice-connector-id abcdef1ghij2klmno3pqr4 \  
  --e164-phone-numbers "+12065550100" "+12065550101"
```

Output:

```
{  
  "PhoneNumberErrors": []  
}
```

For more information, see [Working with Amazon Chime Voice Connectors](#) in the *Amazon Chime Administration Guide*.

- For API details, see [DisassociatePhoneNumbersFromVoiceConnector](#) in *AWS CLI Command Reference*.

disassociate-signin-delegate-groups-from-account

The following code example shows how to use `disassociate-signin-delegate-groups-from-account`.

AWS CLI

To disassociate sign-in delegate groups

The following `disassociate-signin-delegate-groups-from-account` example disassociates the specified sign-in delegate group from the specified Amazon Chime account.

```
aws chime disassociate-signin-delegate-groups-from-account \  
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \  
  --group-names "my_users"
```

This command produces no output.

For more information, see [Managing User Access and Permissions](#) in the *Amazon Chime Administration Guide*.

- For API details, see [DisassociateSigninDelegateGroupsFromAccount](#) in *AWS CLI Command Reference*.

get-account-settings

The following code example shows how to use `get-account-settings`.

AWS CLI

To retrieve settings for an account

The following `get-account-settings` example retrieves the account settings for the specified account.

```
aws chime get-account-settings --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
```

Output:

```
{  
  "AccountSettings": {  
    "DisableRemoteControl": false,  
    "EnableDialOut": false  
  }  
}
```

For more information, see [Managing Your Amazon Chime Accounts](#) in the *Amazon Chime Administration Guide*.

- For API details, see [GetAccountSettings](#) in *AWS CLI Command Reference*.

get-account

The following code example shows how to use `get-account`.

AWS CLI

To retrieve the details for an account

The following `get-account` example retrieves the details for the specified Amazon Chime account.

```
aws chime get-account \  
  --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
```

Output:

```
{  
  "Account": {  
    "AwsAccountId": "111122223333",  
    "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
    "Name": "EnterpriseDirectory",  
    "AccountType": "EnterpriseDirectory",  
    "CreatedTimestamp": "2018-12-20T18:38:02.181Z",  
    "DefaultLicense": "Pro",  
    "SupportedLicenses": [  
      "Basic",  
      "Pro"  
    ],  
    "SigninDelegateGroups": [  
      {  
        "GroupName": "myGroup"  
      },  
    ]  
  }  
}
```

For more information, see [Managing Your Amazon Chime Accounts](#) in the *Amazon Chime Administration Guide*.

- For API details, see [GetAccount](#) in *AWS CLI Command Reference*.

get-bot

The following code example shows how to use `get-bot`.

AWS CLI

To retrieve details about a bot

The following `get-bot` example displays the details for the specified bot.

```
aws chime get-bot \
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
  --bot-id 123abcd4-5ef6-789g-0h12-34j56789012k
```

Output:

```
{
  "Bot": {
    "BotId": "123abcd4-5ef6-789g-0h12-34j56789012k",
    "UserId": "123abcd4-5ef6-789g-0h12-34j56789012k",
    "DisplayName": "myBot (Bot)",
    "BotType": "ChatBot",
    "Disabled": false,
    "CreatedTimestamp": "2019-09-09T18:05:56.749Z",
    "UpdatedTimestamp": "2019-09-09T18:05:56.749Z",
    "BotEmail": "myBot-chimebot@example.com",
    "SecurityToken": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
  }
}
```

For more information, see [Update Chat Bots](#) in the *Amazon Chime Developer Guide*.

- For API details, see [GetBot](#) in *AWS CLI Command Reference*.

get-global-settings

The following code example shows how to use `get-global-settings`.

AWS CLI

To get global settings

The following `get-global-settings` example retrieves the S3 bucket names used to store call detail records for Amazon Chime Business Calling and Amazon Chime Voice Connectors associated with the administrator's AWS account.

```
aws chime get-global-settings
```

Output:

```
{
  "BusinessCalling": {
    "CdrBucket": "s3bucket"
  },
  "VoiceConnector": {
    "CdrBucket": "s3bucket"
  }
}
```

For more information, see [Managing Global Settings](#) in the *Amazon Chime Administration Guide*.

- For API details, see [GetGlobalSettings](#) in *AWS CLI Command Reference*.

get-phone-number-order

The following code example shows how to use `get-phone-number-order`.

AWS CLI

To get details for a phone number order

The following `get-phone-number-order` example displays the details of the specified phone number order.

```
aws chime get-phone-number-order \
  --phone-number-order-id abc12345-de67-89f0-123g-h45i678j9012
```

Output:

```
{
  "PhoneNumberOrder": {
    "PhoneNumberOrderId": "abc12345-de67-89f0-123g-h45i678j9012",
    "ProductType": "VoiceConnector",
```

```
    "Status": "Partial",
    "OrderedPhoneNumbers": [
      {
        "E164PhoneNumber": "+12065550100",
        "Status": "Acquired"
      },
      {
        "E164PhoneNumber": "+12065550101",
        "Status": "Acquired"
      },
      {
        "E164PhoneNumber": "+12065550102",
        "Status": "Failed"
      }
    ],
    "CreatedTimestamp": "2019-08-09T21:35:21.427Z",
    "UpdatedTimestamp": "2019-08-09T21:35:31.926Z"
  }
}
```

For more information, see [Working with Phone Numbers](#) in the *Amazon Chime Administration Guide*.

- For API details, see [GetPhoneNumberOrder](#) in *AWS CLI Command Reference*.

get-phone-number-settings

The following code example shows how to use `get-phone-number-settings`.

AWS CLI

To retrieve an outbound calling name

The following `get-phone-number-settings` example retrieves the default outbound calling name for the calling user's AWS account.

```
aws chime get-phone-number-settings
```

This command produces no output. Output:

```
{
  "CallingName": "myName",
  "CallingNameUpdatedTimestamp": "2019-10-28T18:56:42.911Z"
}
```

```
}
```

For more information, see [Working with Phone Numbers](#) in the *Amazon Chime Administration Guide*.

- For API details, see [GetPhoneNumberSettings](#) in *AWS CLI Command Reference*.

get-phone-number

The following code example shows how to use `get-phone-number`.

AWS CLI

To get phone number details

The following `get-phone-number` example displays the details of the specified phone number.

```
aws chime get-phone-number \
  --phone-number-id +12065550100
```

Output:

```
{
  "PhoneNumber": {
    "PhoneNumberId": "%2B12065550100",
    "E164PhoneNumber": "+12065550100",
    "Type": "Local",
    "ProductType": "VoiceConnector",
    "Status": "Unassigned",
    "Capabilities": {
      "InboundCall": true,
      "OutboundCall": true,
      "InboundSMS": true,
      "OutboundSMS": true,
      "InboundMMS": true,
      "OutboundMMS": true
    },
    "Associations": [
      {
        "Value": "abcdef1ghij2klmno3pqr4",
        "Name": "VoiceConnectorId",
        "AssociatedTimestamp": "2019-10-28T18:40:37.453Z"
      }
    ]
  }
}
```



```
    ],
    "CallingNameStatus": "UpdateInProgress",
    "CreatedTimestamp": "2019-08-09T21:35:21.445Z",
    "UpdatedTimestamp": "2019-08-09T21:35:31.745Z"
  }
}
```

For more information, see [Working with Phone Numbers](#) in the *Amazon Chime Administration Guide*.

- For API details, see [GetPhoneNumber](#) in *AWS CLI Command Reference*.

get-proxy-session

The following code example shows how to use `get-proxy-session`.

AWS CLI

To get proxy session details

The following `get-proxy-session` example lists the details of the specified proxy session.

```
aws chime get-proxy-session \
  --voice-connector-id abcdef1ghij2klmno3pqr4 \
  --proxy-session-id 123a4bc5-67d8-901e-2f3g-h4ghjk567891
```

Output:

```
{
  "ProxySession": {
    "VoiceConnectorId": "abcdef1ghij2klmno3pqr4",
    "ProxySessionId": "123a4bc5-67d8-901e-2f3g-h4ghjk567891",
    "Status": "Open",
    "ExpiryMinutes": 60,
    "Capabilities": [
      "SMS",
      "Voice"
    ],
    "CreatedTimestamp": "2020-04-15T16:10:10.288Z",
    "UpdatedTimestamp": "2020-04-15T16:10:10.288Z",
    "Participants": [
      {
        "PhoneNumber": "+12065550100",
```

```
        "ProxyPhoneNumber": "+19135550199"
      },
      {
        "PhoneNumber": "+14015550101",
        "ProxyPhoneNumber": "+19135550199"
      }
    ]
  }
}
```

For more information, see [Proxy Phone Sessions](#) in the *Amazon Chime Developer Guide*.

- For API details, see [GetProxySession](#) in *AWS CLI Command Reference*.

get-room

The following code example shows how to use `get-room`.

AWS CLI

To get the details about a chat room

The following `get-room` example displays details about the specified chat room.

```
aws chime get-room \
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
  --room-id abcd1e2d-3e45-6789-01f2-3g45h67i890j
```

Output:

```
{
  "Room": {
    "RoomId": "abcd1e2d-3e45-6789-01f2-3g45h67i890j",
    "Name": "chatRoom",
    "AccountId": "12a3456b-7c89-012d-3456-78901e23fg45",
    "CreatedBy": "arn:aws:iam::111122223333:user/alejandro",
    "CreatedTimestamp": "2019-12-02T22:29:31.549Z",
    "UpdatedTimestamp": "2019-12-02T22:29:31.549Z"
  }
}
```

For more information, see [Creating a Chat Room](#) in the *Amazon Chime User Guide*.

- For API details, see [GetRoom](#) in *AWS CLI Command Reference*.

get-user-settings

The following code example shows how to use `get-user-settings`.

AWS CLI

To retrieve user settings

The following `get-user-settings` example displays the specified user settings.

```
aws chime get-user-settings \  
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \  
  --user-id 1ab2345c-67de-8901-f23g-45h678901j2k
```

Output:

```
{  
  "UserSettings": {  
    "Telephony": {  
      "InboundCalling": true,  
      "OutboundCalling": true,  
      "SMS": true  
    }  
  }  
}
```

For more information, see [Managing User Phone Numbers](#) in the *Amazon Chime Administration Guide*.

- For API details, see [GetUserSettings](#) in *AWS CLI Command Reference*.

get-user

The following code example shows how to use `get-user`.

AWS CLI

To get details about a user

The following `get-user` example retrieves the details for the specified user.

```
aws chime get-user \  
  --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \  
  --user-id a1b2c3d4-5678-90ab-cdef-22222EXAMPLE
```

Output:

```
{  
  "User": {  
    "UserId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",  
    "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
    "PrimaryEmail": "marthar@example.com",  
    "DisplayName": "Martha Rivera",  
    "LicenseType": "Pro",  
    "UserRegistrationStatus": "Registered",  
    "RegisteredOn": "2018-12-20T18:45:25.231Z",  
    "InvitedOn": "2018-12-20T18:45:25.231Z",  
    "AlexaForBusinessMetadata": {  
      "IsAlexaForBusinessEnabled": False,  
      "AlexaForBusinessRoomArn": "null"  
    },  
    "PersonalPIN": "XXXXXXXXXX"  
  }  
}
```

For more information, see [Managing Users](#) in the *Amazon Chime Administration Guide*.

- For API details, see [GetUser](#) in *AWS CLI Command Reference*.

get-voice-connector-group

The following code example shows how to use `get-voice-connector-group`.

AWS CLI

To get details for an Amazon Chime Voice Connector group

The following `get-voice-connector-group` example displays details for the specified Amazon Chime Voice Connector group.

```
aws chime get-voice-connector-group \  
  --voice-connector-group-id 123a456b-c7d8-90e1-fg23-4h567jk18901
```

Output:

```
{
  "VoiceConnectorGroup": {
    "VoiceConnectorGroupId": "123a456b-c7d8-90e1-fg23-4h567jk18901",
    "Name": "myGroup",
    "VoiceConnectorItems": [],
    "CreatedTimestamp": "2019-09-18T16:38:34.734Z",
    "UpdatedTimestamp": "2019-09-18T16:38:34.734Z"
  }
}
```

For more information, see [Working with Amazon Chime Voice Connector Groups](#) in the *Amazon Chime Administration Guide*.

- For API details, see [GetVoiceConnectorGroup](#) in *AWS CLI Command Reference*.

get-voice-connector-logging-configuration

The following code example shows how to use `get-voice-connector-logging-configuration`.

AWS CLI**To get logging configuration details**

The following `get-voice-connector-logging-configuration` example retrieves the logging configuration details for the specified Amazon Chime Voice Connector.

```
aws chime get-voice-connector-logging-configuration \
  --voice-connector-id abcdef1ghij2klmno3pqr4
```

Output:

```
{
  "LoggingConfiguration": {
    "EnableSIPLogs": true
  }
}
```

For more information, see [Streaming Amazon Chime Voice Connector Media to Kinesis](#) in the *Amazon Chime Administration Guide*.

- For API details, see [GetVoiceConnectorLoggingConfiguration](#) in *AWS CLI Command Reference*.

get-voice-connector-origination

The following code example shows how to use `get-voice-connector-origination`.

AWS CLI

To retrieve origination settings

The following `get-voice-connector-origination` example retrieves the origination host, port, protocol, priority, and weight for the specified Amazon Chime Voice Connector.

```
aws chime get-voice-connector-origination \  
  --voice-connector-id abcdef1ghij2klmno3pqr4
```

Output:

```
{  
  "Origination": {  
    "Routes": [  
      {  
        "Host": "10.24.34.0",  
        "Port": 1234,  
        "Protocol": "TCP",  
        "Priority": 1,  
        "Weight": 5  
      }  
    ],  
    "Disabled": false  
  }  
}
```

For more information, see [Working with Amazon Chime Voice Connectors](#) in the *Amazon Chime Administration Guide*.

- For API details, see [GetVoiceConnectorOrigination](#) in *AWS CLI Command Reference*.

get-voice-connector-proxy

The following code example shows how to use `get-voice-connector-proxy`.

AWS CLI

To get proxy configuration details

The following `get-voice-connector-proxy` example gets the proxy configuration details for your Amazon Chime Voice Connector.

```
aws chime get-voice-connector-proxy \  
  --voice-connector-id abcdef1ghij2klmno3pqr4
```

Output:

```
{  
  "Proxy": {  
    "DefaultSessionExpiryMinutes": 60,  
    "Disabled": false,  
    "PhoneNumberCountries": [  
      "US"  
    ]  
  }  
}
```

For more information, see [Proxy Phone Sessions](#) in the *Amazon Chime Developer Guide*.

- For API details, see [GetVoiceConnectorProxy](#) in *AWS CLI Command Reference*.

get-voice-connector-streaming-configuration

The following code example shows how to use `get-voice-connector-streaming-configuration`.

AWS CLI

To get streaming configuration details

The following `get-voice-connector-streaming-configuration` example gets the streaming configuration details for the specified Amazon Chime Voice Connector.

```
aws chime get-voice-connector-streaming-configuration \  
  --voice-connector-id abcdef1ghij2klmno3pqr4
```

Output:

```
{
  "StreamingConfiguration": {
    "DataRetentionInHours": 24,
    "Disabled": false
  }
}
```

For more information, see [Streaming Amazon Chime Voice Connector Data to Kinesis](#) in the *Amazon Chime Administration Guide*.

- For API details, see [GetVoiceConnectorStreamingConfiguration](#) in *AWS CLI Command Reference*.

get-voice-connector-termination-health

The following code example shows how to use `get-voice-connector-termination-health`.

AWS CLI**To retrieve termination health details**

The following `get-voice-connector-termination-health` example retrieves the termination health details for the specified Amazon Chime Voice Connector.

```
aws chime get-voice-connector-termination-health \
  --voice-connector-id abcdef1ghij2klmno3pqr4
```

Output:

```
{
  "TerminationHealth": {
    "Timestamp": "Fri Aug 23 16:45:55 UTC 2019",
    "Source": "10.24.34.0"
  }
}
```

For more information, see [Working with Amazon Chime Voice Connectors](#) in the *Amazon Chime Administration Guide*.

- For API details, see [GetVoiceConnectorTerminationHealth](#) in *AWS CLI Command Reference*.

get-voice-connector-termination

The following code example shows how to use `get-voice-connector-termination`.

AWS CLI

To retrieve termination settings

The following `get-voice-connector-termination` example retrieves the termination settings for the specified Amazon Chime Voice Connector.

```
aws chime get-voice-connector-termination \  
  --voice-connector-id abcdef1ghij2klmno3pqr4
```

This command produces no output. Output:

```
{  
  "Termination": {  
    "CpsLimit": 1,  
    "DefaultPhoneNumber": "+12065550100",  
    "CallingRegions": [  
      "US"  
    ],  
    "CidrAllowedList": [  
      "10.24.34.0/23"  
    ],  
    "Disabled": false  
  }  
}
```

For more information, see [Working with Amazon Chime Voice Connectors](#) in the *Amazon Chime Administration Guide*.

- For API details, see [GetVoiceConnectorTermination](#) in *AWS CLI Command Reference*.

get-voice-connector

The following code example shows how to use `get-voice-connector`.

AWS CLI

To get details for an Amazon Chime Voice Connector

The following `get-voice-connector` example displays the details of the specified Amazon Chime Voice Connector.

```
aws chime get-voice-connector \  
  --voice-connector-id abcdef1ghij2klmno3pqr4
```

Output:

```
{  
  "VoiceConnector": {  
    "VoiceConnectorId": "abcdef1ghij2klmno3pqr4",  
    "AwsRegion": "us-west-2",  
    "Name": "newVoiceConnector",  
    "OutboundHostName": "abcdef1ghij2klmno3pqr4.voiceconnector.chime.aws",  
    "RequireEncryption": true,  
    "CreatedTimestamp": "2019-09-18T20:34:01.352Z",  
    "UpdatedTimestamp": "2019-09-18T20:34:01.352Z"  
  }  
}
```

For more information, see [Working with Amazon Chime Voice Connectors](#) in the *Amazon Chime Administration Guide*.

- For API details, see [GetVoiceConnector](#) in *AWS CLI Command Reference*.

invite-users

The following code example shows how to use `invite-users`.

AWS CLI

To invite users to join Amazon Chime

The following `invite-users` example sends an email to invite a user to the specified Amazon Chime account.

```
aws chime invite-users \  
  --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \  
  --user-email-list "alejandr@example.com" "janed@example.com"
```

Output:

```
{
  "Invites": [
    {
      "InviteId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
      "Status": "Pending",
      "EmailAddress": "alejandror@example.com",
      "EmailStatus": "Sent"
    }
    {
      "InviteId": "a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
      "Status": "Pending",
      "EmailAddress": "janed@example.com",
      "EmailStatus": "Sent"
    }
  ]
}
```

For more information, see [Inviting and Suspending Users](#) in the *Amazon Chime Administration Guide*.

- For API details, see [InviteUsers](#) in *AWS CLI Command Reference*.

list-accounts

The following code example shows how to use `list-accounts`.

AWS CLI

To get a list of accounts

The following `list-accounts` example retrieves a list of the Amazon Chime accounts in the administrator's AWS account.

```
aws chime list-accounts
```

Output:

```
{
  "Accounts": [
    {
      "AwsAccountId": "111122223333",
```

```

    "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "Name": "First Chime Account",
    "AccountType": "EnterpriseDirectory",
    "CreatedTimestamp": "2018-12-20T18:38:02.181Z",
    "DefaultLicense": "Pro",
    "SupportedLicenses": [
      "Basic",
      "Pro"
    ],
    "SigninDelegateGroups": [
      {
        "GroupName": "myGroup"
      }
    ]
  },
  {
    "AwsAccountId": "111122223333",
    "AccountId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
    "Name": "Second Chime Account",
    "AccountType": "Team",
    "CreatedTimestamp": "2018-09-04T21:44:22.292Z",
    "DefaultLicense": "Pro",
    "SupportedLicenses": [
      "Basic",
      "Pro"
    ],
    "SigninDelegateGroups": [
      {
        "GroupName": "myGroup"
      }
    ]
  }
]
}

```

For more information, see [Managing Your Amazon Chime Accounts](#) in the *Amazon Chime Administration Guide*.

- For API details, see [ListAccounts](#) in *AWS CLI Command Reference*.

list-bots

The following code example shows how to use `list-bots`.

AWS CLI

To retrieve a list of bots

The following `list-bots` example lists the bots associated with the specified Amazon Chime Enterprise account.

```
aws chime list-bots \  
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45
```

Output:

```
{  
  "Bot": {  
    "BotId": "123abcd4-5ef6-789g-0h12-34j56789012k",  
    "UserId": "123abcd4-5ef6-789g-0h12-34j56789012k",  
    "DisplayName": "myBot (Bot)",  
    "BotType": "ChatBot",  
    "Disabled": false,  
    "CreatedTimestamp": "2019-09-09T18:05:56.749Z",  
    "UpdatedTimestamp": "2019-09-09T18:05:56.749Z",  
    "BotEmail": "myBot-chimebot@example.com",  
    "SecurityToken": "wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY"  
  }  
}
```

For more information, see [Use Chat Bots with Amazon Chime](#) in the *Amazon Chime Developer Guide*.

- For API details, see [ListBots](#) in *AWS CLI Command Reference*.

list-phone-number-orders

The following code example shows how to use `list-phone-number-orders`.

AWS CLI

To list phone number orders

The following `list-phone-number-orders` example lists the phone number orders associated with the Amazon Chime administrator's account.

```
aws chime list-phone-number-orders
```

Output:

```
{
  "PhoneNumberOrders": [
    {
      "PhoneNumberOrderId": "abc12345-de67-89f0-123g-h45i678j9012",
      "ProductType": "VoiceConnector",
      "Status": "Partial",
      "OrderedPhoneNumbers": [
        {
          "E164PhoneNumber": "+12065550100",
          "Status": "Acquired"
        },
        {
          "E164PhoneNumber": "+12065550101",
          "Status": "Acquired"
        },
        {
          "E164PhoneNumber": "+12065550102",
          "Status": "Failed"
        }
      ],
      "CreatedTimestamp": "2019-08-09T21:35:21.427Z",
      "UpdatedTimestamp": "2019-08-09T21:35:31.926Z"
    }
  ],
  {
    "PhoneNumberOrderId": "cba54321-ed76-09f5-321g-h54i876j2109",
    "ProductType": "BusinessCalling",
    "Status": "Partial",
    "OrderedPhoneNumbers": [
      {
        "E164PhoneNumber": "+12065550103",
        "Status": "Acquired"
      },
      {
        "E164PhoneNumber": "+12065550104",
        "Status": "Acquired"
      },
      {
        "E164PhoneNumber": "+12065550105",
        "Status": "Failed"
      }
    ]
  }
}
```

```
    }
  ],
  "CreatedTimestamp": "2019-08-09T21:35:21.427Z",
  "UpdatedTimestamp": "2019-08-09T21:35:31.926Z"
}
]
```

For more information, see [Working with Phone Numbers](#) in the *Amazon Chime Administration Guide*.

- For API details, see [ListPhoneNumberOrders](#) in *AWS CLI Command Reference*.

list-phone-numbers

The following code example shows how to use `list-phone-numbers`.

AWS CLI

To list phone numbers for an Amazon Chime account

The following `list-phone-numbers` example lists the phone numbers associated with the administrator's Amazon Chime account.

```
aws chime list-phone-numbers
```

This command produces no output. Output:

```
{
  "PhoneNumbers": [
    {
      "PhoneNumberId": "%2B12065550100",
      "E164PhoneNumber": "+12065550100",
      "Type": "Local",
      "ProductType": "VoiceConnector",
      "Status": "Assigned",
      "Capabilities": {
        "InboundCall": true,
        "OutboundCall": true,
        "InboundSMS": true,
        "OutboundSMS": true,
        "InboundMMS": true,
        "OutboundMMS": true
      }
    }
  ]
}
```

```

    },
    "Associations": [
      {
        "Value": "abcdef1ghij2klmno3pqr4",
        "Name": "VoiceConnectorId",
        "AssociatedTimestamp": "2019-10-28T18:40:37.453Z"
      }
    ],
    "CallingNameStatus": "UpdateInProgress",
    "CreatedTimestamp": "2019-08-12T22:10:20.521Z",
    "UpdatedTimestamp": "2019-10-28T18:42:07.964Z"
  },
  {
    "PhoneNumberId": "%2B12065550101",
    "E164PhoneNumber": "+12065550101",
    "Type": "Local",
    "ProductType": "VoiceConnector",
    "Status": "Assigned",
    "Capabilities": {
      "InboundCall": true,
      "OutboundCall": true,
      "InboundSMS": true,
      "OutboundSMS": true,
      "InboundMMS": true,
      "OutboundMMS": true
    },
    "Associations": [
      {
        "Value": "abcdef1ghij2klmno3pqr4",
        "Name": "VoiceConnectorId",
        "AssociatedTimestamp": "2019-10-28T18:40:37.511Z"
      }
    ],
    "CallingNameStatus": "UpdateInProgress",
    "CreatedTimestamp": "2019-08-12T22:10:20.521Z",
    "UpdatedTimestamp": "2019-10-28T18:42:07.960Z"
  }
]
}

```

For more information, see [Working with Phone Numbers](#) in the *Amazon Chime Administration Guide*.

- For API details, see [ListPhoneNumbers](#) in *AWS CLI Command Reference*.

list-proxy-sessions

The following code example shows how to use `list-proxy-sessions`.

AWS CLI

To list proxy sessions

The following `list-proxy-sessions` example lists the proxy sessions for your Amazon Chime Voice Connector.

```
aws chime list-proxy-sessions \
  --voice-connector-id abcdef1ghij2klmno3pqr4
```

Output:

```
{
  "ProxySession": {
    "VoiceConnectorId": "abcdef1ghij2klmno3pqr4",
    "ProxySessionId": "123a4bc5-67d8-901e-2f3g-h4ghjk567891",
    "Status": "Open",
    "ExpiryMinutes": 60,
    "Capabilities": [
      "SMS",
      "Voice"
    ],
    "CreatedTimestamp": "2020-04-15T16:10:10.288Z",
    "UpdatedTimestamp": "2020-04-15T16:10:10.288Z",
    "Participants": [
      {
        "PhoneNumber": "+12065550100",
        "ProxyPhoneNumber": "+19135550199"
      },
      {
        "PhoneNumber": "+14015550101",
        "ProxyPhoneNumber": "+19135550199"
      }
    ]
  }
}
```

For more information, see [Proxy Phone Sessions](#) in the *Amazon Chime Developer Guide*.

- For API details, see [ListProxySessions](#) in *AWS CLI Command Reference*.

list-room-memberships

The following code example shows how to use `list-room-memberships`.

AWS CLI

To list room memberships

The following `list-room-memberships` example displays a list of the membership details for the specified chat room.

```
aws chime list-room-memberships \  
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \  
  --room-id abcd1e2d-3e45-6789-01f2-3g45h67i890j
```

Output:

```
{  
  "RoomMemberships": [  
    {  
      "RoomId": "abcd1e2d-3e45-6789-01f2-3g45h67i890j",  
      "Member": {  
        "MemberId": "2ab2345c-67de-8901-f23g-45h678901j2k",  
        "MemberType": "User",  
        "Email": "zhangw@example.com",  
        "FullName": "Zhang Wei",  
        "AccountId": "12a3456b-7c89-012d-3456-78901e23fg45"  
      },  
      "Role": "Member",  
      "InvitedBy": "arn:aws:iam::111122223333:user/alejandro",  
      "UpdatedTimestamp": "2019-12-02T22:46:58.532Z"  
    },  
    {  
      "RoomId": "abcd1e2d-3e45-6789-01f2-3g45h67i890j",  
      "Member": {  
        "MemberId": "1ab2345c-67de-8901-f23g-45h678901j2k",  
        "MemberType": "User",  
        "Email": "janed@example.com",  
        "FullName": "Jane Doe",  
        "AccountId": "12a3456b-7c89-012d-3456-78901e23fg45"  
      }  
    }  
  ]  
}
```

```
    },
    "Role": "Administrator",
    "InvitedBy": "arn:aws:iam::111122223333:user/alejandro",
    "UpdatedTimestamp": "2019-12-02T22:46:58.532Z"
  }
]
}
```

For more information, see [Creating a Chat Room](#) in the *Amazon Chime User Guide*.

- For API details, see [ListRoomMemberships](#) in *AWS CLI Command Reference*.

list-rooms

The following code example shows how to use `list-rooms`.

AWS CLI

To list chat rooms

The following `list-rooms` example displays a list of chat rooms in the specified account. The list is filtered to only those chat rooms that the specified member belongs to.

```
aws chime list-rooms \
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \
  --member-id 1ab2345c-67de-8901-f23g-45h678901j2k
```

Output:

```
{
  "Room": {
    "RoomId": "abcd1e2d-3e45-6789-01f2-3g45h67i890j",
    "Name": "teamRoom",
    "AccountId": "12a3456b-7c89-012d-3456-78901e23fg45",
    "CreatedBy": "arn:aws:iam::111122223333:user/alejandro",
    "CreatedTimestamp": "2019-12-02T22:29:31.549Z",
    "UpdatedTimestamp": "2019-12-02T22:33:19.310Z"
  }
}
```

For more information, see [Creating a Chat Room](#) in the *Amazon Chime User Guide*.

- For API details, see [ListRooms](#) in *AWS CLI Command Reference*.

list-users

The following code example shows how to use `list-users`.

AWS CLI

To list the users in an account

The following `list-users` example lists the users for the specified Amazon Chime account.

```
aws chime list-users --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
```

Output:

```
{
  "Users": [
    {
      "UserId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
      "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
      "PrimaryEmail": "mariag@example.com",
      "DisplayName": "Maria Garcia",
      "LicenseType": "Pro",
      "UserType": "PrivateUser",
      "UserRegistrationStatus": "Registered",
      "RegisteredOn": "2018-12-20T18:45:25.231Z"
      "AlexaForBusinessMetadata": {
        "IsAlexaForBusinessEnabled": false
      }
    },
    {
      "UserId": "a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
      "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
      "PrimaryEmail": "richardr@example.com",
      "DisplayName": "Richard Roe",
      "LicenseType": "Pro",
      "UserType": "PrivateUser",
      "UserRegistrationStatus": "Registered",
      "RegisteredOn": "2018-12-20T18:45:45.415Z"
      "AlexaForBusinessMetadata": {
        "IsAlexaForBusinessEnabled": false
      }
    },
    {
      "UserId": "a1b2c3d4-5678-90ab-cdef-44444EXAMPLE",
```

```

    "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "PrimaryEmail": "saanvis@example.com",
    "DisplayName": "Saanvi Sarkar",
    "LicenseType": "Basic",
    "UserType": "PrivateUser",
    "UserRegistrationStatus": "Registered",
    "RegisteredOn": "2018-12-20T18:46:57.747Z"
    "AlexaForBusinessMetadata": {
      "IsAlexaForBusinessEnabled": false
    }
  },
  {
    "UserId": "a1b2c3d4-5678-90ab-cdef-55555EXAMPLE",
    "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "PrimaryEmail": "wxiulan@example.com",
    "DisplayName": "Wang Xiulan",
    "LicenseType": "Basic",
    "UserType": "PrivateUser",
    "UserRegistrationStatus": "Registered",
    "RegisteredOn": "2018-12-20T18:47:15.390Z"
    "AlexaForBusinessMetadata": {
      "IsAlexaForBusinessEnabled": false
    }
  }
]
}

```

For more information, see [Managing Users](#) in the *Amazon Chime Administration Guide*.

- For API details, see [ListUsers](#) in *AWS CLI Command Reference*.

list-voice-connector-groups

The following code example shows how to use `list-voice-connector-groups`.

AWS CLI

To list Amazon Chime Voice Connector groups for an Amazon Chime account

The following `list-voice-connector-groups` example lists the Amazon Chime Voice Connector groups associated with the administrator's Amazon Chime account.

```
aws chime list-voice-connector-groups
```

Output:

```
{
  "VoiceConnectorGroups": [
    {
      "VoiceConnectorGroupId": "123a456b-c7d8-90e1-fg23-4h567jkl8901",
      "Name": "myGroup",
      "VoiceConnectorItems": [],
      "CreatedTimestamp": "2019-09-18T16:38:34.734Z",
      "UpdatedTimestamp": "2019-09-18T16:38:34.734Z"
    }
  ]
}
```

For more information, see [Working with Amazon Chime Voice Connector groups](#) in the *Amazon Chime Administration Guide*.

- For API details, see [ListVoiceConnectorGroups](#) in *AWS CLI Command Reference*.

list-voice-connector-termination-credentials

The following code example shows how to use `list-voice-connector-termination-credentials`.

AWS CLI**To retrieve a list of termination credentials**

The following `list-voice-connector-termination-credentials` example retrieves a list of the termination credentials for the specified Amazon Chime Voice Connector.

```
aws chime list-voice-connector-termination-credentials \
  --voice-connector-id abcdef1ghij2klmno3pqr4
```

This command produces no output. Output:

```
{
  "Usernames": [
    "jdoe"
  ]
}
```

For more information, see [Working with Amazon Chime Voice Connectors](#) in the *Amazon Chime Administration Guide*.

- For API details, see [ListVoiceConnectorTerminationCredentials](#) in *AWS CLI Command Reference*.

list-voice-connectors

The following code example shows how to use `list-voice-connectors`.

AWS CLI

To list Amazon Chime Voice Connectors for an account

The following `list-voice-connectors` example lists the Amazon Chime Voice Connectors associated with the caller's account.

```
aws chime list-voice-connectors
```

Output:

```
{
  "VoiceConnectors": [
    {
      "VoiceConnectorId": "abcdef1ghij2klmno3pqr4",
      "AwsRegion": "us-east-1",
      "Name": "MyVoiceConnector",
      "OutboundHostName": "abcdef1ghij2klmno3pqr4.voiceconnector.chime.aws",
      "RequireEncryption": true,
      "CreatedTimestamp": "2019-06-04T18:46:56.508Z",
      "UpdatedTimestamp": "2019-09-18T16:33:00.806Z"
    },
    {
      "VoiceConnectorId": "cbadef1ghij2klmno3pqr5",
      "AwsRegion": "us-west-2",
      "Name": "newVoiceConnector",
      "OutboundHostName": "cbadef1ghij2klmno3pqr5.voiceconnector.chime.aws",
      "RequireEncryption": true,
      "CreatedTimestamp": "2019-09-18T20:34:01.352Z",
      "UpdatedTimestamp": "2019-09-18T20:34:01.352Z"
    }
  ]
}
```

```
}
```

For more information, see [Working with Amazon Chime Voice Connectors](#) in the *Amazon Chime Administration Guide*.

- For API details, see [ListVoiceConnectors](#) in *AWS CLI Command Reference*.

logout-user

The following code example shows how to use `logout-user`.

AWS CLI

To log out a user

The following `logout-user` example logs out the specified user.

```
aws chime logout-user \  
  --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \  
  --user-id a1b2c3d4-5678-90ab-cdef-22222EXAMPLE
```

This command produces no output.

- For API details, see [LogoutUser](#) in *AWS CLI Command Reference*.

put-voice-connector-logging-configuration

The following code example shows how to use `put-voice-connector-logging-configuration`.

AWS CLI

To add a logging configuration for an Amazon Chime Voice Connector

The following `put-voice-connector-logging-configuration` example turns on the SIP logging configuration for the specified Amazon Chime Voice Connector.

```
aws chime put-voice-connector-logging-configuration \  
  --voice-connector-id abcdef1ghij2klmno3pqr4 \  
  --logging-configuration EnableSIPLogs=true
```


Output:

```
{
  "LoggingConfiguration": {
    "EnableSIPLogs": true
  }
}
```

For more information, see [Streaming Amazon Chime Voice Connector Media to Kinesis](#) in the *Amazon Chime Administration Guide*.

- For API details, see [PutVoiceConnectorLoggingConfiguration](#) in *AWS CLI Command Reference*.

put-voice-connector-origination

The following code example shows how to use `put-voice-connector-origination`.

AWS CLI**To set up origination settings**

The following `put-voice-connector-origination` example sets up the origination host, port, protocol, priority, and weight for the specified Amazon Chime Voice Connector.

```
aws chime put-voice-connector-origination \
  --voice-connector-id abcdef1ghij2klmno3pqr4 \
  --origination
  Routes=[{Host="10.24.34.0",Port=1234,Protocol="TCP",Priority=1,Weight=5}],Disabled=false
```

Output:

```
{
  "Origination": {
    "Routes": [
      {
        "Host": "10.24.34.0",
        "Port": 1234,
        "Protocol": "TCP",
        "Priority": 1,
        "Weight": 5
      }
    ]
  }
}
```

```
    ],
    "Disabled": false
  }
}
```

For more information, see [Working with Amazon Chime Voice Connectors](#) in the *Amazon Chime Administration Guide*.

- For API details, see [PutVoiceConnectorOrigination](#) in *AWS CLI Command Reference*.

put-voice-connector-proxy

The following code example shows how to use `put-voice-connector-proxy`.

AWS CLI

To put a proxy configuration

The following `put-voice-connector-proxy` example sets a proxy configuration to your Amazon Chime Voice Connector.

```
aws chime put-voice-connector-proxy \
  --voice-connector-id abcdef1ghij2klmno3pqr4 \
  --default-session-expiry-minutes 60 \
  --phone-number-pool-countries "US"
```

Output:

```
{
  "Proxy": {
    "DefaultSessionExpiryMinutes": 60,
    "Disabled": false,
    "PhoneNumberCountries": [
      "US"
    ]
  }
}
```

For more information, see [Proxy Phone Sessions](#) in the *Amazon Chime Developer Guide*.

- For API details, see [PutVoiceConnectorProxy](#) in *AWS CLI Command Reference*.

put-voice-connector-streaming-configuration

The following code example shows how to use `put-voice-connector-streaming-configuration`.

AWS CLI

To create a streaming configuration

The following `put-voice-connector-streaming-configuration` example creates a streaming configuration for the specified Amazon Chime Voice Connector. It enables media streaming from the Amazon Chime Voice Connector to Amazon Kinesis, and sets the data retention period to 24 hours.

```
aws chime put-voice-connector-streaming-configuration \  
  --voice-connector-id abcdef1ghij2klmno3pqr4 \  
  --streaming-configuration DataRetentionInHours=24,Disabled=false
```

Output:

```
{  
  "StreamingConfiguration": {  
    "DataRetentionInHours": 24,  
    "Disabled": false  
  }  
}
```

For more information, see [Streaming Amazon Chime Voice Connector Data to Kinesis](#) in the *Amazon Chime Administration Guide*.

- For API details, see [PutVoiceConnectorStreamingConfiguration](#) in *AWS CLI Command Reference*.

put-voice-connector-termination-credentials

The following code example shows how to use `put-voice-connector-termination-credentials`.

AWS CLI

To set up termination credentials

The following `put-voice-connector-termination-credentials` example sets termination credentials for the specified Amazon Chime Voice Connector.

```
aws chime put-voice-connector-termination-credentials \  
  --voice-connector-id abcdef1ghij2klmno3pqr4 \  
  --credentials Username="jdoe",Password="XXXXXXXXX"
```

This command produces no output.

For more information, see [Working with Amazon Chime Voice Connectors](#) in the *Amazon Chime Administration Guide*.

- For API details, see [PutVoiceConnectorTerminationCredentials](#) in *AWS CLI Command Reference*.

put-voice-connector-termination

The following code example shows how to use `put-voice-connector-termination`.

AWS CLI

To set up termination settings

The following `put-voice-connector-termination` example sets the calling regions and allowed IP host termination settings for the specified Amazon Chime Voice Connector.

```
aws chime put-voice-connector-termination \  
  --voice-connector-id abcdef1ghij2klmno3pqr4 \  
  --termination CallingRegions="US",CidrAllowedList="10.24.34.0/23",Disabled=false
```

Output:

```
{  
  "Termination": {  
    "CpsLimit": 0,  
    "CallingRegions": [  
      "US"  
    ],  
    "CidrAllowedList": [  
      "10.24.34.0/23"  
    ],  
    "Disabled": false  
  }  
}
```

```
}  
}
```

For more information, see [Working with Amazon Chime Voice Connectors](#) in the *Amazon Chime Administration Guide*.

- For API details, see [PutVoiceConnectorTermination](#) in *AWS CLI Command Reference*.

regenerate-security-token

The following code example shows how to use `regenerate-security-token`.

AWS CLI

To regenerate a security token

The following `regenerate-security-token` example regenerates the security token for the specified bot.

```
aws chime regenerate-security-token \  
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \  
  --bot-id 123abcd4-5ef6-789g-0h12-34j56789012k
```

Output:

```
{  
  "Bot": {  
    "BotId": "123abcd4-5ef6-789g-0h12-34j56789012k",  
    "UserId": "123abcd4-5ef6-789g-0h12-34j56789012k",  
    "DisplayName": "myBot (Bot)",  
    "BotType": "ChatBot",  
    "Disabled": false,  
    "CreatedTimestamp": "2019-09-09T18:05:56.749Z",  
    "UpdatedTimestamp": "2019-09-09T18:05:56.749Z",  
    "BotEmail": "myBot-chimebot@example.com",  
    "SecurityToken": "je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY"  
  }  
}
```

For more information, see [Authenticate Chat Bot Requests](#) in the *Amazon Chime Developer Guide*.

- For API details, see [RegenerateSecurityToken](#) in *AWS CLI Command Reference*.

reset-personal-pin

The following code example shows how to use `reset-personal-pin`.

AWS CLI

To reset a user's personal meeting PIN

The following `reset-personal-pin` example resets the specified user's personal meeting PIN.

```
aws chime reset-personal-pin \  
  --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \  
  --user-id a1b2c3d4-5678-90ab-cdef-22222EXAMPLE
```

Output:

```
{  
  "User": {  
    "UserId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",  
    "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
    "PrimaryEmail": "mateo@example.com",  
    "DisplayName": "Mateo Jackson",  
    "LicenseType": "Pro",  
    "UserType": "PrivateUser",  
    "UserRegistrationStatus": "Registered",  
    "RegisteredOn": "2018-12-20T18:45:25.231Z",  
    "AlexaForBusinessMetadata": {  
      "IsAlexaForBusinessEnabled": False,  
      "AlexaForBusinessRoomArn": "null"  
    },  
    "PersonalPIN": "XXXXXXXXXX"  
  }  
}
```

For more information, see [Changing Personal Meeting PINs](#) in the *Amazon Chime Administration Guide*.

- For API details, see [ResetPersonalPin](#) in *AWS CLI Command Reference*.

restore-phone-number

The following code example shows how to use `restore-phone-number`.

AWS CLI

To restore a phone number

The following `restore-phone-number` example restores the specified phone number from the deletion queue.

```
aws chime restore-phone-number \  
  --phone-number-id "+12065550100"
```

Output:

```
{  
  "PhoneNumber": {  
    "PhoneNumberId": "%2B12065550100",  
    "E164PhoneNumber": "+12065550100",  
    "Type": "Local",  
    "ProductType": "BusinessCalling",  
    "Status": "Unassigned",  
    "Capabilities": {  
      "InboundCall": true,  
      "OutboundCall": true,  
      "InboundSMS": true,  
      "OutboundSMS": true,  
      "InboundMMS": true,  
      "OutboundMMS": true  
    },  
    "Associations": [],  
    "CreatedTimestamp": "2019-08-09T21:35:21.445Z",  
    "UpdatedTimestamp": "2019-08-12T22:06:36.355Z"  
  }  
}
```

For more information, see [Working with Phone Numbers](#) in the *Amazon Chime Administration Guide*.

- For API details, see [RestorePhoneNumber](#) in *AWS CLI Command Reference*.

`search-available-phone-numbers`

The following code example shows how to use `search-available-phone-numbers`.

AWS CLI

To search available phone numbers

The following `search-available-phone-numbers` example searches available phone numbers by area code.

```
aws chime search-available-phone-numbers \  
  --area-code "206"
```

Output:

```
{  
  "E164PhoneNumbers": [  
    "+12065550100",  
    "+12065550101",  
    "+12065550102",  
    "+12065550103",  
    "+12065550104",  
    "+12065550105",  
    "+12065550106",  
    "+12065550107",  
    "+12065550108",  
    "+12065550109",  
  ]  
}
```

For more information, see [Working with Phone Numbers](#) in the *Amazon Chime Administration Guide*.

- For API details, see [SearchAvailablePhoneNumbers](#) in *AWS CLI Command Reference*.

update-account-settings

The following code example shows how to use `update-account-settings`.

AWS CLI

To update the settings for your account

The following `update-account-settings` example disables the remote control of shared screens for the specified Amazon Chime account.


```
aws chime update-account-settings \  
  --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \  
  --account-settings DisableRemoteControl=true
```

This command produces no output.

- For API details, see [UpdateAccountSettings](#) in *AWS CLI Command Reference*.

update-account

The following code example shows how to use update-account.

AWS CLI

To update an account

The following update-account example updates the specified account name.

```
aws chime update-account \  
  --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \  
  --name MyAccountName
```

Output:

```
{  
  "Account": {  
    "AwsAccountId": "111122223333",  
    "AccountId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
    "Name": "MyAccountName",  
    "AccountType": "Team",  
    "CreatedTimestamp": "2018-09-04T21:44:22.292Z",  
    "DefaultLicense": "Pro",  
    "SupportedLicenses": [  
      "Basic",  
      "Pro"  
    ],  
    "SigninDelegateGroups": [  
      {  
        "GroupName": "myGroup"  
      },  
    ]  
  }  
}
```

```
}  
}
```

For more information, see [Renaming Your Account](#) in the *Amazon Chime Administration Guide*.

- For API details, see [UpdateAccount](#) in *AWS CLI Command Reference*.

update-bot

The following code example shows how to use `update-bot`.

AWS CLI

To update a bot

The following `update-bot` example updates the status of the specified bot to stop it from running.

```
aws chime update-bot \  
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \  
  --bot-id 123abcd4-5ef6-789g-0h12-34j56789012k \  
  --disabled
```

Output:

```
{  
  "Bot": {  
    "BotId": "123abcd4-5ef6-789g-0h12-34j56789012k",  
    "UserId": "123abcd4-5ef6-789g-0h12-34j56789012k",  
    "DisplayName": "myBot (Bot)",  
    "BotType": "ChatBot",  
    "Disabled": true,  
    "CreatedTimestamp": "2019-09-09T18:05:56.749Z",  
    "UpdatedTimestamp": "2019-09-09T18:05:56.749Z",  
    "BotEmail": "myBot-chimebot@example.com",  
    "SecurityToken": "je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY"  
  }  
}
```

For more information, see [Update Chat Bots](#) in the *Amazon Chime Developer Guide*.

- For API details, see [UpdateBot](#) in *AWS CLI Command Reference*.

update-global-settings

The following code example shows how to use `update-global-settings`.

AWS CLI

To update global settings

The following `update-global-settings` example updates the S3 bucket used to store call detail records for Amazon Chime Business Calling and Amazon Chime Voice Connectors associated with the administrator's AWS account.

```
aws chime update-global-settings \  
  --business-calling CdrBucket="s3bucket" \  
  --voice-connector CdrBucket="s3bucket"
```

This command produces no output.

For more information, see [Managing Global Settings](#) in the *Amazon Chime Administration Guide*.

- For API details, see [UpdateGlobalSettings](#) in *AWS CLI Command Reference*.

update-phone-number-settings

The following code example shows how to use `update-phone-number-settings`.

AWS CLI

To update an outbound calling name

The following `update-phone-number-settings` example updates the default outbound calling name for the administrator's AWS account.

```
aws chime update-phone-number-settings \  
  --calling-name "myName"
```

This command produces no output.

For more information, see [Working with Phone Numbers](#) in the *Amazon Chime Administration Guide*.

- For API details, see [UpdatePhoneNumberSettings](#) in *AWS CLI Command Reference*.

update-phone-number

The following code example shows how to use update-phone-number.

AWS CLI

Example 1: To update the product type for a phone number

The following update-phone-number example updates the specified phone number's product type.

```
aws chime update-phone-number \  
  --phone-number-id "+12065550100" \  
  --product-type "BusinessCalling"
```

Output:

```
{  
  "PhoneNumber": {  
    "PhoneNumberId": "%2B12065550100",  
    "E164PhoneNumber": "+12065550100",  
    "Type": "Local",  
    "ProductType": "BusinessCalling",  
    "Status": "Unassigned",  
    "Capabilities": {  
      "InboundCall": true,  
      "OutboundCall": true,  
      "InboundSMS": true,  
      "OutboundSMS": true,  
      "InboundMMS": true,  
      "OutboundMMS": true  
    },  
    "Associations": [],  
    "CallingName": "phonenumber1",  
    "CreatedTimestamp": "2019-08-09T21:35:21.445Z",  
    "UpdatedTimestamp": "2019-08-12T21:44:07.591Z"  
  }  
}
```

Example 2: To update the outbound calling name for a phone number

The following update-phone-number example updates the outbound calling name for the specified phone number.

```
aws chime update-phone-number --phone-number-id "+12065550100" --calling-name
"phonenumber2"
```

Output:

```
{
  "PhoneNumber": {
    "PhoneNumberId": "%2B12065550100",
    "E164PhoneNumber": "+12065550100",
    "Type": "Local",
    "ProductType": "BusinessCalling",
    "Status": "Unassigned",
    "Capabilities": {
      "InboundCall": true,
      "OutboundCall": true,
      "InboundSMS": true,
      "OutboundSMS": true,
      "InboundMMS": true,
      "OutboundMMS": true
    },
    "Associations": [],
    "CallingName": "phonenumber2",
    "CreatedTimestamp": "2019-08-09T21:35:21.445Z",
    "UpdatedTimestamp": "2019-08-12T21:44:07.591Z"
  }
}
```

For more information, see [Working with Phone Numbers](#) in the *Amazon Chime Administration Guide*.

- For API details, see [UpdatePhoneNumber](#) in *AWS CLI Command Reference*.

update-proxy-session

The following code example shows how to use `update-proxy-session`.

AWS CLI

To update a proxy session

The following `update-proxy-session` example updates the proxy session capabilities.

```
aws chime update-proxy-session \
```

```
--voice-connector-id abcdef1ghij2klmno3pqr4 \  
--proxy-session-id 123a4bc5-67d8-901e-2f3g-h4ghjk567891 \  
--capabilities "Voice"
```

Output:

```
{  
  "ProxySession": {  
    "VoiceConnectorId": "abcdef1ghij2klmno3pqr4",  
    "ProxySessionId": "123a4bc5-67d8-901e-2f3g-h4ghjk567891",  
    "Status": "Open",  
    "ExpiryMinutes": 60,  
    "Capabilities": [  
      "Voice"  
    ],  
    "CreatedTimestamp": "2020-04-15T16:10:10.288Z",  
    "UpdatedTimestamp": "2020-04-15T16:10:10.288Z",  
    "Participants": [  
      {  
        "PhoneNumber": "+12065550100",  
        "ProxyPhoneNumber": "+19135550199"  
      },  
      {  
        "PhoneNumber": "+14015550101",  
        "ProxyPhoneNumber": "+19135550199"  
      }  
    ]  
  }  
}
```

For more information, see [Proxy Phone Sessions](#) in the *Amazon Chime Developer Guide*.

- For API details, see [UpdateProxySession](#) in *AWS CLI Command Reference*.

update-room-membership

The following code example shows how to use `update-room-membership`.

AWS CLI

To update a room membership

The following `update-room-membership` example modifies the role of the specified chat room member to `Administrator`.

```
aws chime update-room-membership \  
  --account-id 12a3456b-7c89-012d-3456-78901e23fg45 \  
  --room-id abcd1e2d-3e45-6789-01f2-3g45h67i890j \  
  --member-id 1ab2345c-67de-8901-f23g-45h678901j2k \  
  --role Administrator
```

Output:

```
{  
  "RoomMembership": {  
    "RoomId": "abcd1e2d-3e45-6789-01f2-3g45h67i890j",  
    "Member": {  
      "MemberId": "1ab2345c-67de-8901-f23g-45h678901j2k",  
      "MemberType": "User",  
      "Email": "sofiamartinez@example.com",  
      "FullName": "Sofia Martinez",  
      "AccountId": "12a3456b-7c89-012d-3456-78901e23fg45"  
    },  
    "Role": "Administrator",  
    "InvitedBy": "arn:aws:iam::111122223333:user/admin",  
    "UpdatedTimestamp": "2019-12-02T22:40:22.931Z"  
  }  
}
```

For more information, see [Creating a Chat Room](#) in the *Amazon Chime User Guide*.

- For API details, see [UpdateRoomMembership](#) in *AWS CLI Command Reference*.

update-room

The following code example shows how to use `update-room`.

AWS CLI

To update a chat room

The following `update-room` example modifies the name of the specified chat room.

```
aws chime update-room \  
  --room-name my-room
```

```
--account-id 12a3456b-7c89-012d-3456-78901e23fg45 \  
--room-id abcd1e2d-3e45-6789-01f2-3g45h67i890j \  
--name teamRoom
```

Output:

```
{  
  "Room": {  
    "RoomId": "abcd1e2d-3e45-6789-01f2-3g45h67i890j",  
    "Name": "teamRoom",  
    "AccountId": "12a3456b-7c89-012d-3456-78901e23fg45",  
    "CreatedBy": "arn:aws:iam::111122223333:user/alejandro",  
    "CreatedTimestamp": "2019-12-02T22:29:31.549Z",  
    "UpdatedTimestamp": "2019-12-02T22:33:19.310Z"  
  }  
}
```

For more information, see [Creating a Chat Room](#) in the *Amazon Chime User Guide*.

- For API details, see [UpdateRoom](#) in *AWS CLI Command Reference*.

update-user-settings

The following code example shows how to use `update-user-settings`.

AWS CLI

To update user settings

The following `update-user-settings` example enables the specified user to make inbound and outbound calls and send and receive SMS messages.

```
aws chime update-user-settings \  
--account-id 12a3456b-7c89-012d-3456-78901e23fg45 \  
--user-id 1ab2345c-67de-8901-f23g-45h678901j2k \  
--user-settings "Telephony={InboundCalling=true,OutboundCalling=true,SMS=true}"
```

This command produces no output.

For more information, see [Managing User Phone Numbers](#) in the *Amazon Chime Administration Guide*.

- For API details, see [UpdateUserSettings](#) in *AWS CLI Command Reference*.

update-user

The following code example shows how to use `update-user`.

AWS CLI

To update user details

This example updates the specified details for the specified user.

Command:

```
aws chime update-user \  
  --account-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \  
  --user-id a1b2c3d4-5678-90ab-cdef-22222EXAMPLE \  
  --license-type "Basic"
```

Output:

```
{  
  "User": {  
    "UserId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE"  
  }  
}
```

- For API details, see [UpdateUser](#) in *AWS CLI Command Reference*.

update-voice-connector-group

The following code example shows how to use `update-voice-connector-group`.

AWS CLI

To update the details for an Amazon Chime Voice Connector group

The following `update-voice-connector-group` example updates the details of the specified Amazon Chime Voice Connector group.

```
aws chime update-voice-connector-group \  
  --voice-connector-group-id 123a456b-c7d8-90e1-fg23-4h567jk18901 \  
  --name "newGroupName" \  
  --voice-connector-items VoiceConnectorId=abcdef1ghij2klmno3pqr4,Priority=1
```

Output:

```
{
  "VoiceConnectorGroup": {
    "VoiceConnectorGroupId": "123a456b-c7d8-90e1-fg23-4h567jk18901",
    "Name": "newGroupName",
    "VoiceConnectorItems": [
      {
        "VoiceConnectorId": "abcdef1ghij2klmno3pqr4",
        "Priority": 1
      }
    ],
    "CreatedTimestamp": "2019-09-18T16:38:34.734Z",
    "UpdatedTimestamp": "2019-10-28T19:00:57.081Z"
  }
}
```

For more information, see [Working with Amazon Chime Voice Connector Groups](#) in the *Amazon Chime Administration Guide*.

- For API details, see [UpdateVoiceConnectorGroup](#) in *AWS CLI Command Reference*.

update-voice-connector

The following code example shows how to use `update-voice-connector`.

AWS CLI**To update the details for an Amazon Chime Voice Connector**

The following `update-voice-connector` example updates the name of the specified Amazon Chime Voice Connector.

```
aws chime update-voice-connector \
  --voice-connector-id abcdef1ghij2klmno3pqr4 \
  --name newName \
  --require-encryption
```

Output:

```
{
  "VoiceConnector": {
```

```
"VoiceConnectorId": "abcdef1ghij2klmno3pqr4",
"AwsRegion": "us-west-2",
"Name": "newName",
"OutboundHostName": "abcdef1ghij2klmno3pqr4.voiceconnector.chime.aws",
"RequireEncryption": true,
"CreatedTimestamp": "2019-09-18T20:34:01.352Z",
"UpdatedTimestamp": "2019-09-18T20:40:52.895Z"
}
}
```

For more information, see [Working with Amazon Chime Voice Connectors](#) in the *Amazon Chime Administration Guide*.

- For API details, see [UpdateVoiceConnector](#) in *AWS CLI Command Reference*.

Cloud Control API examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Cloud Control API.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-resource

The following code example shows how to use `create-resource`.

AWS CLI

To create a resource

The following `create-resource` example creates an `AWS::Kinesis::Stream` resource, named `ResourceExample`, with a retention period of 168 hours and a shard count of three.

```
aws cloudcontrol create-resource \  
  --type-name AWS::Kinesis::Stream \  
  --desired-state "{\"Name\": \"ResourceExample\", \"RetentionPeriodHours\":168, \  
  \"ShardCount\":3}"
```

Output:

```
{  
  "ProgressEvent": {  
    "EventTime": 1632506656.706,  
    "TypeName": "AWS::Kinesis::Stream",  
    "OperationStatus": "IN_PROGRESS",  
    "Operation": "CREATE",  
    "Identifier": "ResourceExample",  
    "RequestToken": "20999d87-e304-4725-ad84-832dcbfd7fc5"  
  }  
}
```

For more information, see [Creating a resource](#) in the *Cloud Control API User Guide*.

- For API details, see [CreateResource](#) in *AWS CLI Command Reference*.

delete-resource

The following code example shows how to use `delete-resource`.

AWS CLI

To delete a resource

The following `delete-resource` example deletes a `AWS::Kinesis::Stream` resource with the identifier `ResourceExample` from your AWS account.

```
aws cloudcontrol delete-resource \  
  --type-name AWS::Kinesis::Stream \  
  --identifier ResourceExample
```

Output:

```
{
  "ProgressEvent": {
    "TypeName": "AWS::Kinesis::Stream",
    "Identifier": "ResourceExample",
    "RequestToken": "e48f26ff-d0f9-4ab8-a878-120db1edf111",
    "Operation": "DELETE",
    "OperationStatus": "IN_PROGRESS",
    "EventTime": 1632950300.14
  }
}
```

For more information, see [Deleting a resource](#) in the *Cloud Control API User Guide*.

- For API details, see [DeleteResource](#) in *AWS CLI Command Reference*.

get-resource-request-status

The following code example shows how to use `get-resource-request-status`.

AWS CLI

To get the status information of a resource request

The following `get-resource-request-status` example returns status information about the specified resource request.

```
aws cloudcontrol get-resource-request-status \
  --request-token "e1a6b86e-46bd-41ac-bfba-001234567890"
```

Output:

```
{
  "ProgressEvent": {
    "TypeName": "AWS::Kinesis::Stream",
    "Identifier": "Demo",
    "RequestToken": "e1a6b86e-46bd-41ac-bfba-001234567890",
    "Operation": "CREATE",
    "OperationStatus": "FAILED",
    "EventTime": 1632950268.481,
    "StatusMessage": "Resource of type 'AWS::Kinesis::Stream' with identifier 'Demo' already exists.",
    "ErrorCode": "AlreadyExists"
  }
}
```

```
}
}
```

For more information, see [Managing resource operation requests](#) in the *Cloud Control API User Guide*.

- For API details, see [GetResourceRequestStatus](#) in *AWS CLI Command Reference*.

get-resource

The following code example shows how to use `get-resource`.

AWS CLI

To get the current state of a resource

The following `get-resource` example returns the current state of the `AWS::Kinesis::Stream` resource named `ResourceExample`.

```
aws cloudcontrol get-resource \
  --type-name AWS::Kinesis::Stream \
  --identifier ResourceExample
```

Output:

```
{
  "TypeName": "AWS::Kinesis::Stream",
  "ResourceDescription": {
    "Identifier": "ResourceExample",
    "Properties": "{\"Arn\":\"arn:aws:kinesis:us-west-2:099908667365:stream/ResourceExample\", \"RetentionPeriodHours\":168, \"Name\":\"ResourceExample\", \"ShardCount\":3}"
  }
}
```

For more information, see [Reading a resource's current state](#) in the *Cloud Control API User Guide*.

- For API details, see [GetResource](#) in *AWS CLI Command Reference*.

list-resource-requests

The following code example shows how to use `list-resource-requests`.

AWS CLI

To list the active resource operation requests

The following `list-resource-requests` example lists the resource requests for CREATE and UPDATE operations that have failed in your AWS account.

```
aws cloudcontrol list-resource-requests \  
  --resource-request-status-filter Operations=CREATE,OperationStatuses=FAILED
```

Output:

```
{  
  "ResourceRequestStatusSummaries": [  
    {  
      "TypeName": "AWS::Kinesis::Stream",  
      "Identifier": "Demo",  
      "RequestToken": "e1a6b86e-46bd-41ac-bfba-633abcdfdbd7",  
      "Operation": "CREATE",  
      "OperationStatus": "FAILED",  
      "EventTime": 1632950268.481,  
      "StatusMessage": "Resource of type 'AWS::Kinesis::Stream' with  
identifier 'Demo' already exists.",  
      "ErrorCode": "AlreadyExists"  
    }  
  ]  
}
```

For more information, see [Managing resource operation requests](#) in the *Cloud Control API User Guide*.

- For API details, see [ListResourceRequests](#) in *AWS CLI Command Reference*.

list-resources

The following code example shows how to use `list-resources`.

AWS CLI

To list the resources of a given type

The following `list-resources` example lists the `AWS::Kinesis::Stream` resources provisioned in your AWS account.

```
aws cloudcontrol list-resources \  
  --type-name AWS::Kinesis::Stream
```

Output:

```
{  
  "TypeName": "AWS::Kinesis::Stream",  
  "ResourceDescriptions": [  
    {  
      "Identifier": "MyKinesisStream",  
      "Properties": "{\"Name\":\"MyKinesisStream\"}"  
    },  
    {  
      "Identifier": "AnotherStream",  
      "Properties": "{\"Name\":\"AnotherStream\"}"  
    }  
  ]  
}
```

For more information, see [Discovering resources](#) in the *Cloud Control API User Guide*.

- For API details, see [ListResources](#) in *AWS CLI Command Reference*.

update-resource

The following code example shows how to use `update-resource`.

AWS CLI

To update the properties of an existing resource

The following `update-resource` example updates the retention policy of an `AWS::Logs::LogGroup` resource named `ExampleLogGroup` to 90 days.

```
aws cloudcontrol update-resource \  
  --type-name AWS::Logs::LogGroup \  
  --identifier ExampleLogGroup \  
  --patch-document "[{\"op\":\"replace\",\"path\":\"/RetentionInDays\",\"value\":"  
  \":90}]"
```

Output:


```
{
  "ProgressEvent": {
    "EventTime": "2021-08-09T18:17:15.219Z",
    "TypeName": "AWS::Logs::LogGroup",
    "OperationStatus": "IN_PROGRESS",
    "Operation": "UPDATE",
    "Identifier": "ExampleLogGroup",
    "RequestToken": "5f40c577-3534-4b20-9599-0b0123456789"
  }
}
```

For more information, see [Updating a resource](#) in the *Cloud Control API User Guide*.

- For API details, see [UpdateResource](#) in *AWS CLI Command Reference*.

AWS Cloud Map examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Cloud Map.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-private-dns-namespace

The following code example shows how to use `create-private-dns-namespace`.

AWS CLI

To create a private DNS namespace

The following `create-private-dns-namespace` example creates a private DNS namespace.

```
aws servicediscovery create-private-dns-namespace \  
  --name example.com \  
  --vpc vpc-1c56417b
```

Output:

```
{  
  "OperationId": "gv4g5meo7ndmeh4fqskygvk23d2fijwa-k9302yzd"  
}
```

To confirm that the operation succeeded, you can run `get-operation`. For more information, see [get-operation](#).

For more information, see [Creating namespaces](#) in the *AWS Cloud Map Developer Guide*.

- For API details, see [CreatePrivateDnsNamespace](#) in *AWS CLI Command Reference*.

create-service

The following code example shows how to use `create-service`.

AWS CLI

To create a service

The following `create-service` example creates a service.

```
aws servicediscovery create-service \  
  --name myservice \  
  --namespace-id ns-ylexjili4cdxy3xm \  
  --dns-config "NamespaceId=ns-  
ylexjili4cdxy3xm,RoutingPolicy=MULTIVALUE,DnsRecords=[{Type=A,TTL=60}]"
```

Output:

```
{
  "Service": {
    "Id": "srv-p5zdwlg5uvvzjita",
    "Arn": "arn:aws:servicediscovery:us-west-2:803642222207:service/srv-
p5zdwlg5uvvzjita",
    "Name": "myservice",
    "NamespaceId": "ns-ylexjili4cdxy3xm",
    "DnsConfig": {
      "NamespaceId": "ns-ylexjili4cdxy3xm",
      "RoutingPolicy": "MULTIVALUE",
      "DnsRecords": [
        {
          "Type": "A",
          "TTL": 60
        }
      ]
    },
    "CreateDate": 1587081768.334,
    "CreatorRequestId": "567c1193-6b00-4308-bd57-ad38a8822d25"
  }
}
```

For more information, see [Creating services](#) in the *AWS Cloud Map Developer Guide*.

- For API details, see [CreateService](#) in *AWS CLI Command Reference*.

delete-namespace

The following code example shows how to use delete-namespace.

AWS CLI

To delete a namespace

The following delete-namespace example deletes a namespace.

```
aws servicediscovery delete-namespace \
  --id ns-ylexjili4cdxy3xm
```

Output:

```
{
```

```
"OperationId": "gv4g5meo7ndmeh4fqskygvk23d2fijwa-k98y6d1k"  
}
```

To confirm that the operation succeeded, you can run `get-operation`. For more information, see [get-operation](#).

For more information, see [Deleting namespaces](#) in the *AWS Cloud Map Developer Guide*.

- For API details, see [DeleteNamespace](#) in *AWS CLI Command Reference*.

delete-service

The following code example shows how to use `delete-service`.

AWS CLI

To delete a service

The following `delete-service` example deletes a service.

```
aws servicediscovery delete-service \  
  --id srv-p5zdwlg5uvvzjita
```

This command produces no output.

For more information, see [Deleting services](#) in the *AWS Cloud Map Developer Guide*.

- For API details, see [DeleteService](#) in *AWS CLI Command Reference*.

deregister-instance

The following code example shows how to use `deregister-instance`.

AWS CLI

To deregister a service instance

The following `deregister-instance` example deregisters a service instance.

```
aws servicediscovery deregister-instance \  
  --id srv-p5zdwlg5uvvzjita
```

```
--service-id srv-p5zdwlg5uvvzjita \  
--instance-id myservice-53
```

Output:

```
{  
  "OperationId": "4yejorelbukcjzpnr6t1mrghsjwpngf4-k98rnaiq"  
}
```

To confirm that the operation succeeded, you can run `get-operation`. For more information, see [get-operation](#).

For more information, see [Deregistering service instances](#) in the *AWS Cloud Map Developer Guide*.

- For API details, see [DeregisterInstance](#) in *AWS CLI Command Reference*.

discover-instances

The following code example shows how to use `discover-instances`.

AWS CLI

To discover registered instances

The following `discover-instances` example discovers registered instances.

```
aws servicediscovery discover-instances \  
  --namespace-name example.com \  
  --service-name myservice \  
  --max-results 10 \  
  --health-status ALL
```

Output:

```
{  
  "Instances": [  
    {  
      "InstanceId": "myservice-53",  
      "NamespaceName": "example.com",
```

```
    "ServiceName": "myservice",
    "HealthStatus": "UNKNOWN",
    "Attributes": {
      "AWS_INSTANCE_IPV4": "172.2.1.3",
      "AWS_INSTANCE_PORT": "808"
    }
  ]
}
```

- For API details, see [DiscoverInstances](#) in *AWS CLI Command Reference*.

get-operation

The following code example shows how to use `get-operation`.

AWS CLI

To get the result of an operation

The following `get-operation` example gets the result of an operation.

```
aws servicediscovery get-operation \
  --operation-id gv4g5meo7ndmeh4fqskygvk23d2fijwa-k9302yzd
```

Output:

```
{
  "Operation": {
    "Id": "gv4g5meo7ndmeh4fqskygvk23d2fijwa-k9302yzd",
    "Type": "CREATE_NAMESPACE",
    "Status": "SUCCESS",
    "CreateDate": 1587055860.121,
    "UpdateDate": 1587055900.469,
    "Targets": {
      "NAMESPACE": "ns-ylexjili4cdxy3xm"
    }
  }
}
```

- For API details, see [GetOperation](#) in *AWS CLI Command Reference*.

list-instances

The following code example shows how to use `list-instances`.

AWS CLI

To list service instances

The following `list-instances` example lists service instances.

```
aws servicediscovery list-instances \  
  --service-id srv-qzpwvt2tfqcegapy
```

Output:

```
{  
  "Instances": [  
    {  
      "Id": "i-06bdabbae60f65a4e",  
      "Attributes": {  
        "AWS_INSTANCE_IPV4": "172.2.1.3",  
        "AWS_INSTANCE_PORT": "808"  
      }  
    }  
  ]  
}
```

For more information, see [Viewing a list of service instances](#) in the *AWS Cloud Map Developer Guide*.

- For API details, see [ListInstances](#) in *AWS CLI Command Reference*.

list-namespaces

The following code example shows how to use `list-namespaces`.

AWS CLI

To list namespaces

The following `list-namespaces` example lists namespaces.

```
aws servicediscovery list-namespaces
```

Output:

```
{
  "Namespaces": [
    {
      "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
a3ccy2e7e3a7rile",
      "CreateDate": 1585354387.357,
      "Id": "ns-a3ccy2e7e3a7rile",
      "Name": "local",
      "Properties": {
        "DnsProperties": {
          "HostedZoneId": "Z06752353VBUDTC32S84S"
        },
        "HttpProperties": {
          "HttpName": "local"
        }
      },
      "Type": "DNS_PRIVATE"
    },
    {
      "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
pocfyjtrsmwtvcxx",
      "CreateDate": 1586468974.698,
      "Description": "My second namespace",
      "Id": "ns-pocfyjtrsmwtvcxx",
      "Name": "My-second-namespace",
      "Properties": {
        "DnsProperties": {},
        "HttpProperties": {
          "HttpName": "My-second-namespace"
        }
      },
      "Type": "HTTP"
    },
    {
      "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
ylexjili4cdxy3xm",
      "CreateDate": 1587055896.798,
      "Id": "ns-ylexjili4cdxy3xm",
      "Name": "example.com",
      "Properties": {
        "DnsProperties": {
          "HostedZoneId": "Z09983722P0QME1B3KC8I"
        }
      }
    }
  ]
}
```



```
        },
        "HttpProperties": {
            "HttpName": "example.com"
        }
    },
    "Type": "DNS_PRIVATE"
}
]
```

For more information, see [Viewing a list of namespaces](#) in the *AWS Cloud Map Developer Guide*.

- For API details, see [ListNamespaces](#) in *AWS CLI Command Reference*.

list-services

The following code example shows how to use `list-services`.

AWS CLI

To list services

The following `list-services` example lists services.

```
aws servicediscovery list-services
```

Output:

```
{
  "Services": [
    {
      "Id": "srv-p5zdwlg5uvvzjita",
      "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:service/srv-p5zdwlg5uvvzjita",
      "Name": "myservice",
      "DnsConfig": {
        "RoutingPolicy": "MULTIVALUE",
        "DnsRecords": [
          {
            "Type": "A",
            "TTL": 60
          }
        ]
      }
    }
  ]
}
```

```
    },
    "CreateDate": 1587081768.334
  }
]
}
```

For more information, see [Viewing a list of services](#) in the *AWS Cloud Map Developer Guide*.

- For API details, see [ListServices](#) in *AWS CLI Command Reference*.

register-instance

The following code example shows how to use `register-instance`.

AWS CLI

To register a service instance

The following `register-instance` example registers a service instance.

```
aws servicediscovery register-instance \
  --service-id srv-p5zdwlg5uvvzjita \
  --instance-id myservice-53 \
  --attributes=AWS_INSTANCE_IPV4=172.2.1.3,AWS_INSTANCE_PORT=808
```

Output:

```
{
  "OperationId": "4yejorelbukcjpnr6t1mrghsjwpngf4-k95yg2u7"
}
```

To confirm that the operation succeeded, you can run `get-operation`. For more information, see [get-operation](#).

For more information, see [Registering instances](#) in the *AWS Cloud Map Developer Guide*.

- For API details, see [RegisterInstance](#) in *AWS CLI Command Reference*.

AWS Cloud9 examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Cloud9.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-environment-ec2

The following code example shows how to use `create-environment-ec2`.

AWS CLI

To create an AWS Cloud9 EC2 development environment

This following `create-environment-ec2` example creates an AWS Cloud9 development environment with the specified settings, launches an Amazon Elastic Compute Cloud (Amazon EC2) instance, and then connects from the instance to the environment.

```
aws cloud9 create-environment-ec2 \  
  --name my-demo-env \  
  --description "My demonstration development environment." \  
  --instance-type t2.micro --image-id amazonlinux-2023-x86_64 \  
  --subnet-id subnet-1fab8aEX \  
  --automatic-stop-time-minutes 60 \  
  --owner-arn arn:aws:iam::123456789012:user/MyDemoUser
```

Output:

```
{  
  "environmentId": "8a34f51ce1e04a08882f1e811bd706EX"  
}
```

For more information, see [Creating an EC2 Environment](#) in the *AWS Cloud9 User Guide*.

- For API details, see [CreateEnvironmentEc2](#) in *AWS CLI Command Reference*.

create-environment-membership

The following code example shows how to use `create-environment-membership`.

AWS CLI

To add an environment member to an AWS Cloud9 development environment

This example adds the specified environment member to the specified AWS Cloud9 development environment.

Command:

```
aws cloud9 create-environment-membership --environment-id
8a34f51ce1e04a08882f1e811bd706EX --user-arn arn:aws:iam::123456789012:user/
AnotherDemoUser --permissions read-write
```

Output:

```
{
  "membership": {
    "environmentId": "8a34f51ce1e04a08882f1e811bd706EX",
    "userId": "AIDAJ3LOROMOUCTBSUGEX",
    "userArn": "arn:aws:iam::123456789012:user/AnotherDemoUser",
    "permissions": "read-write"
  }
}
```

- For API details, see [CreateEnvironmentMembership](#) in *AWS CLI Command Reference*.

delete-environment-membership

The following code example shows how to use `delete-environment-membership`.

AWS CLI

To delete an environment member from an AWS Cloud9 development environment

This example deletes the specified environment member from the specified AWS Cloud9 development environment.

Command:

```
aws cloud9 delete-environment-membership --environment-id
8a34f51ce1e04a08882f1e811bd706EX --user-arn arn:aws:iam::123456789012:user/
AnotherDemoUser
```

Output:

```
None.
```

- For API details, see [DeleteEnvironmentMembership](#) in *AWS CLI Command Reference*.

delete-environment

The following code example shows how to use delete-environment.

AWS CLI

To delete an AWS Cloud9 development environment

This example deletes the specified AWS Cloud9 development environment. If an Amazon EC2 instance is connected to the environment, also terminates the instance.

Command:

```
aws cloud9 delete-environment --environment-id 8a34f51ce1e04a08882f1e811bd706EX
```

Output:

```
None.
```

- For API details, see [DeleteEnvironment](#) in *AWS CLI Command Reference*.

describe-environment-memberships

The following code example shows how to use describe-environment-memberships.

AWS CLI

To get information about environment members for an AWS Cloud9 development environment

This example gets information about environment members for the specified AWS Cloud9 development environment.

Command:

```
aws cloud9 describe-environment-memberships --environment-id
8a34f51ce1e04a08882f1e811bd706EX
```

Output:

```
{
  "memberships": [
    {
      "environmentId": "8a34f51ce1e04a08882f1e811bd706EX",
      "userId": "AIDAJ3LOROMOUCTBSU6EX",
      "userArn": "arn:aws:iam::123456789012:user/AnotherDemoUser",
      "permissions": "read-write"
    },
    {
      "environmentId": "8a34f51ce1e04a08882f1e811bd706EX",
      "userId": "AIDAJNUEDQAQWFELJDLEX",
      "userArn": "arn:aws:iam::123456789012:user/MyDemoUser",
      "permissions": "owner"
    }
  ]
}
```

To get information about the owner of an AWS Cloud9 development environment

This example gets information about the owner of the specified AWS Cloud9 development environment.

Command:

```
aws cloud9 describe-environment-memberships --environment-id
8a34f51ce1e04a08882f1e811bd706EX --permissions owner
```

Output:

```
{
  "memberships": [
    {
      "environmentId": "8a34f51ce1e04a08882f1e811bd706EX",
      "userId": "AIDAJNUEDQAQWFELJDLEX",
      "userArn": "arn:aws:iam::123456789012:user/MyDemoUser",
      "permissions": "owner"
    }
  ]
}
```

To get information about an environment member for multiple AWS Cloud9 development environments

This example gets information about the specified environment member for multiple AWS Cloud9 development environments.

Command:

```
aws cloud9 describe-environment-memberships --user-arn
arn:aws:iam::123456789012:user/MyDemoUser
```

Output:

```
{
  "memberships": [
    {
      "environmentId": "10a75714bd494714929e7f5ec4125aEX",
      "lastAccess": 1516213427.0,
      "userId": "AIDAJNUEDQAQWFELJDLEX",
      "userArn": "arn:aws:iam::123456789012:user/MyDemoUser",
      "permissions": "owner"
    },
    {
      "environmentId": "1980b80e5f584920801c09086667f0EX",
      "lastAccess": 1516144884.0,
      "userId": "AIDAJNUEDQAQWFELJDLEX",
      "userArn": "arn:aws:iam::123456789012:user/MyDemoUser",
      "permissions": "owner"
    }
  ]
}
```

```
}
```

- For API details, see [DescribeEnvironmentMemberships](#) in *AWS CLI Command Reference*.

describe-environment-status

The following code example shows how to use `describe-environment-status`.

AWS CLI

To get status information for an AWS Cloud9 development environment

This example gets status information for the specified AWS Cloud9 development environment.

Command:

```
aws cloud9 describe-environment-status --environment-id
685f892f431b45c2b28cb69eadcdb0EX
```

Output:

```
{
  "status": "ready",
  "message": "Environment is ready to use"
}
```

- For API details, see [DescribeEnvironmentStatus](#) in *AWS CLI Command Reference*.

describe-environments

The following code example shows how to use `describe-environments`.

AWS CLI

To get information about AWS Cloud9 development environments

This example gets information about the specified AWS Cloud9 development environments.

Command:

```
aws cloud9 describe-environments --environment-ids 685f892f431b45c2b28cb69eadcdb0EX
349c86d4579e4e7298d500ff57a6b2EX
```


Output:

```
{
  "environments": [
    {
      "id": "685f892f431b45c2b28cb69eadcdb0EX",
      "name": "my-demo-ec2-env",
      "description": "Created from CodeStar.",
      "type": "ec2",
      "arn": "arn:aws:cloud9:us-east-1:123456789012:environment:685f892f431b45c2b28cb69eadcdb0EX",
      "ownerArn": "arn:aws:iam::123456789012:user/MyDemoUser",
      "lifecycle": {
        "status": "CREATED"
      }
    },
    {
      "id": "349c86d4579e4e7298d500ff57a6b2EX",
      "name": "my-demo-ssh-env",
      "description": "",
      "type": "ssh",
      "arn": "arn:aws:cloud9:us-east-1:123456789012:environment:349c86d4579e4e7298d500ff57a6b2EX",
      "ownerArn": "arn:aws:iam::123456789012:user/MyDemoUser",
      "lifecycle": {
        "status": "CREATED"
      }
    }
  ]
}
```

- For API details, see [DescribeEnvironments](#) in *AWS CLI Command Reference*.

list-environments

The following code example shows how to use `list-environments`.

AWS CLI**To get a list of available AWS Cloud9 development environment identifiers**

This example gets a list of available AWS Cloud9 development environment identifiers.

Command:

```
aws cloud9 list-environments
```

Output:

```
{
  "environmentIds": [
    "685f892f431b45c2b28cb69eadcdb0EX",
    "1980b80e5f584920801c09086667f0EX"
  ]
}
```

- For API details, see [ListEnvironments](#) in *AWS CLI Command Reference*.

update-environment-membership

The following code example shows how to use `update-environment-membership`.

AWS CLI**To change the settings of an existing environment member for an AWS Cloud9 development environment**

This example changes the settings of the specified existing environment member for the specified AWS Cloud9 development environment.

Command:

```
aws cloud9 update-environment-membership --environment-id
8a34f51ce1e04a08882f1e811bd706EX --user-arn arn:aws:iam::123456789012:user/
AnotherDemoUser --permissions read-only
```

Output:

```
{
  "membership": {
    "environmentId": "8a34f51ce1e04a08882f1e811bd706EX",
    "userId": "AIDAJ3LOROMOUCTBSU6EX",
    "userArn": "arn:aws:iam::123456789012:user/AnotherDemoUser",
  }
}
```

```
    "permissions": "read-only"  
  }  
}
```

- For API details, see [UpdateEnvironmentMembership](#) in *AWS CLI Command Reference*.

update-environment

The following code example shows how to use `update-environment`.

AWS CLI

To change the settings of an existing AWS Cloud9 development environment

This example changes the specified settings of the specified existing AWS Cloud9 development environment.

Command:

```
aws cloud9 update-environment --environment-id 8a34f51ce1e04a08882f1e811bd706EX  
--name my-changed-demo-env --description "My changed demonstration development  
environment."
```

Output:

```
None.
```

- For API details, see [UpdateEnvironment](#) in *AWS CLI Command Reference*.

AWS CloudFormation examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS CloudFormation.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

activate-type

The following code example shows how to use `activate-type`.

AWS CLI

To activate a type

The following `activate-type` example activates a public third-party extension, making it available for use in stack templates.

```
aws cloudformation activate-type \  
  --region us-west-2 \  
  --type RESOURCE \  
  --type-name Example::Test::1234567890abcdef0 \  
  --type-name-alias Example::Test::Alias
```

Output:

```
{  
  "Arn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/Example-  
Test-Alias"  
}
```

For more information, see [Using the AWS CloudFormation registry](#) in the *AWS CloudFormation User Guide*.

- For API details, see [ActivateType](#) in *AWS CLI Command Reference*.

batch-describe-type-configurations

The following code example shows how to use `batch-describe-type-configurations`.

AWS CLI

To batch describe a type configuration

The following `batch-describe-type-configurations` example configures the data for the type.

```
aws cloudformation batch-describe-type-configurations \  
  --region us-west-2 \  
  --type-configuration-identifiers TypeArn="arn:aws:cloudformation:us-  
west-2:123456789012:type/resource/Example-Test-  
Type,TypeConfigurationAlias=MyConfiguration"
```

Output:

```
{  
  "Errors": [],  
  "UnprocessedTypeConfigurations": [],  
  "TypeConfigurations": [  
    {  
      "Arn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/  
Example-Test-Type",  
      "Alias": "MyConfiguration",  
      "Configuration": "{\n      \"Example\": {\n          \"ApiKey\":  
\"examplekey\",  
          \"ApplicationKey\": \"examplekey1\",  
          \"ApiURL\": \"exampleurl\"\n        }\n    }",  
      "LastUpdated": "2021-10-01T15:25:46.210000+00:00",  
      "TypeArn": "arn:aws:cloudformation:us-east-1:123456789012:type/resource/  
Example-Test-Type"  
    }  
  ]  
}
```

For more information, see [Using the AWS CloudFormation registry](#) in the *AWS CloudFormation User Guide*.

- For API details, see [BatchDescribeTypeConfigurations](#) in *AWS CLI Command Reference*.

`cancel-update-stack`

The following code example shows how to use `cancel-update-stack`.

AWS CLI

To cancel a stack update that is in progress

The following `cancel-update-stack` command cancels a stack update on the `myteststack` stack:

```
aws cloudformation cancel-update-stack --stack-name myteststack
```

- For API details, see [CancelUpdateStack](#) in *AWS CLI Command Reference*.

continue-update-rollback

The following code example shows how to use `continue-update-rollback`.

AWS CLI

To retry an update rollback

The following `continue-update-rollback` example resumes a rollback operation from a previously failed stack update.

```
aws cloudformation continue-update-rollback \  
  --stack-name my-stack
```

This command produces no output.

- For API details, see [ContinueUpdateRollback](#) in *AWS CLI Command Reference*.

create-change-set

The following code example shows how to use `create-change-set`.

AWS CLI

To create a change set

The following `create-change-set` example creates a change set with the `CAPABILITY_IAM` capability. The file `template.yaml` is an AWS CloudFormation template in the current folder that defines a stack that includes IAM resources.

```
aws cloudformation create-change-set \  
  --stack-name my-application \  
  --change-set-name my-change-set \  
  --template-body file://template.yaml \  
  --capabilities CAPABILITY_IAM
```

Output:

```
{  
  "Id": "arn:aws:cloudformation:us-west-2:123456789012:changeSet/my-change-set/  
bc9555ba-a949-xmpl-bfb8-f41d04ec5784",  
  "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-application/  
d0a825a0-e4cd-xmpl-b9fb-061c69e99204"  
}
```

- For API details, see [CreateChangeSet](#) in *AWS CLI Command Reference*.

create-stack-instances

The following code example shows how to use `create-stack-instances`.

AWS CLI

To create stack instances

The following `create-stack-instances` example creates instances of a stack set in two accounts and in four regions. The `fault tolerance` setting ensures that the update is attempted in all accounts and regions, even if some stacks cannot be created.

```
aws cloudformation create-stack-instances \  
  --stack-set-name my-stack-set \  
  --accounts 123456789012 223456789012 \  
  --regions us-east-1 us-east-2 us-west-1 us-west-2 \  
  --operation-preferences FailureToleranceCount=7
```

Output:

```
{  
  "OperationId": "d7995c31-83c2-xmpl-a3d4-e9ca2811563f"  
}
```

To create a stack set, use the `create-stack-set` command.

- For API details, see [CreateStackInstances](#) in *AWS CLI Command Reference*.

create-stack-set

The following code example shows how to use `create-stack-set`.

AWS CLI

To create a stack set

The following `create-stack-set` example creates a stack set using the specified YAML file `template.yaml`. `template.yaml` is an AWS CloudFormation template in the current folder that defines a stack.

```
aws cloudformation create-stack-set \  
  --stack-set-name my-stack-set \  
  --template-body file://template.yaml \  
  --description "SNS topic"
```

Output:

```
{  
  "StackSetId": "my-stack-set:8d0f160b-d157-xmpl-a8e6-c0ce8e5d8cc1"  
}
```

To add stack instances to the stack set, use the `create-stack-instances` command.

- For API details, see [CreateStackSet](#) in *AWS CLI Command Reference*.

create-stack

The following code example shows how to use `create-stack`.

AWS CLI

To create an AWS CloudFormation stack

The following `create-stacks` command creates a stack with the name `myteststack` using the `sampletemplate.json` template:


```
aws cloudformation create-stack --stack-name myteststack --template-body file://
sampletemplate.json --parameters ParameterKey=KeyPairName,ParameterValue=TestKey
ParameterKey=SubnetIDs,ParameterValue=SubnetID1\\,SubnetID2
```

Output:

```
{
  "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896"
}
```

For more information, see *Stacks* in the *AWS CloudFormation User Guide*.

- For API details, see [CreateStack](#) in *AWS CLI Command Reference*.

deactivate-type

The following code example shows how to use `deactivate-type`.

AWS CLI

To deactivate a type

The following `deactivate-type` example deactivates a public extension that was previously activated in this account and Region.

```
aws cloudformation deactivate-type \
  --region us-west-2 \
  --type MODULE \
  --type-name Example::Test::Type::MODULE
```

This command produces no output.

For more information, see [Using the AWS CloudFormation registry](#) in the *AWS CloudFormation User Guide*.

- For API details, see [DeactivateType](#) in *AWS CLI Command Reference*.

delete-change-set

The following code example shows how to use `delete-change-set`.

AWS CLI

To delete a change set

The following `delete-change-set` example deletes a change set by specifying the change set name and stack name.

```
aws cloudformation delete-change-set \  
  --stack-name my-stack \  
  --change-set-name my-change-set
```

This command produces no output.

The following `delete-change-set` example deletes a change set by specifying the full ARN of the change set.

```
aws cloudformation delete-change-set \  
  --change-set-name arn:aws:cloudformation:us-east-2:123456789012:changeSet/my-  
change-set/4eca1a01-e285-xmpl-8026-9a1967bfb4b0
```

This command produces no output.

- For API details, see [DeleteChangeSet](#) in *AWS CLI Command Reference*.

delete-stack-instances

The following code example shows how to use `delete-stack-instances`.

AWS CLI

To delete stack instances

The following `delete-stack-instances` example deletes instances of a stack set in two accounts in two regions and terminates the stacks.

```
aws cloudformation delete-stack-instances \  
  --stack-set-name my-stack-set \  
  --accounts 123456789012 567890123456 \  
  --regions us-east-1 us-west-1 \  
  --no-retain-stacks
```

Output:

```
{
  "OperationId": "ad49f10c-fd1d-413f-a20a-8de6e2fa8f27"
}
```

To delete an empty stack set, use the `delete-stack-set` command.

- For API details, see [DeleteStackInstances](#) in *AWS CLI Command Reference*.

delete-stack-set

The following code example shows how to use `delete-stack-set`.

AWS CLI**To delete a stack set**

The following command deletes the specified empty stack set. The stack set must be empty.

```
aws cloudformation delete-stack-set \
  --stack-set-name my-stack-set
```

This command produces no output.

To delete instances from the stack set, use the `delete-stack-instances` command.

- For API details, see [DeleteStackSet](#) in *AWS CLI Command Reference*.

delete-stack

The following code example shows how to use `delete-stack`.

AWS CLI**To delete a stack**

The following `delete-stack` example deletes the specified stack.

```
aws cloudformation delete-stack \
```

```
--stack-name my-stack
```

This command produces no output.

- For API details, see [DeleteStack](#) in *AWS CLI Command Reference*.

deploy

The following code example shows how to use `deploy`.

AWS CLI

Following command deploys template named `template.json` to a stack named `my-new-stack`:

```
aws cloudformation deploy --template-file /path_to_template/template.json --stack-name my-new-stack --parameter-overrides Key1=Value1 Key2=Value2 --tags Key1=Value1 Key2=Value2
```

- For API details, see [Deploy](#) in *AWS CLI Command Reference*.

deregister-type

The following code example shows how to use `deregister-type`.

AWS CLI

To deregister a type version

The following `deregister-type` example removes the specified type version from active use in the CloudFormation registry, so that it can no longer be used in CloudFormation operations.

```
aws cloudformation deregister-type \  
  --type RESOURCE \  
  --type-name My::Logs::LogGroup \  
  --version-id 00000002
```

This command produces no output.

For more information, see [Using the CloudFormation Registry](#) in the *AWS CloudFormation Users Guide*.

- For API details, see [DeregisterType](#) in *AWS CLI Command Reference*.

describe-account-limits

The following code example shows how to use `describe-account-limits`.

AWS CLI

To get information about your account limits

The following command retrieves a list of regional limits for the current account.

```
aws cloudformation describe-account-limits
```

Output:

```
{
  "AccountLimits": [
    {
      "Name": "StackLimit",
      "Value": 200
    },
    {
      "Name": "StackOutputsLimit",
      "Value": 60
    },
    {
      "Name": "ConcurrentResourcesLimit",
      "Value": 2500
    }
  ]
}
```

- For API details, see [DescribeAccountLimits](#) in *AWS CLI Command Reference*.

describe-change-set

The following code example shows how to use `describe-change-set`.

AWS CLI

To get information about a change set

The following `describe-change-set` example displays the details of the change set specified by change set name and stack name.

```
aws cloudformation describe-change-set \  
  --change-set-name my-change-set \  
  --stack-name my-stack
```

The following `describe-change-set` example displays the details of the change set specified by the full ARN of the change set:

```
aws cloudformation describe-change-set \  
  --change-set-name arn:aws:cloudformation:us-west-2:123456789012:changeSet/my-  
change-set/bc9555ba-a949-xmpl-bfb8-f41d04ec5784
```

Output:

```
{  
  "Changes": [  
    {  
      "Type": "Resource",  
      "ResourceChange": {  
        "Action": "Modify",  
        "LogicalResourceId": "function",  
        "PhysicalResourceId": "my-function-SEZV4XMPL4S5",  
        "ResourceType": "AWS::Lambda::Function",  
        "Replacement": "False",  
        "Scope": [  
          "Properties"  
        ],  
        "Details": [  
          {  
            "Target": {  
              "Attribute": "Properties",  
              "Name": "Timeout",  
              "RequiresRecreation": "Never"  
            },  
            "Evaluation": "Static",  
            "ChangeSource": "DirectModification"  
          }  
        ]  
      }  
    ]  
  }  
}
```

```

    ],
    "ChangeSetName": "my-change-set",
    "ChangeSetId": "arn:aws:cloudformation:us-west-2:123456789012:changeSet/my-
change-set/4eca1a01-e285-xmpl-8026-9a1967bfb4b0",
    "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-stack/
d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
    "StackName": "my-stack",
    "Description": null,
    "Parameters": null,
    "CreationTime": "2019-10-02T05:20:56.651Z",
    "ExecutionStatus": "AVAILABLE",
    "Status": "CREATE_COMPLETE",
    "StatusReason": null,
    "NotificationARNs": [],
    "RollbackConfiguration": {},
    "Capabilities": [
        "CAPABILITY_IAM"
    ],
    "Tags": null
}

```

- For API details, see [DescribeChangeSet](#) in *AWS CLI Command Reference*.

describe-publisher

The following code example shows how to use `describe-publisher`.

AWS CLI

To describe a publisher

The following `describe-publisher` example configures the information for a publisher.

```

aws cloudformation describe-publisher \
  --region us-west-2 \
  --publisher-id 000q6TfUovXsEMmgKowxDZLLwqr2QUsh

```

Output:

```

{
  "PublisherId": "000q6TfUovXsEMmgKowxDZLLwqr2QUshd2e75c8c",
  "PublisherStatus": "VERIFIED",

```

```
"IdentityProvider": "AWS_Marketplace",
  "PublisherProfile": "https://aws.amazon.com/marketplace/seller-profile?
id=2c5dc1f0-17cd-4259-8e46-822a83gdtegd"
}
```

For more information, see [Using the AWS CloudFormation registry](#) in the *AWS CloudFormation User Guide*.

- For API details, see [DescribePublisher](#) in *AWS CLI Command Reference*.

describe-stack-drift-detection-status

The following code example shows how to use `describe-stack-drift-detection-status`.

AWS CLI

To check a drift detection operation's status

The following `describe-stack-drift-detection-status` example displays the status of a drift detection operation. Get the by ID running the `detect-stack-drift` command.

```
aws cloudformation describe-stack-drift-detection-status \
  --stack-drift-detection-id 1a229160-e4d9-xmpl-ab67-0a4f93df83d4
```

Output:

```
{
  "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-stack/
d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
  "StackDriftDetectionId": "1a229160-e4d9-xmpl-ab67-0a4f93df83d4",
  "StackDriftStatus": "DRIFTED",
  "DetectionStatus": "DETECTION_COMPLETE",
  "DriftedStackResourceCount": 1,
  "Timestamp": "2019-10-02T05:54:30.902Z"
}
```

- For API details, see [DescribeStackDriftDetectionStatus](#) in *AWS CLI Command Reference*.

describe-stack-events

The following code example shows how to use `describe-stack-events`.

AWS CLI

To describe stack events

The following `describe-stack-events` example displays the 2 most recent events for the specified stack.

```
aws cloudformation describe-stack-events \  
  --stack-name my-stack \  
  --max-items 2  
  
{  
  "StackEvents": [  
    {  
      "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-  
stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",  
      "EventId": "4e1516d0-e4d6-xmpl-b94f-0a51958a168c",  
      "StackName": "my-stack",  
      "LogicalResourceId": "my-stack",  
      "PhysicalResourceId": "arn:aws:cloudformation:us-  
west-2:123456789012:stack/my-stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",  
      "ResourceType": "AWS::CloudFormation::Stack",  
      "Timestamp": "2019-10-02T05:34:29.556Z",  
      "ResourceStatus": "UPDATE_COMPLETE"  
    },  
    {  
      "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-  
stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",  
      "EventId": "4dd3c810-e4d6-xmpl-bade-0aaf8b31ab7a",  
      "StackName": "my-stack",  
      "LogicalResourceId": "my-stack",  
      "PhysicalResourceId": "arn:aws:cloudformation:us-  
west-2:123456789012:stack/my-stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",  
      "ResourceType": "AWS::CloudFormation::Stack",  
      "Timestamp": "2019-10-02T05:34:29.127Z",  
      "ResourceStatus": "UPDATE_COMPLETE_CLEANUP_IN_PROGRESS"  
    }  
  ],  
  "NextToken": "eyJ0ZXh0VG9XMPLiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAyfQ=="  
}
```

- For API details, see [DescribeStackEvents](#) in *AWS CLI Command Reference*.

describe-stack-instance

The following code example shows how to use `describe-stack-instance`.

AWS CLI

To describe a stack instance

The following command describes an instance of the specified stack set in the specified account and Region. The stack set is in the current region and account, and the instance is in the `us-west-2` region in account `123456789012`:

```
aws cloudformation describe-stack-instance \  
  --stack-set-name my-stack-set \  
  --stack-instance-account 123456789012 \  
  --stack-instance-region us-west-2
```

Output:

```
{  
  "StackInstance": {  
    "StackSetId": "enable-config:296a3360-xmpl-40af-be78-9341e95bf743",  
    "Region": "us-west-2",  
    "Account": "123456789012",  
    "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/  
StackSet-enable-config-e6cac20f-xmpl-46e9-8314-53e0d4591532/4287f9a0-e615-  
xmpl-894a-12b31d3117be",  
    "ParameterOverrides": [],  
    "Status": "OUTDATED",  
    "StatusReason": "ResourceLogicalId:ConfigBucket,  
ResourceType:AWS::S3::Bucket, ResourceStatusReason:You have attempted to create  
more buckets than allowed (Service: Amazon S3; Status Code: 400; Error Code:  
TooManyBuckets; Request ID: F7F21CXMPL580224; S3 Extended Request ID: egd/  
Fdt89BXMPLYiqbMNLjVk55Yqqvi3NYW2nKLUVWhUGEHnFcmZdyj967lhriaG/dWMobS040o=)."  
  }  
}
```

- For API details, see [DescribeStackInstance](#) in *AWS CLI Command Reference*.

describe-stack-resource-drifts

The following code example shows how to use `describe-stack-resource-drifts`.

AWS CLI

To get information about resources that drifted from the stack definition

The following command displays information about drifted resources for the specified stack. To initiate drift detection, use the `detect-stack-drift` command:

```
aws cloudformation describe-stack-resource-drifts \
  --stack-name my-stack
```

The output shows an AWS Lambda function that was modified out-of-band:

```
{
  "StackResourceDrifts": [
    {
      "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
      "LogicalResourceId": "function",
      "PhysicalResourceId": "my-function-SEZV4XMPL4S5",
      "ResourceType": "AWS::Lambda::Function",
      "ExpectedProperties": "{\"Description\":\"Write a file to S3.\",
        \"Environment\":{\"Variables\":{\"bucket\":\"my-stack-bucket-1vc62xmplgguf\"}},
        \"Handler\":\"index.handler\", \"MemorySize\":128, \"Role\":
        \"arn:aws:iam::123456789012:role/my-functionRole-HIZXMPLEOM9E\",
        \"Runtime\":\"nodejs10.x\", \"Tags\":[{\"Key\":\"lambda:createdBy\",
        \"Value\":\"SAM\"}], \"Timeout\":900,
        \"TracingConfig\":{\"Mode\":\"Active\"}}",
      "ActualProperties": "{\"Description\":\"Write a file to S3.\",
        \"Environment\":{\"Variables\":{\"bucket\":\"my-stack-bucket-1vc62xmplgguf\"}},
        \"Handler\":\"index.handler\", \"MemorySize\":256, \"Role\":
        \"arn:aws:iam::123456789012:role/my-functionRole-HIZXMPLEOM9E\",
        \"Runtime\":\"nodejs10.x\", \"Tags\":[{\"Key\":\"lambda:createdBy\",
        \"Value\":\"SAM\"}], \"Timeout\":22,
        \"TracingConfig\":{\"Mode\":\"Active\"}}",
      "PropertyDifferences": [
        {
          "PropertyPath": "/MemorySize",
          "ExpectedValue": "128",
          "ActualValue": "256",
          "DifferenceType": "NOT_EQUAL"
        },
        {
          "PropertyPath": "/Timeout",
          "ExpectedValue": "900",
          "ActualValue": "22",
        }
      ]
    }
  ]
}
```

```

        "DifferenceType": "NOT_EQUAL"
      }
    ],
    "StackResourceDriftStatus": "MODIFIED",
    "Timestamp": "2019-10-02T05:54:44.064Z"
  }
]
}

```

- For API details, see [DescribeStackResourceDrifts](#) in *AWS CLI Command Reference*.

describe-stack-resource

The following code example shows how to use `describe-stack-resource`.

AWS CLI

To get information about a stack resource

The following `describe-stack-resource` example displays details for the resource named `MyFunction` in the specified stack.

```

aws cloudformation describe-stack-resource \
  --stack-name MyStack \
  --logical-resource-id MyFunction

```

Output:

```

{
  "StackResourceDetail": {
    "StackName": "MyStack",
    "StackId": "arn:aws:cloudformation:us-east-2:123456789012:stack/MyStack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
    "LogicalResourceId": "MyFunction",
    "PhysicalResourceId": "my-function-SEZV4XMPL4S5",
    "ResourceType": "AWS::Lambda::Function",
    "LastUpdatedTimestamp": "2019-10-02T05:34:27.989Z",
    "ResourceStatus": "UPDATE_COMPLETE",
    "Metadata": "{}",
    "DriftInformation": {
      "StackResourceDriftStatus": "IN_SYNC"
    }
  }
}

```

```
    }  
  }  
}
```

- For API details, see [DescribeStackResource](#) in *AWS CLI Command Reference*.

describe-stack-resources

The following code example shows how to use `describe-stack-resources`.

AWS CLI

To get information about a stack resource

The following `describe-stack-resources` example displays details for the resources in the specified stack.

```
aws cloudformation describe-stack-resources \  
  --stack-name my-stack
```

Output:

```
{  
  "StackResources": [  
    {  
      "StackName": "my-stack",  
      "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-  
stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",  
      "LogicalResourceId": "bucket",  
      "PhysicalResourceId": "my-stack-bucket-1vc62xmplgguf",  
      "ResourceType": "AWS::S3::Bucket",  
      "Timestamp": "2019-10-02T04:34:11.345Z",  
      "ResourceStatus": "CREATE_COMPLETE",  
      "DriftInformation": {  
        "StackResourceDriftStatus": "IN_SYNC"  
      }  
    },  
    {  
      "StackName": "my-stack",  
      "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-  
stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",  
      "LogicalResourceId": "function",
```

```

    "PhysicalResourceId": "my-function-SEZV4XMPL4S5",
    "ResourceType": "AWS::Lambda::Function",
    "Timestamp": "2019-10-02T05:34:27.989Z",
    "ResourceStatus": "UPDATE_COMPLETE",
    "DriftInformation": {
      "StackResourceDriftStatus": "IN_SYNC"
    }
  },
  {
    "StackName": "my-stack",
    "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-
stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
    "LogicalResourceId": "functionRole",
    "PhysicalResourceId": "my-functionRole-HIZXMPLE0M9E",
    "ResourceType": "AWS::IAM::Role",
    "Timestamp": "2019-10-02T04:34:06.350Z",
    "ResourceStatus": "CREATE_COMPLETE",
    "DriftInformation": {
      "StackResourceDriftStatus": "IN_SYNC"
    }
  }
]
}

```

- For API details, see [DescribeStackResources](#) in *AWS CLI Command Reference*.

describe-stack-set-operation

The following code example shows how to use `describe-stack-set-operation`.

AWS CLI

To get information about a stack set operation

The following `describe-stack-set-operation`` example displays details for an update operation on the specified stack set.

```

aws cloudformation describe-stack-set-operation \
  --stack-set-name enable-config \
  --operation-id 35d45ebc-ed88-xmpl-ab59-0197a1fc83a0

```

Output:

```
{
  "StackSetOperation": {
    "OperationId": "35d45ebc-ed88-xmpl-ab59-0197a1fc83a0",
    "StackSetId": "enable-config:296a3360-xmpl-40af-be78-9341e95bf743",
    "Action": "UPDATE",
    "Status": "SUCCEEDED",
    "OperationPreferences": {
      "RegionOrder": [
        "us-east-1",
        "us-west-2",
        "eu-west-1",
        "us-west-1"
      ],
      "FailureToleranceCount": 7,
      "MaxConcurrentCount": 2
    },
    "AdministrationRoleARN": "arn:aws:iam::123456789012:role/
AWSCloudFormationStackSetAdministrationRole",
    "ExecutionRoleName": "AWSCloudFormationStackSetExecutionRole",
    "CreationTimestamp": "2019-10-03T16:28:44.377Z",
    "EndTimestamp": "2019-10-03T16:42:08.607Z"
  }
}
```

- For API details, see [DescribeStackSetOperation](#) in *AWS CLI Command Reference*.

describe-stack-set

The following code example shows how to use describe-stack-set.

AWS CLI

To get information about a stack set

The following describe-stack-set` example displays details about the specified stack set.

```
aws cloudformation describe-stack-set \
  --stack-set-name my-stack-set
```

Output:

```
{
```

```

"StackSet": {
  "StackSetName": "my-stack-set",
  "StackSetId": "my-stack-set:296a3360-xmpl-40af-be78-9341e95bf743",
  "Description": "Create an Amazon SNS topic",
  "Status": "ACTIVE",
  "TemplateBody": "AWSTemplateFormatVersion: '2010-09-09'\nDescription: An AWS
SNS topic\nResources:\n  topic:\n    Type: AWS::SNS::Topic",
  "Parameters": [],
  "Capabilities": [],
  "Tags": [],
  "StackSetARN": "arn:aws:cloudformation:us-west-2:123456789012:stackset/
enable-config:296a3360-xmpl-40af-be78-9341e95bf743",
  "AdministrationRoleARN": "arn:aws:iam::123456789012:role/
AWSCloudFormationStackSetAdministrationRole",
  "ExecutionRoleName": "AWSCloudFormationStackSetExecutionRole"
}
}

```

- For API details, see [DescribeStackSet](#) in *AWS CLI Command Reference*.

describe-stacks

The following code example shows how to use describe-stacks.

AWS CLI

To describe AWS CloudFormation stacks

The following describe-stacks command shows summary information for the myteststack stack:

```
aws cloudformation describe-stacks --stack-name myteststack
```

Output:

```

{
  "Stacks": [
    {
      "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896",
      "Description": "AWS CloudFormation Sample Template S3_Bucket: Sample
template showing how to create a publicly accessible S3 bucket. **WARNING** This

```



```

template creates an S3 bucket. You will be billed for the AWS resources used if you
create a stack from this template.",
  "Tags": [],
  "Outputs": [
    {
      "Description": "Name of S3 bucket to hold website content",
      "OutputKey": "BucketName",
      "OutputValue": "myteststack-s3bucket-jssofilzie2w"
    }
  ],
  "StackStatusReason": null,
  "CreationTime": "2013-08-23T01:02:15.422Z",
  "Capabilities": [],
  "StackName": "myteststack",
  "StackStatus": "CREATE_COMPLETE",
  "DisableRollback": false
}
]
}

```

For more information, see *Stacks* in the *AWS CloudFormation User Guide*.

- For API details, see [DescribeStacks](#) in *AWS CLI Command Reference*.

describe-type-registration

The following code example shows how to use `describe-type-registration`.

AWS CLI

To display type registration information

The following `describe-type-registration` example displays information about the specified type registration, including the type's current status, type, and version.

```

aws cloudformation describe-type-registration \
  --registration-token a1b2c3d4-5678-90ab-cdef-EXAMPLE11111

```

Output:

```

{
  "ProgressStatus": "COMPLETE",

```

```

    "TypeArn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/My-Logs-LogGroup",
    "Description": "Deployment is currently in DEPLOY_STAGE of status COMPLETED; ",
    "TypeVersionArn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/My-Logs-LogGroup/00000001"
  }

```

For more information, see [Using the CloudFormation Registry](#) in the *AWS CloudFormation Users Guide*.

- For API details, see [DescribeTypeRegistration](#) in *AWS CLI Command Reference*.

describe-type

The following code example shows how to use `describe-type`.

AWS CLI

To display type information

The following `describe-type` example displays information for the specified type.

```

aws cloudformation describe-type \
  --type-name My::Logs::LogGroup \
  --type RESOURCE

```

Output:

```

{
  "SourceUrl": "https://github.com/aws-cloudformation/aws-cloudformation-resource-providers-logs.git",
  "Description": "Customized resource derived from AWS::Logs::LogGroup",
  "TimeCreated": "2019-12-03T23:29:33.321Z",
  "Visibility": "PRIVATE",
  "TypeName": "My::Logs::LogGroup",
  "LastUpdated": "2019-12-03T23:29:33.321Z",
  "DeprecatedStatus": "LIVE",
  "ProvisioningType": "FULLY_MUTABLE",
  "Type": "RESOURCE",
  "Arn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/My-Logs-LogGroup/00000001",
  "Schema": "[details omitted]"
}

```

```
}
```

For more information, see [Using the CloudFormation Registry](#) in the *AWS CloudFormation Users Guide*.

- For API details, see [DescribeType](#) in *AWS CLI Command Reference*.

detect-stack-drift

The following code example shows how to use `detect-stack-drift`.

AWS CLI

To detect drifted resources

The following `detect-stack-drift` example initiates drift detection for the specified stack.

```
aws cloudformation detect-stack-drift \  
  --stack-name my-stack
```

Output:

```
{  
  "StackDriftDetectionId": "1a229160-e4d9-xmpl-ab67-0a4f93df83d4"  
}
```

You can then use this ID with the `describe-stack-resource-drifts` command to describe drifted resources.

- For API details, see [DetectStackDrift](#) in *AWS CLI Command Reference*.

detect-stack-resource-drift

The following code example shows how to use `detect-stack-resource-drift`.

AWS CLI

To detect drift for a resource

The following `detect-stack-resource-drift` example checks a resource named `MyFunction` in a stack named `MyStack` for drift:

```
aws cloudformation detect-stack-resource-drift \
  --stack-name MyStack \
  --logical-resource-id MyFunction
```

The output shows an AWS Lambda function that was modified out-of-band:

```
{
  "StackResourceDrift": {
    "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/MyStack/
d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
    "LogicalResourceId": "MyFunction",
    "PhysicalResourceId": "my-function-SEZV4XMPL4S5",
    "ResourceType": "AWS::Lambda::Function",
    "ExpectedProperties": "{\"Description\":\"Write a file to S3.\",
\\\"Environment\\\":{\\\"Variables\\\":{\\\"bucket\\\":\\\"my-stack-bucket-1vc62xmplgguf
\\\"}},\\\"Handler\\\":\\\"index.handler\\\",\\\"MemorySize\\\":128,\\\"Role\\\":
\\\"arn:aws:iam::123456789012:role/my-functionRole-HIZXMPLE0M9E\\\",\\\"Runtime\\\":
\\\"nodejs10.x\\\",\\\"Tags\\\":[{\\\"Key\\\":\\\"lambda:createdBy\\\",\\\"Value\\\":\\\"SAM\\\"}],\\\"Timeout
\\\":900,\\\"TracingConfig\\\":{\\\"Mode\\\":\\\"Active\\\"}}\",
    "ActualProperties": "{\"Description\":\"Write a file to S3.\",\\\"Environment
\\\":{\\\"Variables\\\":{\\\"bucket\\\":\\\"my-stack-bucket-1vc62xmplgguf\\\"}},\\\"Handler\\\":
\\\"index.handler\\\",\\\"MemorySize\\\":256,\\\"Role\\\":\\\"arn:aws:iam::123456789012:role/
my-functionRole-HIZXMPLE0M9E\\\",\\\"Runtime\\\":\\\"nodejs10.x\\\",\\\"Tags\\\":[{\\\"Key\\\":
\\\"lambda:createdBy\\\",\\\"Value\\\":\\\"SAM\\\"}],\\\"Timeout\\\":22,\\\"TracingConfig\\\":{\\\"Mode\\\":
\\\"Active\\\"}}\",
    "PropertyDifferences": [
      {
        "PropertyPath": "/MemorySize",
        "ExpectedValue": "128",
        "ActualValue": "256",
        "DifferenceType": "NOT_EQUAL"
      },
      {
        "PropertyPath": "/Timeout",
        "ExpectedValue": "900",
        "ActualValue": "22",
        "DifferenceType": "NOT_EQUAL"
      }
    ],
    "StackResourceDriftStatus": "MODIFIED",
    "Timestamp": "2019-10-02T05:58:47.433Z"
  }
}
```

```
}
```

- For API details, see [DetectStackResourceDrift](#) in *AWS CLI Command Reference*.

detect-stack-set-drift

The following code example shows how to use `detect-stack-set-drift`.

AWS CLI

To detect drift on a stack set and all associated stack instances

The following `detect-stack-set-drift` example initiates drift detection operations on the specified stack set, including all the stack instances associated with that stack set, and returns an operation ID that can be used to track the status of the drift operation.

```
aws cloudformation detect-stack-set-drift \  
  --stack-set-name stack-set-drift-example
```

Output:

```
{  
  "OperationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE111111"  
}
```

For more information, see [Detecting Unmanaged Configuration Changes in Stack Sets](#) in the *AWS CloudFormation Users Guide*.

- For API details, see [DetectStackSetDrift](#) in *AWS CLI Command Reference*.

estimate-template-cost

The following code example shows how to use `estimate-template-cost`.

AWS CLI

To estimate template cost

The following `estimate-template-cost` example generates a cost estimate for a template named `template.yaml` in the current folder.

```
aws cloudformation estimate-template-cost \  
  --template-body file://template.yaml
```

Output:

```
{  
  "Url": "http://calculator.s3.amazonaws.com/calc5.html?  
key=cloudformation/7870825a-xmpl-4def-92e7-c4f8dd360cca"  
}
```

- For API details, see [EstimateTemplateCost](#) in *AWS CLI Command Reference*.

execute-change-set

The following code example shows how to use `execute-change-set`.

AWS CLI

To execute a change set

The following `execute-change-set` example executes a change set specified by change set name and stack name.

```
aws cloudformation execute-change-set \  
  --change-set-name my-change-set \  
  --stack-name my-stack
```

The following `execute-change-set` example executes a change set specified by the full ARN of the change set.

```
aws cloudformation execute-change-set \  
  --change-set-name arn:aws:cloudformation:us-west-2:123456789012:changeSet/my-  
change-set/bc9555ba-a949-xmpl-bfb8-f41d04ec5784
```

- For API details, see [ExecuteChangeSet](#) in *AWS CLI Command Reference*.

get-stack-policy

The following code example shows how to use `get-stack-policy`.

AWS CLI

To view a stack policy

The following `get-stack-policy` example displays the stack policy for the specified stack. To attach a policy to a stack, use the `set-stack-policy` command.

```
aws cloudformation get-stack-policy \  
  --stack-name my-stack
```

Output:

```
{  
  "StackPolicyBody": "{\  
    \"Statement\" : [  
      {  
        \"Effect\" :  
        \"Allow\",  
        \"Action\" : \"Update:*\",  
        \"Principal\": \"*\",  
        \"Resource\" : \"*\"  
      },  
      {  
        \"Effect\" : \"Deny\",  
        \"Action\" : \"Update:*\",  
        \"Principal\": \"*\",  
        \"Resource\" :  
        \"LogicalResourceId/bucket\"  
      }  
    ]  
  }"
```

- For API details, see [GetStackPolicy](#) in *AWS CLI Command Reference*.

get-template-summary

The following code example shows how to use `get-template-summary`.

AWS CLI

To display a template summary

The following command displays summary information about the resources and metadata for the specified template file.

```
aws cloudformation get-template-summary \  
  --template-body file://template.yaml
```

Output:

```
{  
  "Parameters": [],  
  "Description": "A VPC and subnets."
```

```

    "ResourceTypes": [
      "AWS::EC2::VPC",
      "AWS::EC2::Subnet",
      "AWS::EC2::Subnet",
      "AWS::EC2::RouteTable",
      "AWS::EC2::VPCEndpoint",
      "AWS::EC2::SubnetRouteTableAssociation",
      "AWS::EC2::SubnetRouteTableAssociation",
      "AWS::EC2::VPCEndpoint"
    ],
    "Version": "2010-09-09"
  }

```

- For API details, see [GetTemplateSummary](#) in *AWS CLI Command Reference*.

get-template

The following code example shows how to use `get-template`.

AWS CLI

To view the template body for an AWS CloudFormation stack

The following `get-template` command shows the template for the `myteststack` stack:

```
aws cloudformation get-template --stack-name myteststack
```

Output:

```

{
  "TemplateBody": {
    "AWSTemplateFormatVersion": "2010-09-09",
    "Outputs": {
      "BucketName": {
        "Description": "Name of S3 bucket to hold website content",
        "Value": {
          "Ref": "S3Bucket"
        }
      }
    }
  },
  "Description": "AWS CloudFormation Sample Template S3_Bucket: Sample template showing how to create a publicly accessible S3 bucket. **WARNING** This

```



```

    "CreationTime": "2019-10-02T05:38:54.297Z"
  }
]
}

```

- For API details, see [ListChangeSets](#) in *AWS CLI Command Reference*.

list-exports

The following code example shows how to use `list-exports`.

AWS CLI

To list exports

The following `list-exports` example displays a list of the exports from stacks in the current region.

```
aws cloudformation list-exports
```

Output:

```

{
  "Exports": [
    {
      "ExportingStackId": "arn:aws:cloudformation:us-
west-2:123456789012:stack/private-vpc/99764070-b56c-xmpl-bee8-062a88d1d800",
      "Name": "private-vpc-subnet-a",
      "Value": "subnet-07b410xmplddcfa03"
    },
    {
      "ExportingStackId": "arn:aws:cloudformation:us-
west-2:123456789012:stack/private-vpc/99764070-b56c-xmpl-bee8-062a88d1d800",
      "Name": "private-vpc-subnet-b",
      "Value": "subnet-075ed3xmplebd2fb1"
    },
    {
      "ExportingStackId": "arn:aws:cloudformation:us-
west-2:123456789012:stack/private-vpc/99764070-b56c-xmpl-bee8-062a88d1d800",
      "Name": "private-vpc-vpcid",
      "Value": "vpc-011d7xmpl100e9841"
    }
  ]
}

```

```
]
}
```

- For API details, see [ListExports](#) in *AWS CLI Command Reference*.

list-imports

The following code example shows how to use `list-imports`.

AWS CLI

To list imports

The following `list-imports` example lists the stacks that import the specified export. To get the list of available exports, use the `list-exports` command.

```
aws cloudformation list-imports \
  --export-name private-vpc-vpcid
```

Output:

```
{
  "Imports": [
    "my-database-stack"
  ]
}
```

- For API details, see [ListImports](#) in *AWS CLI Command Reference*.

list-stack-instances

The following code example shows how to use `list-stack-instances`.

AWS CLI

To list instances for a stack

The following `list-stack-instances` example lists the instances created from the specified stack set.

```
aws cloudformation list-stack-instances \
```

```
--stack-set-name enable-config
```

The example output includes details about a stack that failed to update due to an error:

```
{
  "Summaries": [
    {
      "StackSetId": "enable-config:296a3360-xmpl-40af-be78-9341e95bf743",
      "Region": "us-west-2",
      "Account": "123456789012",
      "StackId": "arn:aws:cloudformation:ap-northeast-1:123456789012:stack/StackSet-enable-config-35a6ac50-d9f8-4084-86e4-7da34d5de4c4/a1631cd0-e5fb-xmpl-b474-0aa20f14f06e",
      "Status": "CURRENT"
    },
    {
      "StackSetId": "enable-config:296a3360-xmpl-40af-be78-9341e95bf743",
      "Region": "us-west-2",
      "Account": "123456789012",
      "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/StackSet-enable-config-e6cac20f-xmpl-46e9-8314-53e0d4591532/eab53680-e5fa-xmpl-ba14-0a522351f81e",
      "Status": "OUTDATED",
      "StatusReason": "ResourceLogicalId:ConfigDeliveryChannel, ResourceType:AWS::Config::DeliveryChannel, ResourceStatusReason:Failed to put delivery channel 'StackSet-enable-config-e6cac20f-xmpl-46e9-8314-53e0d4591532-ConfigDeliveryChannel-10JWJ7XD59WR0' because the maximum number of delivery channels: 1 is reached. (Service: AmazonConfig; Status Code: 400; Error Code: MaxNumberOfDeliveryChannelsExceededException; Request ID: d14b34a0-ef7c-xmpl-acf8-8a864370ae56).",
    }
  ]
}
```

- For API details, see [ListStackInstances](#) in *AWS CLI Command Reference*.

list-stack-resources

The following code example shows how to use `list-stack-resources`.

AWS CLI

To list resources in a stack

The following command displays the list of resources in the specified stack.

```
aws cloudformation list-stack-resources \  
  --stack-name my-stack
```

Output:

```
{  
  "StackResourceSummaries": [  
    {  
      "LogicalResourceId": "bucket",  
      "PhysicalResourceId": "my-stack-bucket-1vc62xmplgguf",  
      "ResourceType": "AWS::S3::Bucket",  
      "LastUpdatedTimestamp": "2019-10-02T04:34:11.345Z",  
      "ResourceStatus": "CREATE_COMPLETE",  
      "DriftInformation": {  
        "StackResourceDriftStatus": "IN_SYNC"  
      }  
    },  
    {  
      "LogicalResourceId": "function",  
      "PhysicalResourceId": "my-function-SEZV4XMPL4S5",  
      "ResourceType": "AWS::Lambda::Function",  
      "LastUpdatedTimestamp": "2019-10-02T05:34:27.989Z",  
      "ResourceStatus": "UPDATE_COMPLETE",  
      "DriftInformation": {  
        "StackResourceDriftStatus": "IN_SYNC"  
      }  
    },  
    {  
      "LogicalResourceId": "functionRole",  
      "PhysicalResourceId": "my-functionRole-HIZXMPLEOM9E",  
      "ResourceType": "AWS::IAM::Role",  
      "LastUpdatedTimestamp": "2019-10-02T04:34:06.350Z",  
      "ResourceStatus": "CREATE_COMPLETE",  
      "DriftInformation": {  
        "StackResourceDriftStatus": "IN_SYNC"  
      }  
    }  
  ]  
}
```

- For API details, see [ListStackResources](#) in *AWS CLI Command Reference*.

list-stack-set-operation-results

The following code example shows how to use `list-stack-set-operation-results`.

AWS CLI

To list stack set operation results

The following command displays the results of an update operation on instances in the specified stack set.

```
aws cloudformation list-stack-set-operation-results \  
  --stack-set-name enable-config \  
  --operation-id 35d45ebc-ed88-xmpl-ab59-0197a1fc83a0
```

Output:

```
{  
  "Summaries": [  
    {  
      "Account": "223456789012",  
      "Region": "us-west-2",  
      "Status": "SUCCEEDED",  
      "AccountGateResult": {  
        "Status": "SKIPPED",  
        "StatusReason": "Function not found: arn:aws:lambda:eu-west-1:223456789012:function:AWSCloudFormationStackSetAccountGate"  
      }  
    },  
    {  
      "Account": "223456789012",  
      "Region": "ap-south-1",  
      "Status": "CANCELLED",  
      "StatusReason": "Cancelled since failure tolerance has exceeded"  
    }  
  ]  
}
```

Note: The SKIPPED status for `AccountGateResult` is expected for successful operations unless you create an account gate function.

- For API details, see [ListStackSetOperationResults](#) in *AWS CLI Command Reference*.

list-stack-set-operations

The following code example shows how to use `list-stack-set-operations`.

AWS CLI

To list stack set operations

The following `list-stack-set-operations` example displays the list of the most recent operations on the specified stack set.

```
aws cloudformation list-stack-set-operations \  
  --stack-set-name my-stack-set
```

Output:

```
{  
  "Summaries": [  
    {  
      "OperationId": "35d45ebc-ed88-xmpl-ab59-0197a1fc83a0",  
      "Action": "UPDATE",  
      "Status": "SUCCEEDED",  
      "CreationTimestamp": "2019-10-03T16:28:44.377Z",  
      "EndTimestamp": "2019-10-03T16:42:08.607Z"  
    },  
    {  
      "OperationId": "891aa98f-7118-xmpl-00b2-00954d1dd0d6",  
      "Action": "UPDATE",  
      "Status": "FAILED",  
      "CreationTimestamp": "2019-10-03T15:43:53.916Z",  
      "EndTimestamp": "2019-10-03T15:45:58.925Z"  
    }  
  ]  
}
```

- For API details, see [ListStackSetOperations](#) in *AWS CLI Command Reference*.

list-stack-sets

The following code example shows how to use `list-stack-sets`.

AWS CLI

To list stack sets

The following `list-stack-sets` example displays the list of stack sets in the current region and account.

```
aws cloudformation list-stack-sets
```

Output:

```
{
  "Summaries": [
    {
      "StackSetName": "enable-config",
      "StackSetId": "enable-config:296a3360-xmpl-40af-be78-9341e95bf743",
      "Description": "Enable AWS Config",
      "Status": "ACTIVE"
    }
  ]
}
```

- For API details, see [ListStackSets](#) in *AWS CLI Command Reference*.

list-stacks

The following code example shows how to use `list-stacks`.

AWS CLI

To list AWS CloudFormation stacks

The following `list-stacks` command shows a summary of all stacks that have a status of `CREATE_COMPLETE`:

```
aws cloudformation list-stacks --stack-status-filter CREATE_COMPLETE
```

Output:

```
[
```



```

{
  "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896",
  "TemplateDescription": "AWS CloudFormation Sample Template S3_Bucket: Sample
template showing how to create a publicly accessible S3 bucket. **WARNING** This
template creates an S3 bucket. You will be billed for the AWS resources used if you
create a stack from this template.",
  "StackStatusReason": null,
  "CreationTime": "2013-08-26T03:27:10.190Z",
  "StackName": "myteststack",
  "StackStatus": "CREATE_COMPLETE"
}
]

```

- For API details, see [ListStacks](#) in *AWS CLI Command Reference*.

list-type-registrations

The following code example shows how to use `list-type-registrations`.

AWS CLI

To list the completed registrations of a type

The following `list-type-registrations` example displays a list of the completed type registrations for the specified type.

```

aws cloudformation list-type-registrations \
  --type RESOURCE \
  --type-name My::Logs::LogGroup \
  --registration-status-filter COMPLETE

```

Output:

```

{
  "RegistrationTokenList": [
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333"
  ]
}

```

For more information, see [Using the CloudFormation Registry](#) in the *AWS CloudFormation Users Guide*.

- For API details, see [ListTypeRegistrations](#) in *AWS CLI Command Reference*.

list-type-versions

The following code example shows how to use `list-type-versions`.

AWS CLI

To list an extension's version

The following `list-type-versions` example returns summary information about the versions of an extension.

```
aws cloudformation list-type-versions \  
  --endpoint https://example.com \  
  --region us-west-2 \  
  --type RESOURCE \  
  --type-name My::Resource::Example \  
  --publisher-id 123456789012
```

This command produces no output.

For more information, see [Using the AWS CloudFormation registry](#) in the *AWS CloudFormation User Guide*.

- For API details, see [ListTypeVersions](#) in *AWS CLI Command Reference*.

list-types

The following code example shows how to use `list-types`.

AWS CLI

To list the private resource types in an account

The following `list-types` example displays a list of the private resource types currently registered in the current AWS account.

```
aws cloudformation list-types
```

Output:

```
{
  "TypeSummaries": [
    {
      "Description": "WordPress blog resource for internal use",
      "LastUpdated": "2019-12-04T18:28:15.059Z",
      "TypeName": "My::WordPress::BlogExample",
      "TypeArn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/My-WordPress-BlogExample",
      "DefaultVersionId": "00000005",
      "Type": "RESOURCE"
    },
    {
      "Description": "Customized resource derived from AWS::Logs::LogGroup",
      "LastUpdated": "2019-12-04T18:28:15.059Z",
      "TypeName": "My::Logs::LogGroup",
      "TypeArn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/My-Logs-LogGroup",
      "DefaultVersionId": "00000003",
      "Type": "RESOURCE"
    }
  ]
}
```

For more information, see [Using the CloudFormation Registry](#) in the *AWS CloudFormation Users Guide*.

- For API details, see [ListTypes](#) in *AWS CLI Command Reference*.

package

The following code example shows how to use package.

AWS CLI

Following command exports a template named `template.json` by uploading local artifacts to S3 bucket `bucket-name` and writes the exported template to `packaged-template.json`:

```
aws cloudformation package --template-file /path_to_template/template.json --s3-bucket bucket-name --output-template-file packaged-template.json --use-json
```

- For API details, see [Package](#) in *AWS CLI Command Reference*.

publish-type

The following code example shows how to use `publish-type`.

AWS CLI

To publish an extension

The following `publish-type` example publishes the specified extension to the CloudFormation registry as a public extension in this Region.

```
aws cloudformation publish-type \  
  --region us-west-2 \  
  --type RESOURCE \  
  --type-name Example::Test::1234567890abcdef0
```

Output:

```
{  
  "PublicTypeArn": "arn:aws:cloudformation:us-west-2::type/  
resource/000q6TfUovXsEMmgKowxDZLLwqr2QUshd2e75c8c/Example-  
Test-1234567890abcdef0/1.0.0"  
}
```

For more information, see [Using the AWS CloudFormation registry](#) in the *AWS CloudFormation User Guide*.

- For API details, see [PublishType](#) in *AWS CLI Command Reference*.

register-publisher

The following code example shows how to use `register-publisher`.

AWS CLI

To register a publisher

The following `register-publisher` example registers a publisher and accepts the terms and condition parameter.

```
aws cloudformation register-publisher \  
  --region us-west-2 \  
  --accept-terms-and-conditions
```

Output:

```
{  
  "PublisherId": "000q6TfUovXsEMmgKowxDZLlwqr2QUshd2e75c8c"  
}
```

For more information, see [Using the AWS CloudFormation registry](#) in the *AWS CloudFormation User Guide*.

- For API details, see [RegisterPublisher](#) in *AWS CLI Command Reference*.

register-type

The following code example shows how to use `register-type`.

AWS CLI

To register a resource type

The following `register-type` example registers the specified resource type as a private resource type in the user's account.

```
aws cloudformation register-type \  
  --type-name My::Organization::ResourceName \  
  --schema-handler-package s3://bucket_name/my-organization-resource_name.zip \  
  --type RESOURCE
```

Output:

```
{  
  "RegistrationToken": "f5525280-104e-4d35-bef5-8f1f1example"  
}
```

For more information, see [Registering Resource Providers](#) in the *CloudFormation Command Line Interface User Guide for Type Development*.

- For API details, see [RegisterType](#) in *AWS CLI Command Reference*.

set-stack-policy

The following code example shows how to use `set-stack-policy`.

AWS CLI

To apply a stack policy

The following `set-stack-policy` example disables updates for the specified resource in the specified stack. `stack-policy.json` is a JSON document that defines the operations allowed on resources in the stack.

```
aws cloudformation set-stack-policy \  
  --stack-name my-stack \  
  --stack-policy-body file://stack-policy.json
```

Output:

```
{  
  "Statement" : [  
    {  
      "Effect" : "Allow",  
      "Action" : "Update:*",  
      "Principal": "*",  
      "Resource" : "*"   
    },  
    {  
      "Effect" : "Deny",  
      "Action" : "Update:*",  
      "Principal": "*",  
      "Resource" : "LogicalResourceId/bucket"   
    }  
  ]  
}
```

- For API details, see [SetStackPolicy](#) in *AWS CLI Command Reference*.

set-type-configuration

The following code example shows how to use `set-type-configuration`.

AWS CLI

To configure data

The following `set-type-configuration` example specifies the configuration data for a registered CloudFormation extension, in the given account and Region.

```
aws cloudformation set-type-configuration \  
  --region us-west-2 \  
  --type RESOURCE \  
  --type-name Example::Test::Type \  
  --configuration-alias default \  
  --configuration "{\"CredentialKey\": \"testUserCredential\"}"
```

Output:

```
{  
  "ConfigurationArn": "arn:aws:cloudformation:us-west-2:123456789012:type-  
configuration/resource/Example-Test-Type/default"  
}
```

For more information, see [Using the AWS CloudFormation registry](#) in the *AWS CloudFormation User Guide*.

- For API details, see [SetTypeConfiguration](#) in *AWS CLI Command Reference*.

set-type-default-version

The following code example shows how to use `set-type-default-version`.

AWS CLI

To set a type's default version

The following `set-type-default-version` example sets the specified type version to be used as the default for this type.

```
aws cloudformation set-type-default-version \  
  --type RESOURCE \  
  --type-name My::Logs::LogGroup \  
  --version-id 00000003
```

This command produces no output.

For more information, see [Using the CloudFormation Registry](#) in the *AWS CloudFormation Users Guide*.

- For API details, see [SetTypeDefaultVersion](#) in *AWS CLI Command Reference*.

signal-resource

The following code example shows how to use `signal-resource`.

AWS CLI

To signal a resource

The following `signal-resource` example signals success to fulfill the wait condition named `MyWaitCondition` in the stack named `my-stack`.

```
aws cloudformation signal-resource \  
  --stack-name my-stack \  
  --logical-resource-id MyWaitCondition \  
  --unique-id 1234 \  
  --status SUCCESS
```

This command produces no output.

- For API details, see [SignalResource](#) in *AWS CLI Command Reference*.

stop-stack-set-operation

The following code example shows how to use `stop-stack-set-operation`.

AWS CLI

To stop a stack set operation

The following `stop-stack-set-operation` example stops an in-progress update operation on the specified stack set.

```
aws cloudformation stop-stack-set-operation \  
  --stack-set-name my-stack-set \  
  --operation-id 1261cd27-490b-xmpl-ab42-793a896c69e6
```


This command produces no output.

- For API details, see [StopStackSetOperation](#) in *AWS CLI Command Reference*.

test-type

The following code example shows how to use `test-type`.

AWS CLI

To test an extension

The following `test-type` example tests a registered extension to make sure it meets all necessary requirements for being published in the CloudFormation registry.

```
aws cloudformation test-type \  
    --arn arn:aws:cloudformation:us-west-2:123456789012:type/resource/Sample-Test-  
Resource123/00000001
```

Output:

```
{  
  "TypeVersionArn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/  
Sample-Test-Resource123/00000001"  
}
```

For more information, see [Using the AWS CloudFormation registry](#) in the *AWS CloudFormation User Guide*.

- For API details, see [TestType](#) in *AWS CLI Command Reference*.

update-stack-instances

The following code example shows how to use `update-stack-instances`.

AWS CLI

To update stack instances

The following `update-stack-instances` example retries an update on stack instances in two accounts in two regions with the most recent settings. The specified fault tolerance setting

ensures that the update is attempted in all accounts and regions, even if some stacks cannot be updated.

```
aws cloudformation update-stack-instances \  
  --stack-set-name my-stack-set \  
  --accounts 123456789012 567890123456 \  
  --regions us-east-1 us-west-2 \  
  --operation-preferences FailureToleranceCount=3
```

Output:

```
{  
  "OperationId": "103ebdf2-21ea-xmpl-8892-de5e30733132"  
}
```

- For API details, see [UpdateStackInstances](#) in *AWS CLI Command Reference*.

update-stack-set

The following code example shows how to use `update-stack-set`.

AWS CLI

To update a stack set

The following `update-stack-set` example adds a tag with the key name `Owner` and a value of `IT` to the stack instances in the specified stack set.

```
aws cloudformation update-stack-set \  
  --stack-set-name my-stack-set \  
  --use-previous-template \  
  --tags Key=Owner,Value=IT
```

Output:

```
{  
  "OperationId": "e2b60321-6cab-xmpl-bde7-530c6f47950e"  
}
```

- For API details, see [UpdateStackSet](#) in *AWS CLI Command Reference*.

update-stack

The following code example shows how to use `update-stack`.

AWS CLI

To update AWS CloudFormation stacks

The following `update-stack` command updates the template and input parameters for the `mystack` stack:

```
aws cloudformation update-stack --stack-name mystack --
template-url https://s3.amazonaws.com/sample/updated.template --
parameters ParameterKey=KeyPairName,ParameterValue=SampleKeyPair
ParameterKey=SubnetIDs,ParameterValue=SampleSubnetID1\\,SampleSubnetID2
```

The following `update-stack` command updates just the `SubnetIDs` parameter value for the `mystack` stack. If you don't specify a parameter value, the default value that is specified in the template is used:

```
aws cloudformation update-stack --stack-name mystack --
template-url https://s3.amazonaws.com/sample/updated.template
--parameters ParameterKey=KeyPairName,UsePreviousValue=true
ParameterKey=SubnetIDs,ParameterValue=SampleSubnetID1\\,UpdatedSampleSubnetID2
```

The following `update-stack` command adds two stack notification topics to the `mystack` stack:

```
aws cloudformation update-stack --stack-name mystack --use-previous-template --
notification-arns "arn:aws:sns:use-east-1:123456789012:mytopic1" "arn:aws:sns:us-
east-1:123456789012:mytopic2"
```

For more information, see [AWS CloudFormation stack updates](#) in the *AWS CloudFormation User Guide*.

- For API details, see [UpdateStack](#) in *AWS CLI Command Reference*.

update-termination-protection

The following code example shows how to use `update-termination-protection`.

AWS CLI

To enable termination protection

The following `update-termination-protection` example enables termination protection on the specified stack.

```
aws cloudformation update-termination-protection \  
  --stack-name my-stack \  
  --enable-termination-protection
```

Output:

```
{  
  "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-stack/  
d0a825a0-e4cd-xmpl-b9fb-061c69e99204"  
}
```

- For API details, see [UpdateTerminationProtection](#) in *AWS CLI Command Reference*.

validate-template

The following code example shows how to use `validate-template`.

AWS CLI

To validate an AWS CloudFormation template

The following `validate-template` command validates the `sampletemplate.json` template:

```
aws cloudformation validate-template --template-body file://sampletemplate.json
```

Output:

```
{  
  "Description": "AWS CloudFormation Sample Template S3_Bucket: Sample template  
showing how to create a publicly accessible S3 bucket. **WARNING** This template  
creates an S3 bucket. You will be billed for the AWS resources used if you create a  
stack from this template.",  
  "Parameters": [],
```

```
"Capabilities": []  
}
```

For more information, see *Working with AWS CloudFormation Templates* in the *AWS CloudFormation User Guide*.

- For API details, see [ValidateTemplate](#) in *AWS CLI Command Reference*.

CloudFront examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with CloudFront.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-cloud-front-origin-access-identity

The following code example shows how to use `create-cloud-front-origin-access-identity`.

AWS CLI

To create a CloudFront origin access identity

The following example creates a CloudFront origin access identity (OAI) by providing the OAI configuration as a command line argument:

```
aws cloudfront create-cloud-front-origin-access-identity \
```

```
--cloud-front-origin-access-identity-config \  
    CallerReference="cli-example",Comment="Example OAI"
```

You can accomplish the same thing by providing the OAI configuration in a JSON file, as shown in the following example:

```
aws cloudfront create-cloud-front-origin-access-identity \  
    --cloud-front-origin-access-identity-config file://OAI-config.json
```

The file `OAI-config.json` is a JSON document in the current directory that contains the following:

```
{  
  "CallerReference": "cli-example",  
  "Comment": "Example OAI"  
}
```

Whether you provide the OAI configuration with a command line argument or a JSON file, the output is the same:

```
{  
  "Location": "https://cloudfront.amazonaws.com/2019-03-26/origin-access-identity/  
cloudfront/E74FTE3AEXAMPLE",  
  "ETag": "E2QWRUHEXAMPLE",  
  "CloudFrontOriginAccessIdentity": {  
    "Id": "E74FTE3AEXAMPLE",  
    "S3CanonicalUserId":  
"cd13868f797c227fbea2830611a26fe0a21ba1b826ab4bed9b7771c9aEXAMPLE",  
    "CloudFrontOriginAccessIdentityConfig": {  
      "CallerReference": "cli-example",  
      "Comment": "Example OAI"  
    }  
  }  
}
```

- For API details, see [CreateCloudFrontOriginAccessIdentity](#) in *AWS CLI Command Reference*.

create-distribution-with-tags

The following code example shows how to use `create-distribution-with-tags`.

AWS CLI

To create a CloudFront distribution with tags

The following example creates a distribution with two tags by providing the distribution configuration and tags in a JSON file named `dist-config-with-tags.json`:

```
aws cloudfront create-distribution-with-tags \  
  --distribution-config-with-tags file://dist-config-with-tags.json
```

The file `dist-config-with-tags.json` is a JSON document in the current folder that contains the following. Note the `Tags` object at the top of the file, which contains two tags:

Name = ExampleDistributionProject = ExampleProject

```
{  
  "Tags": {  
    "Items": [  
      {  
        "Key": "Name",  
        "Value": "ExampleDistribution"  
      },  
      {  
        "Key": "Project",  
        "Value": "ExampleProject"  
      }  
    ]  
  },  
  "DistributionConfig": {  
    "CallerReference": "cli-example",  
    "Aliases": {  
      "Quantity": 0  
    },  
    "DefaultRootObject": "index.html",  
    "Origins": {  
      "Quantity": 1,  
      "Items": [  
        {  
          "Id": "awsexamplebucket.s3.amazonaws.com-cli-example",  
          "DomainName": "awsexamplebucket.s3.amazonaws.com",  
          "OriginPath": "",  
          "CustomHeaders": {  
            "Quantity": 0  
          }  
        }  
      ]  
    }  
  }  
}
```

```
        },
        "S3OriginConfig": {
            "OriginAccessIdentity": ""
        }
    ]
},
"OriginGroups": {
    "Quantity": 0
},
"DefaultCacheBehavior": {
    "TargetOriginId": "awsexamplebucket.s3.amazonaws.com-cli-example",
    "ForwardedValues": {
        "QueryString": false,
        "Cookies": {
            "Forward": "none"
        },
        "Headers": {
            "Quantity": 0
        },
        "QueryStringCacheKeys": {
            "Quantity": 0
        }
    },
    "TrustedSigners": {
        "Enabled": false,
        "Quantity": 0
    },
    "ViewerProtocolPolicy": "allow-all",
    "MinTTL": 0,
    "AllowedMethods": {
        "Quantity": 2,
        "Items": [
            "HEAD",
            "GET"
        ],
        "CachedMethods": {
            "Quantity": 2,
            "Items": [
                "HEAD",
                "GET"
            ]
        }
    }
},
```



```
    "SmoothStreaming": false,
    "DefaultTTL": 86400,
    "MaxTTL": 31536000,
    "Compress": false,
    "LambdaFunctionAssociations": {
      "Quantity": 0
    },
    "FieldLevelEncryptionId": ""
  },
  "CacheBehaviors": {
    "Quantity": 0
  },
  "CustomErrorResponses": {
    "Quantity": 0
  },
  "Comment": "",
  "Logging": {
    "Enabled": false,
    "IncludeCookies": false,
    "Bucket": "",
    "Prefix": ""
  },
  "PriceClass": "PriceClass_All",
  "Enabled": true,
  "ViewerCertificate": {
    "CloudFrontDefaultCertificate": true,
    "MinimumProtocolVersion": "TLSv1",
    "CertificateSource": "cloudfront"
  },
  "Restrictions": {
    "GeoRestriction": {
      "RestrictionType": "none",
      "Quantity": 0
    }
  },
  "WebACLId": "",
  "HttpVersion": "http2",
  "IsIPV6Enabled": true
}
```

Output:

```
{
  "Location": "https://cloudfront.amazonaws.com/2019-03-26/distribution/
EDFDVBD6EXAMPLE",
  "ETag": "E2QWRUHEXAMPLE",
  "Distribution": {
    "Id": "EDFDVBD6EXAMPLE",
    "ARN": "arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE",
    "Status": "InProgress",
    "LastModifiedTime": "2019-12-04T23:35:41.433Z",
    "InProgressInvalidationBatches": 0,
    "DomainName": "d1111111abcdef8.cloudfront.net",
    "ActiveTrustedSigners": {
      "Enabled": false,
      "Quantity": 0
    },
    "DistributionConfig": {
      "CallerReference": "cli-example",
      "Aliases": {
        "Quantity": 0
      },
      "DefaultRootObject": "index.html",
      "Origins": {
        "Quantity": 1,
        "Items": [
          {
            "Id": "awsexamplebucket.s3.amazonaws.com-cli-example",
            "DomainName": "awsexamplebucket.s3.amazonaws.com",
            "OriginPath": "",
            "CustomHeaders": {
              "Quantity": 0
            },
            "S3OriginConfig": {
              "OriginAccessIdentity": ""
            }
          }
        ]
      },
      "OriginGroups": {
        "Quantity": 0
      },
      "DefaultCacheBehavior": {
        "TargetOriginId": "awsexamplebucket.s3.amazonaws.com-cli-example",
        "ForwardedValues": {
```

```
    "QueryString": false,
    "Cookies": {
      "Forward": "none"
    },
    "Headers": {
      "Quantity": 0
    },
    "QueryStringCacheKeys": {
      "Quantity": 0
    }
  },
  "TrustedSigners": {
    "Enabled": false,
    "Quantity": 0
  },
  "ViewerProtocolPolicy": "allow-all",
  "MinTTL": 0,
  "AllowedMethods": {
    "Quantity": 2,
    "Items": [
      "HEAD",
      "GET"
    ],
  },
  "CachedMethods": {
    "Quantity": 2,
    "Items": [
      "HEAD",
      "GET"
    ]
  }
},
"SmoothStreaming": false,
"DefaultTTL": 86400,
"MaxTTL": 31536000,
"Compress": false,
"LambdaFunctionAssociations": {
  "Quantity": 0
},
"FieldLevelEncryptionId": ""
},
"CacheBehaviors": {
  "Quantity": 0
},
"CustomErrorResponses": {
```

```

        "Quantity": 0
    },
    "Comment": "",
    "Logging": {
        "Enabled": false,
        "IncludeCookies": false,
        "Bucket": "",
        "Prefix": ""
    },
    "PriceClass": "PriceClass_All",
    "Enabled": true,
    "ViewerCertificate": {
        "CloudFrontDefaultCertificate": true,
        "MinimumProtocolVersion": "TLSv1",
        "CertificateSource": "cloudfront"
    },
    "Restrictions": {
        "GeoRestriction": {
            "RestrictionType": "none",
            "Quantity": 0
        }
    },
    "WebACLId": "",
    "HttpVersion": "http2",
    "IsIPV6Enabled": true
}
}
}

```

- For API details, see [CreateDistributionWithTags](#) in *AWS CLI Command Reference*.

create-distribution

The following code example shows how to use `create-distribution`.

AWS CLI

To create a CloudFront distribution

The following example creates a distribution for an S3 bucket named `awsexamplebucket`, and also specifies `index.html` as the default root object, using command line arguments:

```
aws cloudfront create-distribution \
```

```
--origin-domain-name awsexamplebucket.s3.amazonaws.com \  
--default-root-object index.html
```

Instead of using command line arguments, you can provide the distribution configuration in a JSON file, as shown in the following example:

```
aws cloudfront create-distribution \  
--distribution-config file://dist-config.json
```

The file `dist-config.json` is a JSON document in the current folder that contains the following:

```
{  
  "CallerReference": "cli-example",  
  "Aliases": {  
    "Quantity": 0  
  },  
  "DefaultRootObject": "index.html",  
  "Origins": {  
    "Quantity": 1,  
    "Items": [  
      {  
        "Id": "awsexamplebucket.s3.amazonaws.com-cli-example",  
        "DomainName": "awsexamplebucket.s3.amazonaws.com",  
        "OriginPath": "",  
        "CustomHeaders": {  
          "Quantity": 0  
        },  
        "S3OriginConfig": {  
          "OriginAccessIdentity": ""  
        }  
      }  
    ]  
  },  
  "OriginGroups": {  
    "Quantity": 0  
  },  
  "DefaultCacheBehavior": {  
    "TargetOriginId": "awsexamplebucket.s3.amazonaws.com-cli-example",  
    "ForwardedValues": {  
      "QueryString": false,  
      "Cookies": {
```

```
        "Forward": "none"
    },
    "Headers": {
        "Quantity": 0
    },
    "QueryStringCacheKeys": {
        "Quantity": 0
    }
},
"TrustedSigners": {
    "Enabled": false,
    "Quantity": 0
},
"ViewerProtocolPolicy": "allow-all",
"MinTTL": 0,
"AllowedMethods": {
    "Quantity": 2,
    "Items": [
        "HEAD",
        "GET"
    ],
    "CachedMethods": {
        "Quantity": 2,
        "Items": [
            "HEAD",
            "GET"
        ]
    }
},
"SmoothStreaming": false,
"DefaultTTL": 86400,
"MaxTTL": 31536000,
"Compress": false,
"LambdaFunctionAssociations": {
    "Quantity": 0
},
"FieldLevelEncryptionId": ""
},
"CacheBehaviors": {
    "Quantity": 0
},
"CustomErrorResponses": {
    "Quantity": 0
},
}
```

```

"Comment": "",
"Logging": {
  "Enabled": false,
  "IncludeCookies": false,
  "Bucket": "",
  "Prefix": ""
},
"PriceClass": "PriceClass_All",
"Enabled": true,
"ViewerCertificate": {
  "CloudFrontDefaultCertificate": true,
  "MinimumProtocolVersion": "TLSv1",
  "CertificateSource": "cloudfront"
},
"Restrictions": {
  "GeoRestriction": {
    "RestrictionType": "none",
    "Quantity": 0
  }
},
"WebACLId": "",
"HttpVersion": "http2",
"IsIPV6Enabled": true
}

```

Whether you provide the distribution information with a command line argument or a JSON file, the output is the same:

```

{
  "Location": "https://cloudfront.amazonaws.com/2019-03-26/distribution/EMLARXS9EXAMPLE",
  "ETag": "E9LHASXEXAMPLE",
  "Distribution": {
    "Id": "EMLARXS9EXAMPLE",
    "ARN": "arn:aws:cloudfront::123456789012:distribution/EMLARXS9EXAMPLE",
    "Status": "InProgress",
    "LastModifiedTime": "2019-11-22T00:55:15.705Z",
    "InProgressInvalidationBatches": 0,
    "DomainName": "d1111111abcdef8.cloudfront.net",
    "ActiveTrustedSigners": {
      "Enabled": false,
      "Quantity": 0
    }
  },
}

```

```
"DistributionConfig": {
  "CallerReference": "cli-example",
  "Aliases": {
    "Quantity": 0
  },
  "DefaultRootObject": "index.html",
  "Origins": {
    "Quantity": 1,
    "Items": [
      {
        "Id": "awsexamplebucket.s3.amazonaws.com-cli-example",
        "DomainName": "awsexamplebucket.s3.amazonaws.com",
        "OriginPath": "",
        "CustomHeaders": {
          "Quantity": 0
        },
        "S3OriginConfig": {
          "OriginAccessIdentity": ""
        }
      }
    ]
  },
  "OriginGroups": {
    "Quantity": 0
  },
  "DefaultCacheBehavior": {
    "TargetOriginId": "awsexamplebucket.s3.amazonaws.com-cli-example",
    "ForwardedValues": {
      "QueryString": false,
      "Cookies": {
        "Forward": "none"
      },
      "Headers": {
        "Quantity": 0
      },
      "QueryStringCacheKeys": {
        "Quantity": 0
      }
    },
    "TrustedSigners": {
      "Enabled": false,
      "Quantity": 0
    },
    "ViewerProtocolPolicy": "allow-all",
```



```
    "MinTTL": 0,
    "AllowedMethods": {
      "Quantity": 2,
      "Items": [
        "HEAD",
        "GET"
      ],
      "CachedMethods": {
        "Quantity": 2,
        "Items": [
          "HEAD",
          "GET"
        ]
      }
    },
    "SmoothStreaming": false,
    "DefaultTTL": 86400,
    "MaxTTL": 31536000,
    "Compress": false,
    "LambdaFunctionAssociations": {
      "Quantity": 0
    },
    "FieldLevelEncryptionId": ""
  },
  "CacheBehaviors": {
    "Quantity": 0
  },
  "CustomErrorResponses": {
    "Quantity": 0
  },
  "Comment": "",
  "Logging": {
    "Enabled": false,
    "IncludeCookies": false,
    "Bucket": "",
    "Prefix": ""
  },
  "PriceClass": "PriceClass_All",
  "Enabled": true,
  "ViewerCertificate": {
    "CloudFrontDefaultCertificate": true,
    "MinimumProtocolVersion": "TLSv1",
    "CertificateSource": "cloudfront"
  },
}
```

```

    "Restrictions": {
      "GeoRestriction": {
        "RestrictionType": "none",
        "Quantity": 0
      }
    },
    "WebACLId": "",
    "HttpVersion": "http2",
    "IsIPV6Enabled": true
  }
}

```

- For API details, see [CreateDistribution](#) in *AWS CLI Command Reference*.

create-field-level-encryption-config

The following code example shows how to use `create-field-level-encryption-config`.

AWS CLI

To create a CloudFront field-level encryption configuration

The following example creates a field-level encryption configuration by providing the configuration parameters in a JSON file named `fle-config.json`. Before you can create a field-level encryption configuration, you must have a field-level encryption profile. To create a profile, see the `create-field-level-encryption-profile` command.

For more information about CloudFront field-level encryption, see [Using Field-Level Encryption to Help Protect Sensitive Data](#) in the *Amazon CloudFront Developer Guide*.

```

aws cloudfront create-field-level-encryption-config \
  --field-level-encryption-config file://fle-config.json

```

The file `fle-config.json` is a JSON document in the current folder that contains the following:

```

{
  "CallerReference": "cli-example",
  "Comment": "Example FLE configuration",
  "QueryArgProfileConfig": {

```

```

    "ForwardWhenQueryArgProfileIsUnknown": true,
    "QueryArgProfiles": {
      "Quantity": 0
    }
  },
  "ContentTypeProfileConfig": {
    "ForwardWhenContentTypeIsUnknown": true,
    "ContentTypeProfiles": {
      "Quantity": 1,
      "Items": [
        {
          "Format": "URLEncoded",
          "ProfileId": "P280MFCLSYOCVU",
          "ContentType": "application/x-www-form-urlencoded"
        }
      ]
    }
  }
}

```

Output:

```

{
  "Location": "https://cloudfront.amazonaws.com/2019-03-26/field-level-encryption/C3KM2WVD605UAY",
  "ETag": "E2P4Z4VU7TY5SG",
  "FieldLevelEncryption": {
    "Id": "C3KM2WVD605UAY",
    "LastModifiedTime": "2019-12-10T21:30:18.974Z",
    "FieldLevelEncryptionConfig": {
      "CallerReference": "cli-example",
      "Comment": "Example FLE configuration",
      "QueryArgProfileConfig": {
        "ForwardWhenQueryArgProfileIsUnknown": true,
        "QueryArgProfiles": {
          "Quantity": 0,
          "Items": []
        }
      }
    }
  },
  "ContentTypeProfileConfig": {
    "ForwardWhenContentTypeIsUnknown": true,
    "ContentTypeProfiles": {
      "Quantity": 1,

```



```

"EncryptionEntities": {
  "Quantity": 1,
  "Items": [
    {
      "PublicKeyId": "K2K8NC4HVFE3M0",
      "ProviderId": "ExampleFLEProvider",
      "FieldPatterns": {
        "Quantity": 1,
        "Items": [
          "ExampleSensitiveField"
        ]
      }
    }
  ]
}

```

Output:

```

{
  "Location": "https://cloudfront.amazonaws.com/2019-03-26/field-level-encryption-profile/PPK0UOSIF5WSV",
  "ETag": "E2QWRUHEXAMPLE",
  "FieldLevelEncryptionProfile": {
    "Id": "PPK0UOSIF5WSV",
    "LastModifiedTime": "2019-12-10T01:03:16.537Z",
    "FieldLevelEncryptionProfileConfig": {
      "Name": "ExampleFLEProfile",
      "CallerReference": "cli-example",
      "Comment": "FLE profile for AWS CLI example",
      "EncryptionEntities": {
        "Quantity": 1,
        "Items": [
          {
            "PublicKeyId": "K2K8NC4HVFE3M0",
            "ProviderId": "ExampleFLEProvider",
            "FieldPatterns": {
              "Quantity": 1,
              "Items": [
                "ExampleSensitiveField"
              ]
            }
          }
        ]
      }
    }
  }
}

```



```
}  
}
```

In the previous example, the AWS CLI automatically generated a random `CallerReference`. To specify your own `CallerReference`, or to avoid passing the invalidation parameters as command line arguments, you can use a JSON file. The following example creates an invalidation for two files, by providing the invalidation parameters in a JSON file named `inv-batch.json`:

```
aws cloudfront create-invalidation \  
  --distribution-id EDFDVBD6EXAMPLE \  
  --invalidation-batch file://inv-batch.json
```

Contents of `inv-batch.json`:

```
{  
  "Paths": {  
    "Quantity": 2,  
    "Items": [  
      "/example-path/example-file.jpg",  
      "/example-path/example-file2.png"  
    ]  
  },  
  "CallerReference": "cli-example"  
}
```

Output:

```
{  
  "Location": "https://cloudfront.amazonaws.com/2019-03-26/distribution/  
EDFDVBD6EXAMPLE/invalidation/I2J0I21PCUY0IK",  
  "Invalidation": {  
    "Id": "I2J0I21PCUY0IK",  
    "Status": "InProgress",  
    "CreateTime": "2019-12-05T18:40:49.413Z",  
    "InvalidationBatch": {  
      "Paths": {  
        "Quantity": 2,  
        "Items": [  
          "/example-path/example-file.jpg",  
          "/example-path/example-file2.png"  
        ]  
      }  
    }  
  }  
}
```

```

    ]
    },
    "CallerReference": "cli-example"
  }
}

```

- For API details, see [CreateInvalidation](#) in *AWS CLI Command Reference*.

create-public-key

The following code example shows how to use create-public-key.

AWS CLI

To create a CloudFront public key

The following example creates a CloudFront public key by providing the parameters in a JSON file named `pub-key-config.json`. Before you can use this command, you must have a PEM-encoded public key. For more information, see [Create an RSA Key Pair](#) in the *Amazon CloudFront Developer Guide*.

```
aws cloudfront create-public-key \
  --public-key-config file://pub-key-config.json
```

The file `pub-key-config.json` is a JSON document in the current folder that contains the following. Note that the public key is encoded in PEM format.

```
{
  "CallerReference": "cli-example",
  "Name": "ExampleKey",
  "EncodedKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAxPMbCA2Ks01nd7IR+3pw
\nwd3H/7jPGwj8bLUmore7bX+oeGpZ6QmLAe/1U0WcmZX2u70dYcSIzB1ofZtcn4cJ
\nenHBAz03ohBY/L1tQGJfS2A+omnN6H16VZE1JCK8XSJyfze7MDLcUyHZETdxuvRb
\nA9X343/vMAuQPnhinFJ8Wdy8YBXSPpy7r95y1UQd9LfYTBzVZYG2tSesplc0kjM3\n2Uu
+oMwxQAw1NINnSLPinMVsutJy6Zq1V3McwNWe4T+STGtWhrPNqJEn45sIcCx4\nnq
+kGZ2NQ0FyIyT2eiLK0X5RgB/a36E/aMk4VoDsaenBQgG7WLTnstb9sr7MIhS6A\nnrwIDAQAB\n-----END
PUBLIC KEY-----\n",
  "Comment": "example public key"
}
```


Output:

```
{
  "Location": "https://cloudfront.amazonaws.com/2019-03-26/public-key/
KDFB19YGCR002",
  "ETag": "E2QWRUHEXAMPLE",
  "PublicKey": {
    "Id": "KDFB19YGCR002",
    "CreatedTime": "2019-12-05T18:51:43.781Z",
    "PublicKeyConfig": {
      "CallerReference": "cli-example",
      "Name": "ExampleKey",
      "EncodedKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAxPMbCA2Ks0lnd7IR+3pw
\nwd3H/7jPGwj8bLUmore7bX+oeGpZ6QmLAe/1U0WcmZX2u70dYcSIzB1ofZtcn4cJ
\nenHBaz03ohBY/L1tQGJfS2A+omnN6H16VZE1JCK8XSJyfze7MDLcUyHZETdxuvRb
\nA9X343/vMAuQPNhinFJ8Wdy8YBXSPpy7r95y1UQd9LfYTBzVZYG2tSesplc0kjM3\n2Uu
+oMwXQAw1NINnSLPinMVsutJy6Zq1V3McWNWe4T+STGtWhrPNqJEn45sIcCx4\nnq
+kGZ2NQ0FyIyT2eiLK0X5Rgb/a36E/aMk4VoDsaenBQgG7WLTnstb9sr7MIhS6A\nnrwIDAQAB\n-----END
PUBLIC KEY-----\n",
      "Comment": "example public key"
    }
  }
}
```

- For API details, see [CreatePublicKey](#) in *AWS CLI Command Reference*.

delete-cloud-front-origin-access-identity

The following code example shows how to use `delete-cloud-front-origin-access-identity`.

AWS CLI**To delete a CloudFront origin access identity**

The following example deletes the origin access identity (OAI) with the ID `E74FTE3AEXAMPLE`. To delete an OAI, you must have the OAI's ID and ETag. The OAI ID is returned in the output of the `create-cloud-front-origin-access-identity` and `list-cloud-front-origin-access-identities` commands. To get the ETag, use the `get-cloud-front-origin-access-identity` or `get-cloud-front-origin-access-identity-config` command. Use the `--if-match` option to provide the OAI's ETag.

```
aws cloudfront delete-cloud-front-origin-access-identity \  
  --id E74FTE3AEXAMPLE \  
  --if-match E2QWRUHEXAMPLE
```

When successful, this command has no output.

- For API details, see [DeleteCloudFrontOriginAccessIdentity](#) in *AWS CLI Command Reference*.

delete-distribution

The following code example shows how to use `delete-distribution`.

AWS CLI

To delete a CloudFront distribution

The following example deletes the CloudFront distribution with the ID `EDFDVBD6EXAMPLE`. Before you can delete a distribution, you must disable it. To disable a distribution, use the `update-distribution` command. For more information, see the `update-distribution` examples.

When a distribution is disabled, you can delete it. To delete a distribution, you must use the `--if-match` option to provide the distribution's ETag. To get the ETag, use the `get-distribution` or `get-distribution-config` command.

```
aws cloudfront delete-distribution \  
  --id EDFDVBD6EXAMPLE \  
  --if-match E2QWRUHEXAMPLE
```

When successful, this command has no output.

- For API details, see [DeleteDistribution](#) in *AWS CLI Command Reference*.

delete-field-level-encryption-config

The following code example shows how to use `delete-field-level-encryption-config`.

AWS CLI

To delete a CloudFront field-level encryption configuration

The following example deletes the CloudFront field-level encryption configuration with the ID `C3KM2WVD605UAY`. To delete a field-level encryption configuration, you must have its ID

and ETag. The ID is returned in the output of the `create-field-level-encryption-config` and `list-field-level-encryption-configs` commands. To get the ETag, use the `get-field-level-encryption` or `get-field-level-encryption-config` command. Use the `--if-match` option to provide the configuration's ETag.

```
aws cloudfront delete-field-level-encryption-config \  
  --id C3KM2WVD605UAY \  
  --if-match E26M4BIAV81ZF6
```

When successful, this command has no output.

- For API details, see [DeleteFieldLevelEncryptionConfig](#) in *AWS CLI Command Reference*.

delete-field-level-encryption-profile

The following code example shows how to use `delete-field-level-encryption-profile`.

AWS CLI

To delete a CloudFront field-level encryption profile

The following example deletes the CloudFront field-level encryption profile with the ID `PPK0UOSIF5WSV`. To delete a field-level encryption profile, you must have its ID and ETag. The ID is returned in the output of the `create-field-level-encryption-profile` and `list-field-level-encryption-profiles` commands. To get the ETag, use the `get-field-level-encryption-profile` or `get-field-level-encryption-profile-config` command. Use the `--if-match` option to provide the profile's ETag.

```
aws cloudfront delete-field-level-encryption-profile \  
  --id PPK0UOSIF5WSV \  
  --if-match EJETYFJ9CL66D
```

When successful, this command has no output.

- For API details, see [DeleteFieldLevelEncryptionProfile](#) in *AWS CLI Command Reference*.

delete-public-key

The following code example shows how to use `delete-public-key`.

AWS CLI

To delete a CloudFront public key

The following example deletes the CloudFront public key with the ID KDFB19YGCR002. To delete a public key, you must have its ID and ETag. The ID is returned in the output of the `create-public-key` and `list-public-keys` commands. To get the ETag, use the `get-public-key` or `get-public-key-config` command. Use the `--if-match` option to provide the public key's ETag.

```
aws cloudfront delete-public-key \  
  --id KDFB19YGCR002 \  
  --if-match E2QWRUHEXAMPLE
```

When successful, this command has no output.

- For API details, see [DeletePublicKey](#) in *AWS CLI Command Reference*.

get-cloud-front-origin-access-identity-config

The following code example shows how to use `get-cloud-front-origin-access-identity-config`.

AWS CLI

To get a CloudFront origin access identity configuration

The following example gets metadata about the CloudFront origin access identity (OAI) with the ID E74FTE3AEXAMPLE, including its ETag. The OAI ID is returned in the output of the `create-cloud-front-origin-access-identity` and `list-cloud-front-origin-access-identities` commands.

```
aws cloudfront get-cloud-front-origin-access-identity-config --id E74FTE3AEXAMPLE
```

Output:

```
{  
  "ETag": "E2QWRUHEXAMPLE",  
  "CloudFrontOriginAccessIdentityConfig": {  
    "CallerReference": "cli-example",  
    "Comment": "Example OAI"  
  }  
}
```

- For API details, see [GetCloudFrontOriginAccessIdentityConfig](#) in *AWS CLI Command Reference*.

get-cloud-front-origin-access-identity

The following code example shows how to use `get-cloud-front-origin-access-identity`.

AWS CLI

To get a CloudFront origin access identity

The following example gets the CloudFront origin access identity (OAI) with the ID `E74FTE3AEXAMPLE`, including its ETag and the associated S3 canonical ID. The OAI ID is returned in the output of the `create-cloud-front-origin-access-identity` and `list-cloud-front-origin-access-identities` commands.

```
aws cloudfront get-cloud-front-origin-access-identity --id E74FTE3AEXAMPLE
```

Output:

```
{
  "ETag": "E2QWRUHEXAMPLE",
  "CloudFrontOriginAccessIdentity": {
    "Id": "E74FTE3AEXAMPLE",
    "S3CanonicalUserId":
"cd13868f797c227fbea2830611a26fe0a21ba1b826ab4bed9b7771c9aEXAMPLE",
    "CloudFrontOriginAccessIdentityConfig": {
      "CallerReference": "cli-example",
      "Comment": "Example OAI"
    }
  }
}
```

- For API details, see [GetCloudFrontOriginAccessIdentity](#) in *AWS CLI Command Reference*.

get-distribution-config

The following code example shows how to use `get-distribution-config`.

AWS CLI

To get a CloudFront distribution configuration

The following example gets metadata about the CloudFront distribution with the ID EDFDVBD6EXAMPLE, including its ETag. The distribution ID is returned in the create-distribution and list-distributions commands.

```
aws cloudfront get-distribution-config --id EDFDVBD6EXAMPLE
```

Output:

```
{
  "ETag": "E2QWRUHEXAMPLE",
  "DistributionConfig": {
    "CallerReference": "cli-example",
    "Aliases": {
      "Quantity": 0
    },
    "DefaultRootObject": "index.html",
    "Origins": {
      "Quantity": 1,
      "Items": [
        {
          "Id": "awsexamplebucket.s3.amazonaws.com-cli-example",
          "DomainName": "awsexamplebucket.s3.amazonaws.com",
          "OriginPath": "",
          "CustomHeaders": {
            "Quantity": 0
          },
          "S3OriginConfig": {
            "OriginAccessIdentity": ""
          }
        }
      ]
    },
    "OriginGroups": {
      "Quantity": 0
    },
    "DefaultCacheBehavior": {
      "TargetOriginId": "awsexamplebucket.s3.amazonaws.com-cli-example",
      "ForwardedValues": {
        "QueryString": false,
        "Cookies": {
          "Forward": "none"
        }
      },
      "Headers": {
```

```
        "Quantity": 0
      },
      "QueryStringCacheKeys": {
        "Quantity": 0
      }
    },
    "TrustedSigners": {
      "Enabled": false,
      "Quantity": 0
    },
    "ViewerProtocolPolicy": "allow-all",
    "MinTTL": 0,
    "AllowedMethods": {
      "Quantity": 2,
      "Items": [
        "HEAD",
        "GET"
      ],
      "CachedMethods": {
        "Quantity": 2,
        "Items": [
          "HEAD",
          "GET"
        ]
      }
    },
    "SmoothStreaming": false,
    "DefaultTTL": 86400,
    "MaxTTL": 31536000,
    "Compress": false,
    "LambdaFunctionAssociations": {
      "Quantity": 0
    },
    "FieldLevelEncryptionId": ""
  },
  "CacheBehaviors": {
    "Quantity": 0
  },
  "CustomErrorResponse": {
    "Quantity": 0
  },
  "Comment": "",
  "Logging": {
    "Enabled": false,
```

```
        "IncludeCookies": false,
        "Bucket": "",
        "Prefix": ""
    },
    "PriceClass": "PriceClass_All",
    "Enabled": true,
    "ViewerCertificate": {
        "CloudFrontDefaultCertificate": true,
        "MinimumProtocolVersion": "TLSv1",
        "CertificateSource": "cloudfront"
    },
    "Restrictions": {
        "GeoRestriction": {
            "RestrictionType": "none",
            "Quantity": 0
        }
    },
    "WebACLId": "",
    "HttpVersion": "http2",
    "IsIPV6Enabled": true
}
}
```

- For API details, see [GetDistributionConfig](#) in *AWS CLI Command Reference*.

get-distribution

The following code example shows how to use get-distribution.

AWS CLI

To get a CloudFront distribution

The following example gets the CloudFront distribution with the ID EDFDVBD6EXAMPLE, including its ETag. The distribution ID is returned in the create-distribution and list-distributions commands.

```
aws cloudfront get-distribution --id EDFDVBD6EXAMPLE
```

Output:

```
{
```



```
"ETag": "E2QWRUHEXAMPLE",
"Distribution": {
  "Id": "EDFDVBD6EXAMPLE",
  "ARN": "arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE",
  "Status": "Deployed",
  "LastModifiedTime": "2019-12-04T23:35:41.433Z",
  "InProgressInvalidationBatches": 0,
  "DomainName": "d111111abcdef8.cloudfront.net",
  "ActiveTrustedSigners": {
    "Enabled": false,
    "Quantity": 0
  },
  "DistributionConfig": {
    "CallerReference": "cli-example",
    "Aliases": {
      "Quantity": 0
    },
    "DefaultRootObject": "index.html",
    "Origins": {
      "Quantity": 1,
      "Items": [
        {
          "Id": "awsexamplebucket.s3.amazonaws.com-cli-example",
          "DomainName": "awsexamplebucket.s3.amazonaws.com",
          "OriginPath": "",
          "CustomHeaders": {
            "Quantity": 0
          },
          "S3OriginConfig": {
            "OriginAccessIdentity": ""
          }
        }
      ]
    },
    "OriginGroups": {
      "Quantity": 0
    },
    "DefaultCacheBehavior": {
      "TargetOriginId": "awsexamplebucket.s3.amazonaws.com-cli-example",
      "ForwardedValues": {
        "QueryString": false,
        "Cookies": {
          "Forward": "none"
        }
      }
    }
  }
}
```

```
    "Headers": {
      "Quantity": 0
    },
    "QueryStringCacheKeys": {
      "Quantity": 0
    }
  },
  "TrustedSigners": {
    "Enabled": false,
    "Quantity": 0
  },
  "ViewerProtocolPolicy": "allow-all",
  "MinTTL": 0,
  "AllowedMethods": {
    "Quantity": 2,
    "Items": [
      "HEAD",
      "GET"
    ],
    "CachedMethods": {
      "Quantity": 2,
      "Items": [
        "HEAD",
        "GET"
      ]
    }
  },
  "SmoothStreaming": false,
  "DefaultTTL": 86400,
  "MaxTTL": 31536000,
  "Compress": false,
  "LambdaFunctionAssociations": {
    "Quantity": 0
  },
  "FieldLevelEncryptionId": ""
},
"CacheBehaviors": {
  "Quantity": 0
},
"CustomErrorResponses": {
  "Quantity": 0
},
"Comment": "",
"Logging": {
```

```

        "Enabled": false,
        "IncludeCookies": false,
        "Bucket": "",
        "Prefix": ""
    },
    "PriceClass": "PriceClass_All",
    "Enabled": true,
    "ViewerCertificate": {
        "CloudFrontDefaultCertificate": true,
        "MinimumProtocolVersion": "TLSv1",
        "CertificateSource": "cloudfront"
    },
    "Restrictions": {
        "GeoRestriction": {
            "RestrictionType": "none",
            "Quantity": 0
        }
    },
    "WebACLId": "",
    "HttpVersion": "http2",
    "IsIPV6Enabled": true
}
}
}

```

- For API details, see [GetDistribution](#) in *AWS CLI Command Reference*.

get-field-level-encryption-config

The following code example shows how to use `get-field-level-encryption-config`.

AWS CLI

To get metadata about a CloudFront field-level encryption configuration

The following example gets metadata about the CloudFront field-level encryption configuration with the ID C3KM2WVD605UAY, including its ETag:

```
aws cloudfront get-field-level-encryption-config --id C3KM2WVD605UAY
```

Output:

```

{
  "ETag": "E2P4Z4VU7TY5SG",
  "FieldLevelEncryptionConfig": {
    "CallerReference": "cli-example",
    "Comment": "Example FLE configuration",
    "QueryArgProfileConfig": {
      "ForwardWhenQueryArgProfileIsUnknown": true,
      "QueryArgProfiles": {
        "Quantity": 0,
        "Items": []
      }
    },
    "ContentTypeProfileConfig": {
      "ForwardWhenContentTypeIsUnknown": true,
      "ContentTypeProfiles": {
        "Quantity": 1,
        "Items": [
          {
            "Format": "URLEncoded",
            "ProfileId": "P280MFCLSYOCVU",
            "ContentType": "application/x-www-form-urlencoded"
          }
        ]
      }
    }
  }
}

```

- For API details, see [GetFieldLevelEncryptionConfig](#) in *AWS CLI Command Reference*.

get-field-level-encryption-profile-config

The following code example shows how to use `get-field-level-encryption-profile-config`.

AWS CLI

To get a CloudFront field-level encryption profile configuration

The following example gets metadata about the CloudFront field-level encryption profile with ID `PPK0UOSIF5WSV`, including its ETag:

```
aws cloudfront get-field-level-encryption-profile-config --id PPK0U0SIF5WSV
```

Output:

```
{
  "ETag": "E1QQG65FS2L2GC",
  "FieldLevelEncryptionProfileConfig": {
    "Name": "ExampleFLEProfile",
    "CallerReference": "cli-example",
    "Comment": "FLE profile for AWS CLI example",
    "EncryptionEntities": {
      "Quantity": 1,
      "Items": [
        {
          "PublicKeyId": "K2K8NC4HVFE3M0",
          "ProviderId": "ExampleFLEProvider",
          "FieldPatterns": {
            "Quantity": 1,
            "Items": [
              "ExampleSensitiveField"
            ]
          }
        }
      ]
    }
  }
}
```

- For API details, see [GetFieldLevelEncryptionProfileConfig](#) in *AWS CLI Command Reference*.

get-field-level-encryption-profile

The following code example shows how to use `get-field-level-encryption-profile`.

AWS CLI

To get a CloudFront field-level encryption profile

The following example gets the CloudFront field-level encryption profile with ID `PPK0U0SIF5WSV`, including its ETag:

```
aws cloudfront get-field-level-encryption-profile --id PPK0U0SIF5WSV
```

Output:

```
{
  "ETag": "E1QQG65FS2L2GC",
  "FieldLevelEncryptionProfile": {
    "Id": "PPK0U0SIF5WSV",
    "LastModifiedTime": "2019-12-10T01:03:16.537Z",
    "FieldLevelEncryptionProfileConfig": {
      "Name": "ExampleFLEProfile",
      "CallerReference": "cli-example",
      "Comment": "FLE profile for AWS CLI example",
      "EncryptionEntities": {
        "Quantity": 1,
        "Items": [
          {
            "PublicKeyId": "K2K8NC4HVFE3M0",
            "ProviderId": "ExampleFLEProvider",
            "FieldPatterns": {
              "Quantity": 1,
              "Items": [
                "ExampleSensitiveField"
              ]
            }
          }
        ]
      }
    }
  }
}
```

- For API details, see [GetFieldLevelEncryptionProfile](#) in *AWS CLI Command Reference*.

get-field-level-encryption

The following code example shows how to use `get-field-level-encryption`.

AWS CLI

To get a CloudFront field-level encryption configuration

The following example gets the CloudFront field-level encryption configuration with the ID C3KM2WVD605UAY, including its ETag:

```
aws cloudfront get-field-level-encryption --id C3KM2WVD605UAY
```

Output:

```
{
  "ETag": "E2P4Z4VU7TY5SG",
  "FieldLevelEncryption": {
    "Id": "C3KM2WVD605UAY",
    "LastModifiedTime": "2019-12-10T21:30:18.974Z",
    "FieldLevelEncryptionConfig": {
      "CallerReference": "cli-example",
      "Comment": "Example FLE configuration",
      "QueryArgProfileConfig": {
        "ForwardWhenQueryArgProfileIsUnknown": true,
        "QueryArgProfiles": {
          "Quantity": 0,
          "Items": []
        }
      },
      "ContentTypeProfileConfig": {
        "ForwardWhenContentTypeIsUnknown": true,
        "ContentTypeProfiles": {
          "Quantity": 1,
          "Items": [
            {
              "Format": "URLEncoded",
              "ProfileId": "P280MFCLSYOCVU",
              "ContentType": "application/x-www-form-urlencoded"
            }
          ]
        }
      }
    }
  }
}
```

- For API details, see [GetFieldLevelEncryption](#) in *AWS CLI Command Reference*.

get-invalidation

The following code example shows how to use `get-invalidation`.

AWS CLI

To get a CloudFront invalidation

The following example gets the invalidation with the ID `I2J0I21PCUY0IK` for the CloudFront distribution with the ID `EDFDVBD6EXAMPLE`:

```
aws cloudfront get-invalidation --id I2J0I21PCUY0IK --distribution-id
EDFDVBD6EXAMPLE
```

Output:

```
{
  "Invalidation": {
    "Status": "Completed",
    "InvalidationBatch": {
      "Paths": {
        "Items": [
          "/example-path/example-file.jpg",
          "/example-path/example-file-2.jpg"
        ],
        "Quantity": 2
      },
      "CallerReference": "cli-example"
    },
    "Id": "I2J0I21PCUY0IK",
    "CreateTime": "2019-12-05T18:40:49.413Z"
  }
}
```

- For API details, see [GetInvalidation](#) in *AWS CLI Command Reference*.

get-public-key-config

The following code example shows how to use `get-public-key-config`.

AWS CLI

To get a CloudFront public key configuration

The following example gets metadata about the CloudFront public key with the ID `KDFB19YGCR002`, including its ETag. The public key ID is returned in the `create-public-key` and `list-public-keys` commands.

```
aws cloudfront get-public-key-config --id KDFB19YGCR002
```

Output:

```
{
  "ETag": "E2QWRUHEXAMPLE",
  "PublicKeyConfig": {
    "CallerReference": "cli-example",
    "Name": "ExampleKey",
    "EncodedKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAA0CAQ8AMIIBCgKCAQEAxPmbCA2Ks0lnd7IR+3pw
\nwd3H/7jPGwj8bLUmore7bX+oeGpZ6QmLAe/1U0WcmZX2u70dYcSIzB1ofZtcn4cJ
\nenHBAz03ohBY/L1tQGJfS2A+omnN6H16VZE1JCK8XSJyfze7MDLcUyHZETdxuvRb
\nA9X343/vMAuQPnhinFJ8Wdy8YBXSPpy7r95y1UQd9LfYTBzVZYG2tSesplc0kjM3\n2Uu
+oMwxQAw1NINnSLPinMVsutJy6Zq1V3McWNWe4T+STGtWhrPNqJEn45sIcCx4\nnq
+kGZ2NQ0FyIyT2eiLK0X5Rgb/a36E/aMk4VoDsaenBQgG7WLTnstb9sr7MIhS6A\nnrwIDAQAB\n-----END
PUBLIC KEY-----\n",
    "Comment": "example public key"
  }
}
```

- For API details, see [GetPublicKeyConfig](#) in *AWS CLI Command Reference*.

get-public-key

The following code example shows how to use `get-public-key`.

AWS CLI

To get a CloudFront public key

The following example gets the CloudFront public key with the ID `KDFB19YGCR002`, including its ETag. The public key ID is returned in the `create-public-key` and `list-public-keys` commands.

```
aws cloudfront get-public-key --id KDFB19YGCR002
```

Output:

```
{
  "ETag": "E2QWRUHEXAMPLE",
  "PublicKey": {
    "Id": "KDFB19YGCR002",
    "CreatedTime": "2019-12-05T18:51:43.781Z",
    "PublicKeyConfig": {
      "CallerReference": "cli-example",
      "Name": "ExampleKey",
      "EncodedKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAxPmBCA2Ks01nd7IR+3pw
\nwd3H/7jPGwj8bLUmore7bX+oeGpZ6QmLae/1U0WcmZX2u70dYcSIzB1ofZtcn4cJ
\nenHBAz03ohBY/L1tQGJfS2A+omnN6H16VZE1JCK8XSJyfze7MDLcUyHZETdxuvRb
\nA9X343/vMAuQPnhinFJ8Wdy8YBXSPpy7r95y1UQd9LfYTBzVZYG2tSesplc0kjM3\n2Uu
+oMwxQAw1NINnSLPinMVsutJy6Zq1V3McWNWe4T+STGtWhrPNqJEn45sIcCx4\nnq
+kGZ2NQ0FyIyT2eiLK0X5RgB/a36E/aMk4VoDsaenBQgG7WLTnstb9sr7MIhS6A\nnrwIDAQAB\n-----END
PUBLIC KEY-----\n",
      "Comment": "example public key"
    }
  }
}
```

- For API details, see [GetPublicKey](#) in *AWS CLI Command Reference*.

list-cloud-front-origin-access-identities

The following code example shows how to use `list-cloud-front-origin-access-identities`.

AWS CLI

To list CloudFront origin access identities

The following example gets a list of the CloudFront origin access identities (OAs) in your AWS account:

```
aws cloudfront list-cloud-front-origin-access-identities
```

Output:

```
{
  "CloudFrontOriginAccessIdentityList": {
    "Items": [
```

```

    {
      "Id": "E74FTE3AEXAMPLE",
      "S3CanonicalUserId":
"cd13868f797c227fbea2830611a26fe0a21ba1b826ab4bed9b7771c9aEXAMPLE",
      "Comment": "Example OAI"
    },
    {
      "Id": "EH1HDMBEXAMPLE",
      "S3CanonicalUserId":
"1489f6f2e6faacaae7ff64c4c3e6956c24f78788abfc1718c3527c263bf7a17EXAMPLE",
      "Comment": "Test OAI"
    },
    {
      "Id": "E2X2C9TEXAMPLE",
      "S3CanonicalUserId":
"cbfeebb915a64749f9be546a45b3fcfd3a31c779673c13c4dd460911ae402c2EXAMPLE",
      "Comment": "Example OAI #2"
    }
  ]
}

```

- For API details, see [ListCloudFrontOriginAccessIdentities](#) in *AWS CLI Command Reference*.

list-distributions

The following code example shows how to use `list-distributions`.

AWS CLI

To list CloudFront distributions

The following example gets a list of the CloudFront distributions in your AWS account:

```
aws cloudfront list-distributions
```

Output:

```

{
  "DistributionList": {
    "Items": [
      {
        "Id": "EMLARXS9EXAMPLE",

```

```

    "ARN": "arn:aws:cloudfront::123456789012:distribution/
EMLARXS9EXAMPLE",
    "Status": "InProgress",
    "LastModifiedTime": "2019-11-22T00:55:15.705Z",
    "InProgressInvalidationBatches": 0,
    "DomainName": "d111111abcdef8.cloudfront.net",
    "ActiveTrustedSigners": {
        "Enabled": false,
        "Quantity": 0
    },
    "DistributionConfig": {
        "CallerReference": "cli-example",
        "Aliases": {
            "Quantity": 0
        },
        "DefaultRootObject": "index.html",
        "Origins": {
            "Quantity": 1,
            "Items": [
                {
                    "Id": "awsexamplebucket.s3.amazonaws.com-cli-
example",
                    "DomainName": "awsexamplebucket.s3.amazonaws.com",
                    "OriginPath": "",
                    "CustomHeaders": {
                        "Quantity": 0
                    },
                    "S3OriginConfig": {
                        "OriginAccessIdentity": ""
                    }
                }
            ]
        },
        "OriginGroups": {
            "Quantity": 0
        },
        "DefaultCacheBehavior": {
            "TargetOriginId": "awsexamplebucket.s3.amazonaws.com-cli-
example",
            "ForwardedValues": {
                "QueryString": false,
                "Cookies": {
                    "Forward": "none"
                }
            },

```

```
    "Headers": {
      "Quantity": 0
    },
    "QueryStringCacheKeys": {
      "Quantity": 0
    }
  },
  "TrustedSigners": {
    "Enabled": false,
    "Quantity": 0
  },
  "ViewerProtocolPolicy": "allow-all",
  "MinTTL": 0,
  "AllowedMethods": {
    "Quantity": 2,
    "Items": [
      "HEAD",
      "GET"
    ],
    "CachedMethods": {
      "Quantity": 2,
      "Items": [
        "HEAD",
        "GET"
      ]
    }
  },
  "SmoothStreaming": false,
  "DefaultTTL": 86400,
  "MaxTTL": 31536000,
  "Compress": false,
  "LambdaFunctionAssociations": {
    "Quantity": 0
  },
  "FieldLevelEncryptionId": ""
},
"CacheBehaviors": {
  "Quantity": 0
},
"CustomErrorResponses": {
  "Quantity": 0
},
"Comment": "",
"Logging": {
```

```

        "Enabled": false,
        "IncludeCookies": false,
        "Bucket": "",
        "Prefix": ""
    },
    "PriceClass": "PriceClass_All",
    "Enabled": true,
    "ViewerCertificate": {
        "CloudFrontDefaultCertificate": true,
        "MinimumProtocolVersion": "TLSv1",
        "CertificateSource": "cloudfront"
    },
    "Restrictions": {
        "GeoRestriction": {
            "RestrictionType": "none",
            "Quantity": 0
        }
    },
    "WebACLId": "",
    "HttpVersion": "http2",
    "IsIPV6Enabled": true
}
},
{
    "Id": "EDFDVBD6EXAMPLE",
    "ARN": "arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE",
    "Status": "InProgress",
    "LastModifiedTime": "2019-12-04T23:35:41.433Z",
    "InProgressInvalidationBatches": 0,
    "DomainName": "d930174dauwrn8.cloudfront.net",
    "ActiveTrustedSigners": {
        "Enabled": false,
        "Quantity": 0
    },
    "DistributionConfig": {
        "CallerReference": "cli-example",
        "Aliases": {
            "Quantity": 0
        },
        "DefaultRootObject": "index.html",
        "Origins": {
            "Quantity": 1,
            "Items": [

```

```

        {
            "Id": "awsexamplebucket1.s3.amazonaws.com-cli-
example",
            "DomainName": "awsexamplebucket1.s3.amazonaws.com",
            "OriginPath": "",
            "CustomHeaders": {
                "Quantity": 0
            },
            "S3OriginConfig": {
                "OriginAccessIdentity": ""
            }
        }
    ],
    },
    "OriginGroups": {
        "Quantity": 0
    },
    },
    "DefaultCacheBehavior": {
        "TargetOriginId": "awsexamplebucket1.s3.amazonaws.com-cli-
example",
        "ForwardedValues": {
            "QueryString": false,
            "Cookies": {
                "Forward": "none"
            },
        },
        "Headers": {
            "Quantity": 0
        },
        "QueryStringCacheKeys": {
            "Quantity": 0
        }
    },
    },
    "TrustedSigners": {
        "Enabled": false,
        "Quantity": 0
    },
    },
    "ViewerProtocolPolicy": "allow-all",
    "MinTTL": 0,
    "AllowedMethods": {
        "Quantity": 2,
        "Items": [
            "HEAD",
            "GET"
        ],
    },

```

```
        "CachedMethods": {
            "Quantity": 2,
            "Items": [
                "HEAD",
                "GET"
            ]
        }
    },
    "SmoothStreaming": false,
    "DefaultTTL": 86400,
    "MaxTTL": 31536000,
    "Compress": false,
    "LambdaFunctionAssociations": {
        "Quantity": 0
    },
    "FieldLevelEncryptionId": ""
},
"CacheBehaviors": {
    "Quantity": 0
},
"CustomErrorResponse": {
    "Quantity": 0
},
"Comment": "",
"Logging": {
    "Enabled": false,
    "IncludeCookies": false,
    "Bucket": "",
    "Prefix": ""
},
"PriceClass": "PriceClass_All",
"Enabled": true,
"ViewerCertificate": {
    "CloudFrontDefaultCertificate": true,
    "MinimumProtocolVersion": "TLSv1",
    "CertificateSource": "cloudfront"
},
"Restrictions": {
    "GeoRestriction": {
        "RestrictionType": "none",
        "Quantity": 0
    }
},
"WebACLId": "",
```



```

        "HttpVersion": "http2",
        "IsIPV6Enabled": true
    }
},
{
    "Id": "E1X5IZQEXAMPLE",
    "ARN": "arn:aws:cloudfront::123456789012:distribution/
E1X5IZQEXAMPLE",
    "Status": "Deployed",
    "LastModifiedTime": "2019-11-06T21:31:48.864Z",
    "DomainName": "d2e04y12345678.cloudfront.net",
    "Aliases": {
        "Quantity": 0
    },
    "Origins": {
        "Quantity": 1,
        "Items": [
            {
                "Id": "awsexamplebucket2",
                "DomainName": "awsexamplebucket2.s3.us-
west-2.amazonaws.com",
                "OriginPath": "",
                "CustomHeaders": {
                    "Quantity": 0
                },
                "S3OriginConfig": {
                    "OriginAccessIdentity": ""
                }
            }
        ]
    },
    "OriginGroups": {
        "Quantity": 0
    },
    "DefaultCacheBehavior": {
        "TargetOriginId": "awsexamplebucket2",
        "ForwardedValues": {
            "QueryString": false,
            "Cookies": {
                "Forward": "none"
            },
            "Headers": {
                "Quantity": 0
            }
        }
    }
},

```

```
        "QueryStringCacheKeys": {
            "Quantity": 0
        },
    },
    "TrustedSigners": {
        "Enabled": false,
        "Quantity": 0
    },
    "ViewerProtocolPolicy": "allow-all",
    "MinTTL": 0,
    "AllowedMethods": {
        "Quantity": 2,
        "Items": [
            "HEAD",
            "GET"
        ],
    },
    "CachedMethods": {
        "Quantity": 2,
        "Items": [
            "HEAD",
            "GET"
        ]
    },
    "SmoothStreaming": false,
    "DefaultTTL": 86400,
    "MaxTTL": 31536000,
    "Compress": false,
    "LambdaFunctionAssociations": {
        "Quantity": 0
    },
    "FieldLevelEncryptionId": ""
},
"CacheBehaviors": {
    "Quantity": 0
},
"CustomErrorResponse": {
    "Quantity": 0
},
"Comment": "",
"PriceClass": "PriceClass_All",
"Enabled": true,
"ViewerCertificate": {
    "CloudFrontDefaultCertificate": true,
```

```

        "MinimumProtocolVersion": "TLSv1",
        "CertificateSource": "cloudfront"
    },
    "Restrictions": {
        "GeoRestriction": {
            "RestrictionType": "none",
            "Quantity": 0
        }
    },
    "WebACLId": "",
    "HttpVersion": "HTTP1_1",
    "IsIPV6Enabled": true
}
]
}

```

- For API details, see [ListDistributions](#) in *AWS CLI Command Reference*.

list-field-level-encryption-configs

The following code example shows how to use `list-field-level-encryption-configs`.

AWS CLI

To list CloudFront field-level encryption configurations

The following example gets a list of the CloudFront field-level encryption configurations in your AWS account:

```
aws cloudfront list-field-level-encryption-configs
```

Output:

```

{
  "FieldLevelEncryptionList": {
    "MaxItems": 100,
    "Quantity": 1,
    "Items": [
      {
        "Id": "C3KM2WVD605UAY",
        "LastModifiedTime": "2019-12-10T21:30:18.974Z",

```

```

    "Comment": "Example FLE configuration",
    "QueryArgProfileConfig": {
      "ForwardWhenQueryArgProfileIsUnknown": true,
      "QueryArgProfiles": {
        "Quantity": 0,
        "Items": []
      }
    },
    "ContentTypeProfileConfig": {
      "ForwardWhenContentTypeIsUnknown": true,
      "ContentTypeProfiles": {
        "Quantity": 1,
        "Items": [
          {
            "Format": "URLEncoded",
            "ProfileId": "P280MFCLSY0CVU",
            "ContentType": "application/x-www-form-urlencoded"
          }
        ]
      }
    }
  }
]
}

```

- For API details, see [ListFieldLevelEncryptionConfigs](#) in *AWS CLI Command Reference*.

list-field-level-encryption-profiles

The following code example shows how to use `list-field-level-encryption-profiles`.

AWS CLI

To list CloudFront field-level encryption profiles

The following example gets a list of the CloudFront field-level encryption profiles in your AWS account:

```
aws cloudfront list-field-level-encryption-profiles
```

Output:

```
{
  "FieldLevelEncryptionProfileList": {
    "MaxItems": 100,
    "Quantity": 2,
    "Items": [
      {
        "Id": "P280MFCLSY0CVU",
        "LastModifiedTime": "2019-12-05T01:05:39.896Z",
        "Name": "ExampleFLEProfile",
        "EncryptionEntities": {
          "Quantity": 1,
          "Items": [
            {
              "PublicKeyId": "K2K8NC4HVFE3M0",
              "ProviderId": "ExampleFLEProvider",
              "FieldPatterns": {
                "Quantity": 1,
                "Items": [
                  "ExampleSensitiveField"
                ]
              }
            }
          ]
        },
        "Comment": "FLE profile for AWS CLI example"
      },
      {
        "Id": "PPK0UOSIF5WSV",
        "LastModifiedTime": "2019-12-10T01:03:16.537Z",
        "Name": "ExampleFLEProfile2",
        "EncryptionEntities": {
          "Quantity": 1,
          "Items": [
            {
              "PublicKeyId": "K2ABC10EXAMPLE",
              "ProviderId": "ExampleFLEProvider2",
              "FieldPatterns": {
                "Quantity": 1,
                "Items": [
                  "ExampleSensitiveField2"
                ]
              }
            }
          ]
        }
      }
    ]
  }
}
```

```

        ]
      },
      "Comment": "FLE profile #2 for AWS CLI example"
    }
  ]
}

```

- For API details, see [ListFieldLevelEncryptionProfiles](#) in *AWS CLI Command Reference*.

list-invalidations

The following code example shows how to use `list-invalidations`.

AWS CLI

To list CloudFront invalidations

The following example gets a list of the invalidations for the CloudFront distribution with the ID EDFDVBD6EXAMPLE:

```
aws cloudfront list-invalidations --distribution-id EDFDVBD6EXAMPLE
```

Output:

```

{
  "InvalidationList": {
    "Marker": "",
    "Items": [
      {
        "Status": "Completed",
        "Id": "YNY2LI2BVJ4NJU",
        "CreateTime": "2019-08-31T21:15:52.042Z"
      }
    ],
    "IsTruncated": false,
    "MaxItems": 100,
    "Quantity": 1
  }
}

```

- For API details, see [ListInvalidations](#) in *AWS CLI Command Reference*.

list-public-keys

The following code example shows how to use `list-public-keys`.

AWS CLI

To list CloudFront public keys

The following example gets a list of the CloudFront public keys in your AWS account:

```
aws cloudfront list-public-keys
```

Output:

```
{
  "PublicKeyList": {
    "MaxItems": 100,
    "Quantity": 2,
    "Items": [
      {
        "Id": "K2K8NC4HVFE3M0",
        "Name": "ExampleKey",
        "CreatedTime": "2019-12-05T01:04:28.818Z",
        "EncodedKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAA0CAQ8AMIIBCgKCAQEApMbCA2Ks01nd7IR+3pw
\nwd3H/7jPGwj8bLUmore7bX+oeGpZ6QmLAe/1U0WcmZX2u70dYcSIzB1ofZtcn4cJ
\nenHBAz03ohBY/L1tQGJfS2A+omnN6H16VZE1JCK8XSJyfze7MDLcUyHZETdxuvRb
\nA9X343/vMAuQPnhinFJ8Wdy8YBXSPpy7r95y1UQd9LfYTBzVZYG2tSesplc0kjm3\n2Uu
+oMWxQAw1NINnSLPinMVsutJy6Zq1V3McWNWe4T+STGtWhrPNqJEn45sIcCx4\nnq
+kGZ2NQ0FyIyT2eiLK0X5RgB/a36E/aMk4VoDsaenBQgG7WLTnstb9sr7MIhS6A\nnrwIDAQAB\n-----END
PUBLIC KEY-----\n",
        "Comment": "example public key"
      },
      {
        "Id": "K1S0LWQ2L5HTBU",
        "Name": "ExampleKey2",
        "CreatedTime": "2019-12-09T23:28:11.110Z",
        "EncodedKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAA0CAQ8AMIIBCgKCAQEAp0CAG88A8+f4dujn9Izt
\n26LxtgAkn2opGgo/NKpMiaisyw5qlg3f1gol17FV6pYN178iJg3E08JBbwt1H
+cR9\nLGSf60NDeVhm760c39Np/vWg0dsGQcRbi9WmKZeS0DqjQGzVZWqPmito3FzWV6b
\nfVY5N36U/RdbVAJm95Km+qaMY1bIdF40t72bi3IkKYV5h1B2XoDjlQ9F6ajQKyTB
\nMHa3SN8q+3ZjQ4sJJ7D1V6r4wR8jDcFVD5NckWJmngIVnk0QM37NYeoDnka0uTpu\nnha/
```

```
+3b8t0b2z3LBVHPkp85zJRA0XacSwf5rZtPYKBNFsixTa2n55k2r218m0kMC4\nUwIDAQAB\n-----END
PUBLIC KEY-----",
    "Comment": "example public key #2"
  }
]
}
}
```

- For API details, see [ListPublicKeys](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags for a CloudFront distribution

The following example gets a list of the tags for a CloudFront distribution:

```
aws cloudfront list-tags-for-resource \
  --resource arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE
```

Output:

```
{
  "Tags": {
    "Items": [
      {
        "Key": "DateCreated",
        "Value": "2019-12-04"
      },
      {
        "Key": "Name",
        "Value": "Example name"
      },
      {
        "Key": "Project",
        "Value": "Example project"
      }
    ]
  }
}
```



```
}
```

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

sign

The following code example shows how to use `sign`.

AWS CLI

To sign a CloudFront URL

The following example signs a CloudFront URL. To sign a URL, you need the key pair ID (called the **Access Key ID** in the AWS Management Console) and the private key of the trusted signer's CloudFront key pair. For more information about signed URLs, see [Serving Private Content with Signed URLs and Signed Cookies](#) in the *Amazon CloudFront Developer Guide*.

```
aws cloudfront sign \  
  --url https://d111111abcdef8.cloudfront.net/private-content/private-file.html \  
  --key-pair-id APKAEIBAERJR2EXAMPLE \  
  --private-key file://cf-signer-priv-key.pem \  
  --date-less-than 2020-01-01
```

Output:

```
https://d111111abcdef8.cloudfront.net/private-content/private-  
file.html?Expires=1577836800&Signature=nEXK7Kby47XKeZQKvc6pwkif6oZc-  
JWSpDkH0UH7EBGGqvgurkecCbgL5VfUAXyLQuJxFwRQWscz-  
owcq9KpmewCXrXQbPaJZNi9XSNwf4YKurPDQYaRQawKoenH0GFteRf9ELK-  
Bs3n1jTLjtbgzIUt7QJNKXcWr8AuUYikzGdJ4-qzx6WnxXfH~fxg4-  
GGl6l2kgCpXUB6Jx6K~Y3kpV0dzUP0IqFLHANJojbhxqrVejomZZ2XrquDvNUCCIbePGnR3d24UPaLXG4FK0qNEaWDIB  
GNvjRJxqWf93uMobeM0iVYahb-e0KIitiQewGcm0eLZQ__&Key-Pair-Id=APKAEIBAERJR2EXAMPLE
```

- For API details, see [Sign](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To tag a CloudFront distribution

The following `tag-resource` example adds two tags to the specified CloudFront distribution.

```
aws cloudfront tag-resource \  
  --resource arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE \  
  --tags 'Items=[{Key=Name,Value="Example name"},{Key=Project,Value="Example project"}]'
```

Instead of using command line arguments, you can provide the tags in a JSON file, as shown in the following example:

```
aws cloudfront tag-resource \  
  --resource arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE \  
  --tags file://tags.json
```

Contents of `tags.json`:

```
{  
  "Items": [  
    {  
      "Key": "Name",  
      "Value": "Example name"  
    },  
    {  
      "Key": "Project",  
      "Value": "Example project"  
    }  
  ]  
}
```

This command produces no output.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from a CloudFront distribution

The following example removes two tags from a CloudFront distribution by using command line arguments:

```
aws cloudfront untag-resource \  
  --resource arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE \  
  --tag-keys Items=Name,Project
```

Instead of using command line arguments, you can provide the tag keys in a JSON file, as shown in the following example:

```
aws cloudfront untag-resource \  
  --resource arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE \  
  --tag-keys file://tag-keys.json
```

The file `tag-keys.json` is a JSON document in the current folder that contains the following:

```
{  
  "Items": [  
    "Name",  
    "Project"  
  ]  
}
```

When successful, this command has no output.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-cloud-front-origin-access-identity

The following code example shows how to use `update-cloud-front-origin-access-identity`.

AWS CLI

To update a CloudFront origin access identity

The following example updates the origin access identity (OAI) with the ID `E74FTE3AEXAMPLE`. The only field that you can update is the OAI's `Comment`.

To update an OAI, you must have the OAI's ID and ETag. The OAI ID is returned in the output of the `create-cloud-front-origin-access-identity` and `list-cloud-front-origin-access-identities` commands. To get the ETag, use the `get-cloud-front-origin-access-identity` or `get-cloud-front-origin-access-identity-config` command. Use the `--if-match` option to provide the OAI's ETag.

```
aws cloudfront update-cloud-front-origin-access-identity \  
  --id E74FTE3AEXAMPLE \  
  --if-match E2QWRUHEXAMPLE \  
  --cloud-front-origin-access-identity-config \  
    CallerReference=cli-example,Comment="Example OAI Updated"
```

You can accomplish the same thing by providing the OAI configuration in a JSON file, as shown in the following example:

```
aws cloudfront update-cloud-front-origin-access-identity \  
  --id E74FTE3AEXAMPLE \  
  --if-match E2QWRUHEXAMPLE \  
  --cloud-front-origin-access-identity-config file://OAI-config.json
```

The file `OAI-config.json` is a JSON document in the current directory that contains the following:

```
{  
  "CallerReference": "cli-example",  
  "Comment": "Example OAI Updated"  
}
```

Whether you provide the OAI configuration with a command line argument or a JSON file, the output is the same:

```
{  
  "ETag": "E9LHASXEXAMPLE",  
  "CloudFrontOriginAccessIdentity": {  
    "Id": "E74FTE3AEXAMPLE",  
    "S3CanonicalUserId":  
      "cd13868f797c227fbea2830611a26fe0a21ba1b826ab4bed9b7771c9aEXAMPLE",  
    "CloudFrontOriginAccessIdentityConfig": {  
      "CallerReference": "cli-example",  
      "Comment": "Example OAI Updated"  
    }  
  }  
}
```

```
}
```

- For API details, see [UpdateCloudFrontOriginAccessIdentity](#) in *AWS CLI Command Reference*.

update-distribution

The following code example shows how to use update-distribution.

AWS CLI

To update a CloudFront distribution's default root object

The following example updates the default root object to `index.html` for the CloudFront distribution with the ID `EDFDVBD6EXAMPLE`:

```
aws cloudfront update-distribution --id EDFDVBD6EXAMPLE \  
  --default-root-object index.html
```

Output:

```
{  
  "ETag": "E2QWRUHEXAMPLE",  
  "Distribution": {  
    "Id": "EDFDVBD6EXAMPLE",  
    "ARN": "arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE",  
    "Status": "InProgress",  
    "LastModifiedTime": "2019-12-06T18:55:39.870Z",  
    "InProgressInvalidationBatches": 0,  
    "DomainName": "d111111abcdef8.cloudfront.net",  
    "ActiveTrustedSigners": {  
      "Enabled": false,  
      "Quantity": 0  
    },  
    "DistributionConfig": {  
      "CallerReference": "6b10378d-49be-4c4b-a642-419ccaf8f3b5",  
      "Aliases": {  
        "Quantity": 0  
      },  
      "DefaultRootObject": "index.html",  
      "Origins": {  
        "Quantity": 1,  
        "Items": [  

```

```
    {
      "Id": "example-website",
      "DomainName": "www.example.com",
      "OriginPath": "",
      "CustomHeaders": {
        "Quantity": 0
      },
      "CustomOriginConfig": {
        "HTTPPort": 80,
        "HTTPSPort": 443,
        "OriginProtocolPolicy": "match-viewer",
        "OriginSslProtocols": {
          "Quantity": 2,
          "Items": [
            "SSLv3",
            "TLSv1"
          ]
        },
        "OriginReadTimeout": 30,
        "OriginKeepaliveTimeout": 5
      }
    }
  ],
  "OriginGroups": {
    "Quantity": 0
  },
  "DefaultCacheBehavior": {
    "TargetOriginId": "example-website",
    "ForwardedValues": {
      "QueryString": false,
      "Cookies": {
        "Forward": "none"
      }
    },
    "Headers": {
      "Quantity": 1,
      "Items": [
        "*"
      ]
    },
    "QueryStringCacheKeys": {
      "Quantity": 0
    }
  },
},
```

```
    "TrustedSigners": {
      "Enabled": false,
      "Quantity": 0
    },
    "ViewerProtocolPolicy": "allow-all",
    "MinTTL": 0,
    "AllowedMethods": {
      "Quantity": 2,
      "Items": [
        "HEAD",
        "GET"
      ],
      "CachedMethods": {
        "Quantity": 2,
        "Items": [
          "HEAD",
          "GET"
        ]
      }
    },
    "SmoothStreaming": false,
    "DefaultTTL": 86400,
    "MaxTTL": 31536000,
    "Compress": false,
    "LambdaFunctionAssociations": {
      "Quantity": 0
    },
    "FieldLevelEncryptionId": ""
  },
  "CacheBehaviors": {
    "Quantity": 0
  },
  "CustomErrorResponses": {
    "Quantity": 0
  },
  "Comment": "",
  "Logging": {
    "Enabled": false,
    "IncludeCookies": false,
    "Bucket": "",
    "Prefix": ""
  },
  "PriceClass": "PriceClass_All",
  "Enabled": true,
```

```

    "ViewerCertificate": {
      "CloudFrontDefaultCertificate": true,
      "MinimumProtocolVersion": "TLSv1",
      "CertificateSource": "cloudfront"
    },
    "Restrictions": {
      "GeoRestriction": {
        "RestrictionType": "none",
        "Quantity": 0
      }
    },
    "WebACLId": "",
    "HttpVersion": "http1.1",
    "IsIPV6Enabled": true
  }
}

```

To update a CloudFront distribution

The following example disables the CloudFront distribution with the ID EMLARXS9EXAMPLE by providing the distribution configuration in a JSON file named `dist-config-disable.json`. To update a distribution, you must use the `--if-match` option to provide the distribution's ETag. To get the ETag, use the `get-distribution` or `get-distribution-config` command.

After you use the following example to disable a distribution, you can use the `delete-distribution` command to delete it.

```

aws cloudfront update-distribution \
  --id EMLARXS9EXAMPLE \
  --if-match E2QWRUHEXAMPLE \
  --distribution-config file://dist-config-disable.json

```

The file `dist-config-disable.json` is a JSON document in the current folder that contains the following. Note that the `Enabled` field is set to `false`:

```

{
  "CallerReference": "cli-1574382155-496510",
  "Aliases": {
    "Quantity": 0
  },
  "DefaultRootObject": "index.html",

```



```
"Origins": {
  "Quantity": 1,
  "Items": [
    {
      "Id": "awsexamplebucket.s3.amazonaws.com-1574382155-273939",
      "DomainName": "awsexamplebucket.s3.amazonaws.com",
      "OriginPath": "",
      "CustomHeaders": {
        "Quantity": 0
      },
      "S3OriginConfig": {
        "OriginAccessIdentity": ""
      }
    }
  ]
},
"OriginGroups": {
  "Quantity": 0
},
"DefaultCacheBehavior": {
  "TargetOriginId": "awsexamplebucket.s3.amazonaws.com-1574382155-273939",
  "ForwardedValues": {
    "QueryString": false,
    "Cookies": {
      "Forward": "none"
    },
  },
  "Headers": {
    "Quantity": 0
  },
  "QueryStringCacheKeys": {
    "Quantity": 0
  }
},
"TrustedSigners": {
  "Enabled": false,
  "Quantity": 0
},
"ViewerProtocolPolicy": "allow-all",
"MinTTL": 0,
"AllowedMethods": {
  "Quantity": 2,
  "Items": [
    "HEAD",
    "GET"
  ]
}
```

```
    ],
    "CachedMethods": {
      "Quantity": 2,
      "Items": [
        "HEAD",
        "GET"
      ]
    }
  },
  "SmoothStreaming": false,
  "DefaultTTL": 86400,
  "MaxTTL": 31536000,
  "Compress": false,
  "LambdaFunctionAssociations": {
    "Quantity": 0
  },
  "FieldLevelEncryptionId": ""
},
"CacheBehaviors": {
  "Quantity": 0
},
"CustomErrorResponses": {
  "Quantity": 0
},
"Comment": "",
"Logging": {
  "Enabled": false,
  "IncludeCookies": false,
  "Bucket": "",
  "Prefix": ""
},
"PriceClass": "PriceClass_All",
"Enabled": false,
"ViewerCertificate": {
  "CloudFrontDefaultCertificate": true,
  "MinimumProtocolVersion": "TLSv1",
  "CertificateSource": "cloudfront"
},
"Restrictions": {
  "GeoRestriction": {
    "RestrictionType": "none",
    "Quantity": 0
  }
},
},
```

```

    "WebACLId": "",
    "HttpVersion": "http2",
    "IsIPV6Enabled": true
  }

```

Output:

```

{
  "ETag": "E9LHASXEXAMPLE",
  "Distribution": {
    "Id": "EMLARXS9EXAMPLE",
    "ARN": "arn:aws:cloudfront::123456789012:distribution/EMLARXS9EXAMPLE",
    "Status": "InProgress",
    "LastModifiedTime": "2019-12-06T18:32:35.553Z",
    "InProgressInvalidationBatches": 0,
    "DomainName": "d1111111abcdef8.cloudfront.net",
    "ActiveTrustedSigners": {
      "Enabled": false,
      "Quantity": 0
    },
  },
  "DistributionConfig": {
    "CallerReference": "cli-1574382155-496510",
    "Aliases": {
      "Quantity": 0
    },
    "DefaultRootObject": "index.html",
    "Origins": {
      "Quantity": 1,
      "Items": [
        {
          "Id": "awsexamplebucket.s3.amazonaws.com-1574382155-273939",
          "DomainName": "awsexamplebucket.s3.amazonaws.com",
          "OriginPath": "",
          "CustomHeaders": {
            "Quantity": 0
          },
          "S3OriginConfig": {
            "OriginAccessIdentity": ""
          }
        }
      ]
    },
  },
  "OriginGroups": {

```

```
    "Quantity": 0
  },
  "DefaultCacheBehavior": {
    "TargetOriginId":
"awsexamplebucket.s3.amazonaws.com-1574382155-273939",
    "ForwardedValues": {
      "QueryString": false,
      "Cookies": {
        "Forward": "none"
      },
      "Headers": {
        "Quantity": 0
      },
      "QueryStringCacheKeys": {
        "Quantity": 0
      }
    },
    "TrustedSigners": {
      "Enabled": false,
      "Quantity": 0
    },
    "ViewerProtocolPolicy": "allow-all",
    "MinTTL": 0,
    "AllowedMethods": {
      "Quantity": 2,
      "Items": [
        "HEAD",
        "GET"
      ],
      "CachedMethods": {
        "Quantity": 2,
        "Items": [
          "HEAD",
          "GET"
        ]
      }
    },
    "SmoothStreaming": false,
    "DefaultTTL": 86400,
    "MaxTTL": 31536000,
    "Compress": false,
    "LambdaFunctionAssociations": {
      "Quantity": 0
    },
  },
```

```

        "FieldLevelEncryptionId": ""
    },
    "CacheBehaviors": {
        "Quantity": 0
    },
    "CustomErrorResponses": {
        "Quantity": 0
    },
    "Comment": "",
    "Logging": {
        "Enabled": false,
        "IncludeCookies": false,
        "Bucket": "",
        "Prefix": ""
    },
    "PriceClass": "PriceClass_All",
    "Enabled": false,
    "ViewerCertificate": {
        "CloudFrontDefaultCertificate": true,
        "MinimumProtocolVersion": "TLSv1",
        "CertificateSource": "cloudfront"
    },
    "Restrictions": {
        "GeoRestriction": {
            "RestrictionType": "none",
            "Quantity": 0
        }
    },
    "WebACLId": "",
    "HttpVersion": "http2",
    "IsIPV6Enabled": true
    }
}
}
}

```

- For API details, see [UpdateDistribution](#) in *AWS CLI Command Reference*.

update-field-level-encryption-config

The following code example shows how to use `update-field-level-encryption-config`.

AWS CLI

To update a CloudFront field-level encryption configuration

The following example updates the Comment field of the field-level encryption configuration with the ID C3KM2WVD605UAY by providing the parameters in a JSON file.

To update a field-level encryption configuration, you must have the configuration's ID and ETag. The ID is returned in the output of the create-field-level-encryption-config and list-field-level-encryption-configs commands. To get the ETag, use the get-field-level-encryption or get-field-level-encryption-config command. Use the --if-match option to provide the configuration's ETag.

```
aws cloudfront update-field-level-encryption-config \  
  --id C3KM2WVD605UAY \  
  --if-match E2P4Z4VU7TY5SG \  
  --field-level-encryption-config file://fle-config.json
```

The file fle-config.json is a JSON document in the current directory that contains the following:

```
{  
  "CallerReference": "cli-example",  
  "Comment": "Updated example FLE configuration",  
  "QueryArgProfileConfig": {  
    "ForwardWhenQueryArgProfileIsUnknown": true,  
    "QueryArgProfiles": {  
      "Quantity": 0  
    }  
  },  
  "ContentTypeProfileConfig": {  
    "ForwardWhenContentTypeIsUnknown": true,  
    "ContentTypeProfiles": {  
      "Quantity": 1,  
      "Items": [  
        {  
          "Format": "URLEncoded",  
          "ProfileId": "P280MFCLSY0CVU",  
          "ContentType": "application/x-www-form-urlencoded"  
        }  
      ]  
    }  
  }  
}
```

```
}
}
```

Output:

```
{
  "ETag": "E26M4BIAV81ZF6",
  "FieldLevelEncryption": {
    "Id": "C3KM2WVD605UAY",
    "LastModifiedTime": "2019-12-10T22:26:26.170Z",
    "FieldLevelEncryptionConfig": {
      "CallerReference": "cli-example",
      "Comment": "Updated example FLE configuration",
      "QueryArgProfileConfig": {
        "ForwardWhenQueryArgProfileIsUnknown": true,
        "QueryArgProfiles": {
          "Quantity": 0,
          "Items": []
        }
      },
      "ContentTypeProfileConfig": {
        "ForwardWhenContentTypeIsUnknown": true,
        "ContentTypeProfiles": {
          "Quantity": 1,
          "Items": [
            {
              "Format": "URLEncoded",
              "ProfileId": "P280MFCLSYOCVU",
              "ContentType": "application/x-www-form-urlencoded"
            }
          ]
        }
      }
    }
  }
}
```

- For API details, see [UpdateFieldLevelEncryptionConfig](#) in *AWS CLI Command Reference*.

update-field-level-encryption-profile

The following code example shows how to use `update-field-level-encryption-profile`.

AWS CLI

To update a CloudFront field-level encryption profile

The following example updates the field-level encryption profile with the ID PPK0UOSIF5WSV. This example updates the profile's Name and Comment, and adds a second FieldPatterns item, by providing the parameters in a JSON file.

To update a field-level encryption profile, you must have the profile's ID and ETag. The ID is returned in the output of the create-field-level-encryption-profile and list-field-level-encryption-profiles commands. To get the ETag, use the get-field-level-encryption-profile or get-field-level-encryption-profile-config command. Use the --if-match option to provide the profile's ETag.

```
aws cloudfront update-field-level-encryption-profile \  
  --id PPK0UOSIF5WSV \  
  --if-match E1QQG65FS2L2GC \  
  --field-level-encryption-profile-config file://fle-profile-config.json
```

The file fle-profile-config.json is a JSON document in the current directory that contains the following:

```
{  
  "Name": "ExampleFLEProfileUpdated",  
  "CallerReference": "cli-example",  
  "Comment": "Updated FLE profile for AWS CLI example",  
  "EncryptionEntities": {  
    "Quantity": 1,  
    "Items": [  
      {  
        "PublicKeyId": "K2K8NC4HVFE3M0",  
        "ProviderId": "ExampleFLEProvider",  
        "FieldPatterns": {  
          "Quantity": 2,  
          "Items": [  
            "ExampleSensitiveField",  
            "SecondExampleSensitiveField"  
          ]  
        }  
      }  
    ]  
  }  
}
```



```
}
}
```

Output:

```
{
  "ETag": "EJETYFJ9CL66D",
  "FieldLevelEncryptionProfile": {
    "Id": "PPK0U0SIF5WSV",
    "LastModifiedTime": "2019-12-10T19:05:58.296Z",
    "FieldLevelEncryptionProfileConfig": {
      "Name": "ExampleFLEProfileUpdated",
      "CallerReference": "cli-example",
      "Comment": "Updated FLE profile for AWS CLI example",
      "EncryptionEntities": {
        "Quantity": 1,
        "Items": [
          {
            "PublicKeyId": "K2K8NC4HVFE3M0",
            "ProviderId": "ExampleFLEProvider",
            "FieldPatterns": {
              "Quantity": 2,
              "Items": [
                "ExampleSensitiveField",
                "SecondExampleSensitiveField"
              ]
            }
          }
        ]
      }
    }
  }
}
```

- For API details, see [UpdateFieldLevelEncryptionProfile](#) in *AWS CLI Command Reference*.

Amazon CloudSearch examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon CloudSearch.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

upload-documents

The following code example shows how to use `upload-documents`.

AWS CLI

The following `upload-documents` command uploads a batch of JSON documents to an Amazon CloudSearch domain:

```
aws cloudsearchdomain upload-documents --endpoint-url https://doc-my-domain.us-west-1.cloudsearch.amazonaws.com --content-type application/json --documents document-batch.json
```

Output:

```
{
  "status": "success",
  "adds": 5000,
  "deletes": 0
}
```

- For API details, see [UploadDocuments](#) in *AWS CLI Command Reference*.

CloudTrail examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with CloudTrail.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-tags

The following code example shows how to use add-tags.

AWS CLI

To add tags to trail

The following add-tags command adds tags for Trail1:

```
aws cloudtrail add-tags --resource-id arn:aws:cloudtrail:us-  
east-1:123456789012:trail/Trail1 --tags-list Key=name,Value=Alice  
Key=location,Value=us
```

- For API details, see [AddTags](#) in *AWS CLI Command Reference*.

create-subscription

The following code example shows how to use create-subscription.

AWS CLI

To create and configure AWS resources for a trail

The following `create-subscription` command creates a new S3 bucket and SNS topic for Trail1:

```
aws cloudtrail create-subscription --name Trail1 --s3-new-bucket my-bucket --sns-new-topic my-topic
```

Output:

```
Setting up new S3 bucket my-bucket...
Setting up new SNS topic my-topic...
Creating/updating CloudTrail configuration...
CloudTrail configuration:
{
  "trailList": [
    {
      "IncludeGlobalServiceEvents": true,
      "Name": "Trail1",
      "TrailARN": "arn:aws:cloudtrail:us-east-1:123456789012:trail/Trail1",
      "LogFileValidationEnabled": false,
      "IsMultiRegionTrail": false,
      "S3BucketName": "my-bucket",
      "SnsTopicName": "my-topic",
      "HomeRegion": "us-east-1"
    }
  ],
  "ResponseMetadata": {
    "HTTPStatusCode": 200,
    "RequestId": "f39e51f6-c615-11e5-85bd-d35ca21ee3e2"
  }
}
Starting CloudTrail service...
Logs will be delivered to my-bucket
```

- For API details, see [CreateSubscription](#) in *AWS CLI Command Reference*.

create-trail

The following code example shows how to use `create-trail`.

AWS CLI

To create a trail

The following `create-trail` command creates a multi-region trail named `Trail1` and specifies an S3 bucket:

```
aws cloudtrail create-trail --name Trail1 --s3-bucket-name my-bucket --is-multi-region-trail
```

Output:

```
{
  "IncludeGlobalServiceEvents": true,
  "Name": "Trail1",
  "TrailARN": "arn:aws:cloudtrail:us-west-2:123456789012:trail/Trail1",
  "LogFileValidationEnabled": false,
  "IsMultiRegionTrail": true,
  "S3BucketName": "my-bucket"
}
```

- For API details, see [CreateTrail](#) in *AWS CLI Command Reference*.

delete-trail

The following code example shows how to use `delete-trail`.

AWS CLI

To delete a trail

The following `delete-trail` command deletes a trail named `Trail1`:

```
aws cloudtrail delete-trail --name Trail1
```

- For API details, see [DeleteTrail](#) in *AWS CLI Command Reference*.

describe-trails

The following code example shows how to use `describe-trails`.

AWS CLI

To describe a trail

The following `describe-trails` command returns the settings for `Trail1` and `Trail2`:

```
aws cloudtrail describe-trails --trail-name-list Trail1 Trail2
```

Output:

```
{
  "trailList": [
    {
      "IncludeGlobalServiceEvents": true,
      "Name": "Trail1",
      "TrailARN": "arn:aws:cloudtrail:us-east-1:123456789012:trail/Trail1",
      "LogFileValidationEnabled": false,
      "IsMultiRegionTrail": false,
      "S3BucketName": "my-bucket",
      "CloudWatchLogsRoleArn": "arn:aws:iam::123456789012:role/
CloudTrail_CloudWatchLogs_Role",
      "CloudWatchLogsLogGroupArn": "arn:aws:logs:us-east-1:123456789012:log-
group:CloudTrail:*",
      "SnsTopicName": "my-topic",
      "HomeRegion": "us-east-1"
    },
    {
      "IncludeGlobalServiceEvents": true,
      "Name": "Trail2",
      "S3KeyPrefix": "my-prefix",
      "TrailARN": "arn:aws:cloudtrail:us-east-1:123456789012:trail/Trail2",
      "LogFileValidationEnabled": false,
      "IsMultiRegionTrail": false,
      "S3BucketName": "my-bucket",
      "KmsKeyId": "arn:aws:kms:us-
east-1:123456789012:key/4c5ae5ac-3c13-421e-8335-c7868ef6a769",
      "HomeRegion": "us-east-1"
    }
  ]
}
```

- For API details, see [DescribeTrails](#) in *AWS CLI Command Reference*.

get-event-selectors

The following code example shows how to use `get-event-selectors`.

AWS CLI

To view the event selector settings for a trail

The following `get-event-selectors` command returns the settings for `Trail1`:

```
aws cloudtrail get-event-selectors --trail-name Trail1
```

Output:

```
{
  "EventSelectors": [
    {
      "IncludeManagementEvents": true,
      "DataResources": [],
      "ReadWriteType": "All"
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-1:123456789012:trail/Trail1"
}
```

- For API details, see [GetEventSelectors](#) in *AWS CLI Command Reference*.

get-trail-status

The following code example shows how to use `get-trail-status`.

AWS CLI

To get the status of a trail

The following `get-trail-status` command returns the delivery and logging details for `Trail1`:

```
aws cloudtrail get-trail-status --name Trail1
```

Output:

```
{
  "LatestNotificationTime": 1454022144.869,
  "LatestNotificationAttemptSucceeded": "2016-01-28T23:02:24Z",
  "LatestDeliveryAttemptTime": "2016-01-28T23:02:24Z",
  "LatestDeliveryTime": 1454022144.869,
  "TimeLoggingStarted": "2015-11-06T18:36:38Z",
  "LatestDeliveryAttemptSucceeded": "2016-01-28T23:02:24Z",
  "IsLogging": true,
  "LatestCloudWatchLogsDeliveryTime": 1454022144.918,
  "StartLoggingTime": 1446834998.695,
  "StopLoggingTime": 1446834996.933,
  "LatestNotificationAttemptTime": "2016-01-28T23:02:24Z",
  "TimeLoggingStopped": "2015-11-06T18:36:36Z"
}
```

- For API details, see [GetTrailStatus](#) in *AWS CLI Command Reference*.

list-public-keys

The following code example shows how to use `list-public-keys`.

AWS CLI

To list all public keys for a trail

The following `list-public-keys` command returns all public keys whose private keys were used to sign the digest files within the specified time range:

```
aws cloudtrail list-public-keys --start-time 2016-01-01T20:30:00.000Z
```

Output:

```
{
  "PublicKeyList": [
    {
      "ValidityStartTime": 1453076702.0,
      "ValidityEndTime": 1455668702.0,
      "Value": "MIIBCgKCAQEA1SS3cl92HDycr/MTj0mo0has8habjrxaXw+Kz1WF0axSI2tcF
+3iJ9BKQAVSKxGwxwu3m0wG3J
+kU11xboEcEPHYoIYMbgfSw7KGNUdKwkLzsQWhUJ0cIb0HASox1vv/5fNXkrHhGbDCHeVXm804c83nvHUEFYThr1PfyP
+4WGDk+BGH5m9iuiAKkipEHWmU18/P7XpfpWQuk4h8g3pXZ0rNXr081bh4d39svj7Uqdhv0XoBISp9t/
EXYuePGEtBdrKD9Dz+VHwyUPtBQvYr9BnkF88qBnaPNhS44rzwIDAQAB",
```



```
    "Fingerprint": "7f3f401420072e50a65a141430817ab3"  
  }  
]  
}
```

- For API details, see [ListPublicKeys](#) in *AWS CLI Command Reference*.

list-tags

The following code example shows how to use `list-tags`.

AWS CLI

To list the tags for a trail

The following `list-tags` command lists the tags for `Trail1` and `Trail2`:

```
aws cloudtrail list-tags --resource-id-list arn:aws:cloudtrail:us-  
east-1:123456789012:trail/Trail1 arn:aws:cloudtrail:us-east-1:123456789012:trail/  
Trail2
```

Output:

```
{  
  "ResourceTagList": [  
    {  
      "ResourceId": "arn:aws:cloudtrail:us-east-1:123456789012:trail/Trail1",  
      "TagsList": [  
        {  
          "Value": "Alice",  
          "Key": "name"  
        },  
        {  
          "Value": "us",  
          "Key": "location"  
        }  
      ]  
    },  
    {  
      "ResourceId": "arn:aws:cloudtrail:us-east-1:123456789012:trail/Trail2",  
      "TagsList": [  
        {
```

```

        "Value": "Bob",
        "Key": "name"
    }
]
}

```

- For API details, see [ListTags](#) in *AWS CLI Command Reference*.

lookup-events

The following code example shows how to use lookup-events.

AWS CLI

To look up events for a trail

The following lookup-events command looks up API activity events by the attribute EventName:

```
aws cloudtrail lookup-events --lookup-attributes
AttributeKey=EventName,AttributeValue=ConsoleLogin
```

Output:

```
{
  "Events": [
    {
      "EventId": "654ccbc0-ba0d-486a-9076-dbf7274677a7",
      "Username": "my-session-name",
      "EventTime": "2021-11-18T09:41:02-08:00",
      "CloudTrailEvent": "{\"eventVersion\":\"1.02\",\"userIdentity\":{\"type\":\"AssumedRole\",\"principalId\":\"AR0AJIKPFTA72SWU4L7T4:my-session-name\",\"arn\":\"arn:aws:sts::123456789012:assumed-role/my-role/my-session-name\",\"accountId\":\"123456789012\",\"sessionContext\":{\"attributes\":{\"mfaAuthenticated\":\"false\",\"creationDate\":\"2016-01-26T21:42:12Z\"},\"sessionIssuer\":{\"type\":\"Role\",\"principalId\":\"AR0AJIKPFTA72SWU4L7T4\",\"arn\":\"arn:aws:iam::123456789012:role/my-role\",\"accountId\":\"123456789012\",\"userName\":\"my-role\"}}},\"eventTime\":\"2016-01-26T21:42:12Z\",\"eventSource\":\"signin.amazonaws.com\",\"eventName\":\"ConsoleLogin\",\"awsRegion\":\"us-east-1\",\"sourceIPAddress\":\"72.21.198.70\",\"userAgent\":\"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_5) AppleWebKit/537.36
```

```
(KHTML, like Gecko) Chrome/47.0.2526.111 Safari/537.36\", \"requestParameters
\":null, \"responseElements\": {\"ConsoleLogin\": {\"Success\": true}, \"additionalEventData\":
{\"MobileVersion\": \"No\", \"MFAUsed\": \"No\"}, \"eventID\": \"654ccbc0-ba0d-486a-9076-
dbf7274677a7\", \"eventType\": \"AwsConsoleSignIn\", \"recipientAccountId\":
\"123456789012\"},
  \"EventName\": \"ConsoleLogin\",
  \"Resources\": []
}
]
}
```

- For API details, see [LookupEvents](#) in *AWS CLI Command Reference*.

put-event-selectors

The following code example shows how to use `put-event-selectors`.

AWS CLI

To configure event selectors for a trail

To create an event selector, run the `put-event-selectors` command. When an event occurs in your account, CloudTrail evaluates the configuration for your trails. If the event matches any event selector for a trail, the trail processes and logs the event. You can configure up to 5 event selectors for a trail and up to 250 data resources for a trail.

The following example creates an event selector for a trail named `TrailName` to include read-only and write-only management events, data events for two Amazon S3 bucket/prefix combinations, and data events for a single AWS Lambda function named `hello-world-python-function`:

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-
selectors '[{"ReadWriteType": "All", "IncludeManagementEvents":
true, "DataResources": [{"Type": "AWS::S3::Object", "Values":
["arn:aws:s3:::mybucket/prefix", "arn:aws:s3:::mybucket2/prefix2"]},
{"Type": "AWS::Lambda::Function", "Values": ["arn:aws:lambda:us-
west-2:999999999999:function:hello-world-python-function"]}]]'
```

Output:

```
{
```

```

"EventSelectors": [
  {
    "IncludeManagementEvents": true,
    "DataResources": [
      {
        "Values": [
          "arn:aws:s3::mybucket/prefix",
          "arn:aws:s3::mybucket2/prefix2"
        ],
        "Type": "AWS::S3::Object"
      },
      {
        "Values": [
          "arn:aws:lambda:us-west-2:123456789012:function:hello-world-
python-function"
        ],
        "Type": "AWS::Lambda::Function"
      },
    ],
    "ReadWriteType": "All"
  }
],
"TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}

```

The following example creates an event selector for a trail named "TrailName2" that includes all events, including read-only and write-only management events, and all data events for all Amazon S3 buckets and AWS Lambda functions in the AWS account:

```

aws cloudtrail put-event-selectors --trail-name TrailName2 --event-selectors
' [{"ReadWriteType": "All", "IncludeManagementEvents": true, "DataResources":
[{"Type": "AWS::S3::Object", "Values": ["arn:aws:s3::"]}, {"Type":
"AWS::Lambda::Function", "Values": ["arn:aws:lambda"]}]} ]'

```

Output:

```

{
  "EventSelectors": [
    {
      "IncludeManagementEvents": true,
      "DataResources": [
        {

```

```

        "Values": [
            "arn:aws:s3:::"
        ],
        "Type": "AWS::S3::Object"
    },
    {
        "Values": [
            "arn:aws:lambda"
        ],
        "Type": "AWS::Lambda::Function"
    },
],
"ReadWriteType": "All"
}
],
"TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName2"
}

```

- For API details, see [PutEventSelectors](#) in *AWS CLI Command Reference*.

remove-tags

The following code example shows how to use `remove-tags`.

AWS CLI

To remove tags for a trail

The following `remove-tags` command removes the specified tags for `Trail1`:

```
aws cloudtrail remove-tags --resource-id arn:aws:cloudtrail:us-east-1:123456789012:trail/Trail1 --tags-list Key=name Key=location
```

- For API details, see [RemoveTags](#) in *AWS CLI Command Reference*.

start-logging

The following code example shows how to use `start-logging`.

AWS CLI

To start logging for a trail

The following `start-logging` command turns on logging for `Trail1`:

```
aws cloudtrail start-logging --name Trail1
```

- For API details, see [StartLogging](#) in *AWS CLI Command Reference*.

stop-logging

The following code example shows how to use `stop-logging`.

AWS CLI

To stop logging a trail

The following `stop-logging` command turns off logging for `Trail1`:

```
aws cloudtrail stop-logging --name Trail1
```

- For API details, see [StopLogging](#) in *AWS CLI Command Reference*.

update-subscription

The following code example shows how to use `update-subscription`.

AWS CLI

To update the configuration settings for a trail

The following `update-subscription` command updates the trail to specify a new S3 bucket and SNS topic:

```
aws cloudtrail update-subscription --name Trail1 --s3-new-bucket my-bucket-new --  
sns-new-topic my-topic-new
```

Output:

```
Setting up new S3 bucket my-bucket-new...  
Setting up new SNS topic my-topic-new...  
Creating/Updating CloudTrail configuration...  
CloudTrail configuration:
```

```
{
  "trailList": [
    {
      "IncludeGlobalServiceEvents": true,
      "Name": "Trail1",
      "TrailARN": "arn:aws:cloudtrail:us-east-1:123456789012:trail/Trail1",
      "LogFileValidationEnabled": false,
      "IsMultiRegionTrail": false,
      "S3BucketName": "my-bucket-new",
      "SnsTopicName": "my-topic-new",
      "HomeRegion": "us-east-1"
    }
  ],
  "ResponseMetadata": {
    "HTTPStatusCode": 200,
    "RequestId": "31126f8a-c616-11e5-9cc6-2fd637936879"
  }
}
```

- For API details, see [UpdateSubscription](#) in *AWS CLI Command Reference*.

update-trail

The following code example shows how to use `update-trail`.

AWS CLI

To update a trail

The following `update-trail` command updates a trail to use an existing bucket for log delivery:

```
aws cloudtrail update-trail --name Trail1 --s3-bucket-name my-bucket
```

Output:

```
{
  "IncludeGlobalServiceEvents": true,
  "Name": "Trail1",
  "TrailARN": "arn:aws:cloudtrail:us-west-2:123456789012:trail/Trail1",
  "LogFileValidationEnabled": false,
  "IsMultiRegionTrail": true,
```

```
"S3BucketName": "my-bucket"
}
```

- For API details, see [UpdateTrail](#) in *AWS CLI Command Reference*.

validate-logs

The following code example shows how to use `validate-logs`.

AWS CLI

To validate a log file

The following `validate-logs` command validates the logs for `Trail1`:

```
aws cloudtrail validate-logs --trail-arn arn:aws:cloudtrail:us-east-1:123456789012:trail/Trail1 --start-time 20160129T19:00:00Z
```

Output:

```
Validating log files for trail arn:aws:cloudtrail:us-east-1:123456789012:trail/Trail1 between 2016-01-29T19:00:00Z and 2016-01-29T22:15:43Z
Results requested for 2016-01-29T19:00:00Z to 2016-01-29T22:15:43Z
Results found for 2016-01-29T19:24:57Z to 2016-01-29T21:24:57Z:
3/3 digest files valid
15/15 log files valid
```

- For API details, see [ValidateLogs](#) in *AWS CLI Command Reference*.

CloudWatch examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with CloudWatch.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

delete-alarms

The following code example shows how to use `delete-alarms`.

AWS CLI

To delete an alarm

The following example uses the `delete-alarms` command to delete the Amazon CloudWatch alarm named "myalarm":

```
aws cloudwatch delete-alarms --alarm-names myalarm
```

Output:

```
This command returns to the prompt if successful.
```

- For API details, see [DeleteAlarms](#) in *AWS CLI Command Reference*.

describe-alarm-history

The following code example shows how to use `describe-alarm-history`.

AWS CLI

To retrieve history for an alarm

The following example uses the `describe-alarm-history` command to retrieve history for the Amazon CloudWatch alarm named "myalarm":

```
aws cloudwatch describe-alarm-history --alarm-name "myalarm" --history-item-type  
StateUpdate
```

Output:

```
{
  "AlarmHistoryItems": [
    {
      "Timestamp": "2014-04-09T18:59:06.442Z",
      "HistoryItemType": "StateUpdate",
      "AlarmName": "myalarm",
      "HistoryData": "{\"version\":\"1.0\",\"oldState\":{\"stateValue\":
\\\"ALARM\\\",\\\"stateReason\\\":\\\"testing purposes\\\"},\\\"newState\\\":{\\\"stateValue\\\":\\\"OK
\\\",\\\"stateReason\\\":\\\"Threshold Crossed: 2 datapoints were not greater than the
threshold (70.0). The most recent datapoints: [38.958, 40.292].\\\",\\\"stateReasonData
\\\":{\\\"version\\\":\\\"1.0\\\",\\\"queryDate\\\":\\\"2014-04-09T18:59:06.419+0000\\\",\\\"startDate
\\\":\\\"2014-04-09T18:44:00.000+0000\\\",\\\"statistic\\\":\\\"Average\\\",\\\"period\\\":300,
\\\"recentDatapoints\\\":[38.958,40.292],\\\"threshold\\\":70.0}}}\",
      "HistorySummary": "Alarm updated from ALARM to OK"
    },
    {
      "Timestamp": "2014-04-09T18:59:05.805Z",
      "HistoryItemType": "StateUpdate",
      "AlarmName": "myalarm",
      "HistoryData": "{\"version\":\"1.0\",\"oldState\":{\"stateValue
\\\":\\\"OK\\\",\\\"stateReason\\\":\\\"Threshold Crossed: 2 datapoints were
not greater than the threshold (70.0). The most recent datapoints:
[38.839999999999996, 39.714].\\\",\\\"stateReasonData\\\":{\\\"version\\\":
\\\"1.0\\\",\\\"queryDate\\\":\\\"2014-03-11T22:45:41.569+0000\\\",\\\"startDate\\\":
\\\"2014-03-11T22:30:00.000+0000\\\",\\\"statistic\\\":\\\"Average\\\",\\\"period\\\":300,
\\\"recentDatapoints\\\":[38.839999999999996,39.714],\\\"threshold\\\":70.0}}\",\\\"newState\\\":
{\\\"stateValue\\\":\\\"ALARM\\\",\\\"stateReason\\\":\\\"testing purposes\\\"}}",
      "HistorySummary": "Alarm updated from OK to ALARM"
    }
  ]
}
```

- For API details, see [DescribeAlarmHistory](#) in *AWS CLI Command Reference*.

describe-alarms-for-metric

The following code example shows how to use `describe-alarms-for-metric`.

AWS CLI**To display information about alarms associated with a metric**

The following example uses the `describe-alarms-for-metric` command to display information about any alarms associated with the Amazon EC2 CPUUtilization metric and the instance with the ID `i-0c986c72`:

```
aws cloudwatch describe-alarms-for-metric --metric-name CPUUtilization --namespace
AWS/EC2 --dimensions Name=InstanceId,Value=i-0c986c72
```

Output:

```
{
  "MetricAlarms": [
    {
      "EvaluationPeriods": 10,
      "AlarmArn": "arn:aws:cloudwatch:us-
east-1:111122223333:alarm:myHighCpuAlarm2",
      "StateUpdatedTimestamp": "2013-10-30T03:03:51.479Z",
      "AlarmConfigurationUpdatedTimestamp": "2013-10-30T03:03:50.865Z",
      "ComparisonOperator": "GreaterThanOrEqualToThreshold",
      "AlarmActions": [
        "arn:aws:sns:us-east-1:111122223333:NotifyMe"
      ],
      "Namespace": "AWS/EC2",
      "AlarmDescription": "CPU usage exceeds 70 percent",
      "StateReasonData": "{\"version\":\"1.0\",\"queryDate\":
\"2013-10-30T03:03:51.479+0000\",\"startDate\":\"2013-10-30T02:08:00.000+0000\",
\"statistic\":\"Average\",\"period\":300,\"recentDatapoints\":
[40.698,39.612,42.432,39.796,38.816,42.28,42.854,40.088,40.760000000000005,41.316],
\"threshold\":70.0}",
      "Period": 300,
      "StateValue": "OK",
      "Threshold": 70.0,
      "AlarmName": "myHighCpuAlarm2",
      "Dimensions": [
        {
          "Name": "InstanceId",
          "Value": "i-0c986c72"
        }
      ],
      "Statistic": "Average",
      "StateReason": "Threshold Crossed: 10 datapoints were not greater than
or equal to the threshold (70.0). The most recent datapoints: [40.760000000000005,
41.316].",
      "InsufficientDataActions": [],
    }
  ]
}
```

```

    "OKActions": [],
    "ActionsEnabled": true,
    "MetricName": "CPUUtilization"
  },
  {
    "EvaluationPeriods": 2,
    "AlarmArn": "arn:aws:cloudwatch:us-
east-1:111122223333:alarm:myHighCpuAlarm",
    "StateUpdatedTimestamp": "2014-04-09T18:59:06.442Z",
    "AlarmConfigurationUpdatedTimestamp": "2014-04-09T22:26:05.958Z",
    "ComparisonOperator": "GreaterThanThreshold",
    "AlarmActions": [
      "arn:aws:sns:us-east-1:111122223333:HighCPUAlarm"
    ],
    "Namespace": "AWS/EC2",
    "AlarmDescription": "CPU usage exceeds 70 percent",
    "StateReasonData": "{\"version\":\"1.0\",\"queryDate\":
\\\"2014-04-09T18:59:06.419+0000\\\",\\\"startDate\\\":\\\"2014-04-09T18:44:00.000+0000\\\",
\\\"statistic\\\":\\\"Average\\\",\\\"period\\\":300,\\\"recentDatapoints\\\":[38.958,40.292],
\\\"threshold\\\":70.0}\",
    "Period": 300,
    "StateValue": "OK",
    "Threshold": 70.0,
    "AlarmName": "myHighCpuAlarm",
    "Dimensions": [
      {
        "Name": "InstanceId",
        "Value": "i-0c986c72"
      }
    ],
    "Statistic": "Average",
    "StateReason": "Threshold Crossed: 2 datapoints were not greater than
the threshold (70.0). The most recent datapoints: [38.958, 40.292].",
    "InsufficientDataActions": [],
    "OKActions": [],
    "ActionsEnabled": false,
    "MetricName": "CPUUtilization"
  }
]
}

```

- For API details, see [DescribeAlarmsForMetric](#) in *AWS CLI Command Reference*.

describe-alarms

The following code example shows how to use describe-alarms.

AWS CLI

To list information about an alarm

The following example uses the describe-alarms command to provide information about the alarm named "myalarm":

```
aws cloudwatch describe-alarms --alarm-names "myalarm"
```

Output:

```
{
  "MetricAlarms": [
    {
      "EvaluationPeriods": 2,
      "AlarmArn": "arn:aws:cloudwatch:us-east-1:123456789012:alarm:myalarm",
      "StateUpdatedTimestamp": "2014-04-09T18:59:06.442Z",
      "AlarmConfigurationUpdatedTimestamp": "2012-12-27T00:49:54.032Z",
      "ComparisonOperator": "GreaterThanThreshold",
      "AlarmActions": [
        "arn:aws:sns:us-east-1:123456789012:myHighCpuAlarm"
      ],
      "Namespace": "AWS/EC2",
      "AlarmDescription": "CPU usage exceeds 70 percent",
      "StateReasonData": "{\"version\":\"1.0\",\"queryDate\":\
\\\"2014-04-09T18:59:06.419+0000\\\", \"startDate\": \"2014-04-09T18:44:00.000+0000\\\",
\\\"statistic\\\": \"Average\\\", \"period\": 300, \"recentDatapoints\": [38.958, 40.292],
\\\"threshold\\\": 70.0}\",
      "Period": 300,
      "StateValue": "OK",
      "Threshold": 70.0,
      "AlarmName": "myalarm",
      "Dimensions": [
        {
          "Name": "InstanceId",
          "Value": "i-0c986c72"
        }
      ],
      "Statistic": "Average",
```

```
        "StateReason": "Threshold Crossed: 2 datapoints were not greater than  
the threshold (70.0). The most recent datapoints: [38.958, 40.292].",  
        "InsufficientDataActions": [],  
        "OKActions": [],  
        "ActionsEnabled": true,  
        "MetricName": "CPUUtilization"  
    }  
]  
}
```

- For API details, see [DescribeAlarms](#) in *AWS CLI Command Reference*.

disable-alarm-actions

The following code example shows how to use `disable-alarm-actions`.

AWS CLI

To disable actions for an alarm

The following example uses the `disable-alarm-actions` command to disable all actions for the alarm named `myalarm`:

```
aws cloudwatch disable-alarm-actions --alarm-names myalarm
```

This command returns to the prompt if successful.

- For API details, see [DisableAlarmActions](#) in *AWS CLI Command Reference*.

enable-alarm-actions

The following code example shows how to use `enable-alarm-actions`.

AWS CLI

To enable all actions for an alarm

The following example uses the `enable-alarm-actions` command to enable all actions for the alarm named `myalarm`:

```
aws cloudwatch enable-alarm-actions --alarm-names myalarm
```

This command returns to the prompt if successful.

- For API details, see [EnableAlarmActions](#) in *AWS CLI Command Reference*.

get-metric-statistics

The following code example shows how to use `get-metric-statistics`.

AWS CLI

To get the CPU utilization per EC2 instance

The following example uses the `get-metric-statistics` command to get the CPU utilization for an EC2 instance with the ID `i-abcdef`.

```
aws cloudwatch get-metric-statistics --metric-name CPUUtilization --start-time
2014-04-08T23:18:00Z --end-time 2014-04-09T23:18:00Z --period 3600 --namespace AWS/
EC2 --statistics Maximum --dimensions Name=InstanceId,Value=i-abcdef
```

Output:

```
{
  "Datapoints": [
    {
      "Timestamp": "2014-04-09T11:18:00Z",
      "Maximum": 44.79,
      "Unit": "Percent"
    },
    {
      "Timestamp": "2014-04-09T20:18:00Z",
      "Maximum": 47.92,
      "Unit": "Percent"
    },
    {
      "Timestamp": "2014-04-09T19:18:00Z",
      "Maximum": 50.85,
      "Unit": "Percent"
    },
    {
      "Timestamp": "2014-04-09T09:18:00Z",
      "Maximum": 47.92,
      "Unit": "Percent"
    },
  ],
}
```

```
{
  "Timestamp": "2014-04-09T03:18:00Z",
  "Maximum": 76.84,
  "Unit": "Percent"
},
{
  "Timestamp": "2014-04-09T21:18:00Z",
  "Maximum": 48.96,
  "Unit": "Percent"
},
{
  "Timestamp": "2014-04-09T14:18:00Z",
  "Maximum": 47.92,
  "Unit": "Percent"
},
{
  "Timestamp": "2014-04-09T08:18:00Z",
  "Maximum": 47.92,
  "Unit": "Percent"
},
{
  "Timestamp": "2014-04-09T16:18:00Z",
  "Maximum": 45.55,
  "Unit": "Percent"
},
{
  "Timestamp": "2014-04-09T06:18:00Z",
  "Maximum": 47.92,
  "Unit": "Percent"
},
{
  "Timestamp": "2014-04-09T13:18:00Z",
  "Maximum": 45.08,
  "Unit": "Percent"
},
{
  "Timestamp": "2014-04-09T05:18:00Z",
  "Maximum": 47.92,
  "Unit": "Percent"
},
{
  "Timestamp": "2014-04-09T18:18:00Z",
  "Maximum": 46.88,
  "Unit": "Percent"
}
```



```
  },
  {
    "Timestamp": "2014-04-09T17:18:00Z",
    "Maximum": 52.08,
    "Unit": "Percent"
  },
  {
    "Timestamp": "2014-04-09T07:18:00Z",
    "Maximum": 47.92,
    "Unit": "Percent"
  },
  {
    "Timestamp": "2014-04-09T02:18:00Z",
    "Maximum": 51.23,
    "Unit": "Percent"
  },
  {
    "Timestamp": "2014-04-09T12:18:00Z",
    "Maximum": 47.67,
    "Unit": "Percent"
  },
  {
    "Timestamp": "2014-04-08T23:18:00Z",
    "Maximum": 46.88,
    "Unit": "Percent"
  },
  {
    "Timestamp": "2014-04-09T10:18:00Z",
    "Maximum": 51.91,
    "Unit": "Percent"
  },
  {
    "Timestamp": "2014-04-09T04:18:00Z",
    "Maximum": 47.13,
    "Unit": "Percent"
  },
  {
    "Timestamp": "2014-04-09T15:18:00Z",
    "Maximum": 48.96,
    "Unit": "Percent"
  },
  {
    "Timestamp": "2014-04-09T00:18:00Z",
    "Maximum": 48.16,
```

```

        "Unit": "Percent"
    },
    {
        "Timestamp": "2014-04-09T01:18:00Z",
        "Maximum": 49.18,
        "Unit": "Percent"
    }
],
"Label": "CPUUtilization"
}

```

Specifying multiple dimensions

The following example illustrates how to specify multiple dimensions. Each dimension is specified as a Name/Value pair, with a comma between the name and the value. Multiple dimensions are separated by a space. If a single metric includes multiple dimensions, you must specify a value for every defined dimension.

For more examples using the `get-metric-statistics` command, see *Get Statistics for a Metric* in the *Amazon CloudWatch Developer Guide*.

```

aws cloudwatch get-metric-statistics --metric-name Buffers --namespace MyNameSpace
--dimensions Name=InstanceID,Value=i-abcdef Name=InstanceType,Value=m1.small --
start-time 2016-10-15T04:00:00Z --end-time 2016-10-19T07:00:00Z --statistics Average
--period 60

```

- For API details, see [GetMetricStatistics](#) in *AWS CLI Command Reference*.

list-metrics

The following code example shows how to use `list-metrics`.

AWS CLI

To list the metrics for Amazon SNS

The following `list-metrics` example displays the metrics for Amazon SNS.

```

aws cloudwatch list-metrics \
--namespace "AWS/SNS"

```

Output:

```
{
  "Metrics": [
    {
      "Namespace": "AWS/SNS",
      "Dimensions": [
        {
          "Name": "TopicName",
          "Value": "NotifyMe"
        }
      ],
      "MetricName": "PublishSize"
    },
    {
      "Namespace": "AWS/SNS",
      "Dimensions": [
        {
          "Name": "TopicName",
          "Value": "CF0"
        }
      ],
      "MetricName": "PublishSize"
    },
    {
      "Namespace": "AWS/SNS",
      "Dimensions": [
        {
          "Name": "TopicName",
          "Value": "NotifyMe"
        }
      ],
      "MetricName": "NumberOfNotificationsFailed"
    },
    {
      "Namespace": "AWS/SNS",
      "Dimensions": [
        {
          "Name": "TopicName",
          "Value": "NotifyMe"
        }
      ],
      "MetricName": "NumberOfNotificationsDelivered"
    }
  ],
}
```

```
{
  "Namespace": "AWS/SNS",
  "Dimensions": [
    {
      "Name": "TopicName",
      "Value": "NotifyMe"
    }
  ],
  "MetricName": "NumberOfMessagesPublished"
},
{
  "Namespace": "AWS/SNS",
  "Dimensions": [
    {
      "Name": "TopicName",
      "Value": "CF0"
    }
  ],
  "MetricName": "NumberOfMessagesPublished"
},
{
  "Namespace": "AWS/SNS",
  "Dimensions": [
    {
      "Name": "TopicName",
      "Value": "CF0"
    }
  ],
  "MetricName": "NumberOfNotificationsDelivered"
},
{
  "Namespace": "AWS/SNS",
  "Dimensions": [
    {
      "Name": "TopicName",
      "Value": "CF0"
    }
  ],
  "MetricName": "NumberOfNotificationsFailed"
}
]
```

- For API details, see [ListMetrics](#) in *AWS CLI Command Reference*.

put-metric-alarm

The following code example shows how to use `put-metric-alarm`.

AWS CLI

To send an Amazon Simple Notification Service email message when CPU utilization exceeds 70 percent

The following example uses the `put-metric-alarm` command to send an Amazon Simple Notification Service email message when CPU utilization exceeds 70 percent:

```
aws cloudwatch put-metric-alarm --alarm-name cpu-mon --alarm-description "Alarm
when CPU exceeds 70 percent" --metric-name CPUUtilization --namespace AWS/
EC2 --statistic Average --period 300 --threshold 70 --comparison-operator
GreaterThanThreshold --dimensions "Name=InstanceId,Value=i-12345678" --evaluation-
periods 2 --alarm-actions arn:aws:sns:us-east-1:111122223333:MyTopic --unit Percent
```

This command returns to the prompt if successful. If an alarm with the same name already exists, it will be overwritten by the new alarm.

To specify multiple dimensions

The following example illustrates how to specify multiple dimensions. Each dimension is specified as a Name/Value pair, with a comma between the name and the value. Multiple dimensions are separated by a space:

```
aws cloudwatch put-metric-alarm --alarm-name "Default_Test_Alarm3" --alarm-
description "The default example alarm" --namespace "CW EXAMPLE METRICS" --
metric-name Default_Test --statistic Average --period 60 --evaluation-periods 3
--threshold 50 --comparison-operator GreaterThanOrEqualToThreshold --dimensions
Name=key1,Value=value1 Name=key2,Value=value2
```

- For API details, see [PutMetricAlarm](#) in *AWS CLI Command Reference*.

put-metric-data

The following code example shows how to use `put-metric-data`.

AWS CLI

To publish a custom metric to Amazon CloudWatch

The following example uses the `put-metric-data` command to publish a custom metric to Amazon CloudWatch:

```
aws cloudwatch put-metric-data --namespace "Usage Metrics" --metric-data file://metric.json
```

The values for the metric itself are stored in the JSON file, `metric.json`.

Here are the contents of that file:

```
[
  {
    "MetricName": "New Posts",
    "Timestamp": "Wednesday, June 12, 2013 8:28:20 PM",
    "Value": 0.50,
    "Unit": "Count"
  }
]
```

For more information, see *Publishing Custom Metrics in the Amazon CloudWatch Developer Guide*.

To specify multiple dimensions

The following example illustrates how to specify multiple dimensions. Each dimension is specified as a `Name=Value` pair. Multiple dimensions are separated by a comma:

```
aws cloudwatch put-metric-data --metric-name Buffers --namespace MyNameSpace --unit Bytes --value 231434333 --dimensions InstanceID=1-23456789,InstanceType=m1.small
```

- For API details, see [PutMetricData](#) in *AWS CLI Command Reference*.

set-alarm-state

The following code example shows how to use `set-alarm-state`.

AWS CLI

To temporarily change the state of an alarm

The following example uses the `set-alarm-state` command to temporarily change the state of an Amazon CloudWatch alarm named "myalarm" and set it to the ALARM state for testing purposes:

```
aws cloudwatch set-alarm-state --alarm-name "myalarm" --state-value ALARM --state-reason "testing purposes"
```

This command returns to the prompt if successful.

- For API details, see [SetAlarmState](#) in *AWS CLI Command Reference*.

CloudWatch Logs examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with CloudWatch Logs.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-log-group

The following code example shows how to use `create-log-group`.

AWS CLI

The following command creates a log group named `my-logs`:

```
aws logs create-log-group --log-group-name my-logs
```

- For API details, see [CreateLogGroup](#) in *AWS CLI Command Reference*.

create-log-stream

The following code example shows how to use `create-log-stream`.

AWS CLI

The following command creates a log stream named `20150601` in the log group `my-logs`:

```
aws logs create-log-stream --log-group-name my-logs --log-stream-name 20150601
```

- For API details, see [CreateLogStream](#) in *AWS CLI Command Reference*.

delete-log-group

The following code example shows how to use `delete-log-group`.

AWS CLI

The following command deletes a log group named `my-logs`:

```
aws logs delete-log-group --log-group-name my-logs
```

- For API details, see [DeleteLogGroup](#) in *AWS CLI Command Reference*.

delete-log-stream

The following code example shows how to use `delete-log-stream`.

AWS CLI

The following command deletes a log stream named `20150531` from a log group named `my-logs`:

```
aws logs delete-log-stream --log-group-name my-logs --log-stream-name 20150531
```

- For API details, see [DeleteLogStream](#) in *AWS CLI Command Reference*.

delete-retention-policy

The following code example shows how to use delete-retention-policy.

AWS CLI

The following command removes the retention policy that has previously been applied to a log group named my-logs:

```
aws logs delete-retention-policy --log-group-name my-logs
```

- For API details, see [DeleteRetentionPolicy](#) in *AWS CLI Command Reference*.

describe-log-groups

The following code example shows how to use describe-log-groups.

AWS CLI

The following command describes a log group named my-logs:

```
aws logs describe-log-groups --log-group-name-prefix my-logs
```

Output:

```
{
  "logGroups": [
    {
      "storedBytes": 0,
      "metricFilterCount": 0,
      "creationTime": 1433189500783,
      "logGroupName": "my-logs",
      "retentionInDays": 5,
      "arn": "arn:aws:logs:us-west-2:0123456789012:log-group:my-logs:*"
    }
  ]
}
```

- For API details, see [DescribeLogGroups](#) in *AWS CLI Command Reference*.

describe-log-streams

The following code example shows how to use `describe-log-streams`.

AWS CLI

The following command shows all log streams starting with the prefix `2015` in the log group `my-logs`:

```
aws logs describe-log-streams --log-group-name my-logs --log-stream-name-prefix 2015
```

Output:

```
{
  "logStreams": [
    {
      "creationTime": 1433189871774,
      "arn": "arn:aws:logs:us-west-2:0123456789012:log-group:my-logs:log-stream:20150531",
      "logStreamName": "20150531",
      "storedBytes": 0
    },
    {
      "creationTime": 1433189873898,
      "arn": "arn:aws:logs:us-west-2:0123456789012:log-group:my-logs:log-stream:20150601",
      "logStreamName": "20150601",
      "storedBytes": 0
    }
  ]
}
```

- For API details, see [DescribeLogStreams](#) in *AWS CLI Command Reference*.

get-log-events

The following code example shows how to use `get-log-events`.

AWS CLI

The following command retrieves log events from a log stream named `20150601` in the log group `my-logs`:

```
aws logs get-log-events --log-group-name my-logs --log-stream-name 20150601
```

Output:

```
{
  "nextForwardToken":
  "f/31961209122447488583055879464742346735121166569214640130",
  "events": [
    {
      "ingestionTime": 1433190494190,
      "timestamp": 1433190184356,
      "message": "Example Event 1"
    },
    {
      "ingestionTime": 1433190516679,
      "timestamp": 1433190184356,
      "message": "Example Event 1"
    },
    {
      "ingestionTime": 1433190494190,
      "timestamp": 1433190184358,
      "message": "Example Event 2"
    }
  ],
  "nextBackwardToken":
  "b/31961209122358285602261756944988674324553373268216709120"
}
```

- For API details, see [GetLogEvents](#) in *AWS CLI Command Reference*.

put-log-events

The following code example shows how to use `put-log-events`.

AWS CLI

The following command puts log events to a log stream named `20150601` in the log group `my-logs`:

```
aws logs put-log-events --log-group-name my-logs --log-stream-name 20150601 --log-events file://events
```

Output:

```
{
  "nextSequenceToken": "49542672486831074009579604567656788214806863282469607346"
}
```

The above example reads a JSON array of events from a file named `events` in the current directory:

```
[
  {
    "timestamp": 1433190184356,
    "message": "Example Event 1"
  },
  {
    "timestamp": 1433190184358,
    "message": "Example Event 2"
  },
  {
    "timestamp": 1433190184360,
    "message": "Example Event 3"
  }
]
```

Each subsequent call requires the next sequence token provided by the previous call to be specified with the sequence token option:

```
aws logs put-log-events --log-group-name my-logs --log-stream-
name 20150601 --log-events file://events2 --sequence-token
"49542672486831074009579604567656788214806863282469607346"
```

Output:

```
{
  "nextSequenceToken": "49542672486831074009579604567900991230369019956308219826"
}
```

- For API details, see [PutLogEvents](#) in *AWS CLI Command Reference*.

put-retention-policy

The following code example shows how to use `put-retention-policy`.

AWS CLI

The following command adds a 5 day retention policy to a log group named `my-logs`:

```
aws logs put-retention-policy --log-group-name my-logs --retention-in-days 5
```

- For API details, see [PutRetentionPolicy](#) in *AWS CLI Command Reference*.

CloudWatch Network Monitoring examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with CloudWatch Network Monitoring.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-probe

The following code example shows how to use `create-probe`.

AWS CLI

Example 1: To create a probe that uses TCP and add it to a network monitor

The following `create-probe` example creates a probe that uses the TCP protocol and adds the probe to a monitor named `Example_NetworkMonitor`. Once created, the state of the monitor with the probe will be `PENDING` until the monitor is `ACTIVE`. This might take several minutes, at which point the state will change to `ACTIVE`, and you can start viewing CloudWatch metrics.

```
aws networkmonitor create-probe \  
  --monitor-name Example_NetworkMonitor \  
  --probe sourceArn=arn:aws:ec2:region:111122223333:subnet/subnet-  
id,destination=10.0.0.100,destinationPort=80,protocol=TCP,packetSize=56,tags={Name=Probe1}
```

Output:

```
{  
  "probeId": "probe-12345",  
  "probeArn": "arn:aws:networkmonitor:region:111122223333:probe/probe-12345",  
  "destination": "10.0.0.100",  
  "destinationPort": 80,  
  "packetSize": 56,  
  "addressFamily": "IPV4",  
  "vpcId": "vpc-12345",  
  "state": "PENDING",  
  "createdAt": "2024-03-29T12:41:57.314000-04:00",  
  "modifiedAt": "2024-03-29T12:41:57.314000-04:00",  
  "tags": {  
    "Name": "Probe1"  
  }  
}
```

Example 2: To create a probe that uses probe using ICMP and add it to a network monitor

The following `create-probe` example creates a probe that uses the ICMP protocol and adds the probe to a monitor named `Example_NetworkMonitor`. Once created, the state of the monitor with the probe will be `PENDING` until the monitor is `ACTIVE`. This might take several minutes, at which point the state will change to `ACTIVE`, and you can start viewing CloudWatch metrics.

```
aws networkmonitor create-probe \  
  --monitor-name Example_NetworkMonitor \  
  --probe sourceArn=arn:aws:ec2:region:012345678910:subnet/subnet-  
id,destination=10.0.0.100,protocol=ICMP,packetSize=56,tags={Name=Probe1}
```

Output:

```
{
  "probeId": "probe-12345",
  "probeArn": "arn:aws:networkmonitor:region:111122223333:probe/probe-12345",
  "destination": "10.0.0.100",
  "packetSize": 56,
  "addressFamily": "IPv4",
  "vpcId": "vpc-12345",
  "state": "PENDING",
  "createdAt": "2024-03-29T12:44:02.452000-04:00",
  "modifiedAt": "2024-03-29T12:44:02.452000-04:00",
  "tags": {
    "Name": "Probe1"
  }
}
```

For more information, see [How Amazon CloudWatch Network Monitor Works](#) in the *Amazon CloudWatch User Guide*.

- For API details, see [CreateProbe](#) in *AWS CLI Command Reference*.

delete-monitor

The following code example shows how to use delete-monitor.

AWS CLI**To delete a monitor**

The following delete-monitor example deletes a monitor named Example_NetworkMonitor.

```
aws networkmonitor delete-monitor \
  --monitor-name Example_NetworkMonitor
```

This command produces no output.

For more information, see [How Amazon CloudWatch Network Monitor Works](#) in the *Amazon CloudWatch User Guide*.

- For API details, see [DeleteMonitor](#) in *AWS CLI Command Reference*.

delete-probe

The following code example shows how to use `delete-probe`.

AWS CLI

To delete a probe

The following `delete-probe` example deletes a probe with the ID `probe-12345` from a network monitor named `Example_NetworkMonitor`.

```
aws networkmonitor delete-probe \  
  --monitor-name Example_NetworkMonitor \  
  --probe-id probe-12345
```

This command produces no output.

For more information, see [How Amazon CloudWatch Network Monitor Works](#) in the *Amazon CloudWatch User Guide*.

- For API details, see [DeleteProbe](#) in *AWS CLI Command Reference*.

get-probe

The following code example shows how to use `get-probe`.

AWS CLI

To view probe details

The following `get-probe` example returns details about a probe with the `probeID` `probe-12345` that's associated with a monitor named `Example_NetworkMonitor`.

```
aws networkmonitor get-probe \  
  --monitor-name Example_NetworkMonitor \  
  --probe-id probe-12345
```

Output:

```
{  
  "probeId": "probe-12345",  
  "probeArn": "arn:aws:networkmonitor:region:012345678910:probe/probe-12345",
```



```
"sourceArn": "arn:aws:ec2:region:012345678910:subnet/subnet-12345",
"destination": "10.0.0.100",
"destinationPort": 80,
"protocol": "TCP",
"packetSize": 56,
"addressFamily": "IPV4",
"vpcId": "vpc-12345",
"state": "ACTIVE",
"createdAt": "2024-03-29T12:41:57.314000-04:00",
"modifiedAt": "2024-03-29T12:42:28.610000-04:00",
"tags": {
  "Name": "Probe1"
}
}
```

For more information, see [How Amazon CloudWatch Network Monitor Works](#) in the *Amazon CloudWatch User Guide*.

- For API details, see [GetProbe](#) in *AWS CLI Command Reference*.

list-monitors

The following code example shows how to use `list-monitors`.

AWS CLI

Example 1: To list all monitors (single monitor)

The following `list-monitors` example returns a list of only a single monitor. The monitor's state is `ACTIVE` and it has an `aggregationPeriod` of 60 seconds.

```
aws networkmonitor list-monitors
```

Output:

```
{
  "monitors": [{
    "monitorArn": "arn:aws:networkmonitor:region:012345678910:monitor/
Example_NetworkMonitor",
    "monitorName": "Example_NetworkMonitor",
    "state": "ACTIVE",
    "aggregationPeriod": 60,
```

```
        "tags": {
            "Monitor": "Monitor1"
        }
    ]
}
```

For more information, see [How Amazon CloudWatch Network Monitor Works](#) in the *Amazon CloudWatch User Guide*.

Example 2: To list all monitors (multiple monitors)

The following `list-monitors` example returns a list of three monitors. The state of one monitor is `ACTIVE` and generating CloudWatch metrics. The states of the other two monitors are `INACTIVE` and not generating CloudWatch metrics. All three monitors use an `aggregationPeriod` of 60 seconds.

```
aws networkmonitor list-monitors
```

Output:

```
{
  "monitors": [
    {
      "monitorArn": "arn:aws:networkmonitor:us-east-1:111122223333:monitor/
Example_NetworkMonitor",
      "monitorName": "Example_NetworkMonitor",
      "state": "INACTIVE",
      "aggregationPeriod": 60,
      "tags": {}
    },
    {
      "monitorArn": "arn:aws:networkmonitor:us-east-1:111122223333:monitor/
Example_NetworkMonitor2",
      "monitorName": "Example_NetworkMonitor2",
      "state": "ACTIVE",
      "aggregationPeriod": 60,
      "tags": {
        "Monitor": "Monitor1"
      }
    },
    {
```

```
        "monitorArn": "arn:aws:networkmonitor:us-east-1:111122223333:monitor/
TestNetworkMonitor_CLI",
        "monitorName": "TestNetworkMonitor_CLI",
        "state": "INACTIVE",
        "aggregationPeriod": 60,
        "tags": {}
    }
]
}
```

For more information, see [How Amazon CloudWatch Network Monitor Works](#) in the *Amazon CloudWatch User Guide*.

- For API details, see [ListMonitors](#) in *AWS CLI Command Reference*.

update-monitor

The following code example shows how to use `update-monitor`.

AWS CLI

To update a monitor

The following `update-monitor` example changes a monitor's `aggregationPeriod` from 60 seconds to 30 seconds.

```
aws networkmonitor update-monitor \
  --monitor-name Example_NetworkMonitor \
  --aggregation-period 30
```

Output:

```
{
  "monitorArn": "arn:aws:networkmonitor:region:012345678910:monitor/
Example_NetworkMonitor",
  "monitorName": "Example_NetworkMonitor",
  "state": "PENDING",
  "aggregationPeriod": 30,
  "tags": {
    "Monitor": "Monitor1"
  }
}
```

For more information, see [How Amazon CloudWatch Network Monitor Works](#) in the *Amazon CloudWatch User Guide*.

- For API details, see [UpdateMonitor](#) in *AWS CLI Command Reference*.

update-probe

The following code example shows how to use `update-probe`.

AWS CLI

To update a probe

The following `update-probe` example updates a probe's original destination IP address and also updates the `packetSize` to 60.

```
aws networkmonitor update-probe \  
  --monitor-name Example_NetworkMonitor \  
  --probe-id probe-12345 \  
  --destination 10.0.0.150 \  
  --packet-size 60
```

Output:

```
{  
  "probeId": "probe-12345",  
  "probeArn": "arn:aws:networkmonitor:region:012345678910:probe/probe-12345",  
  "sourceArn": "arn:aws:ec2:region:012345678910:subnet/subnet-12345",  
  "destination": "10.0.0.150",  
  "destinationPort": 80,  
  "protocol": "TCP",  
  "packetSize": 60,  
  "addressFamily": "IPV4",  
  "vpcId": "vpc-12345",  
  "state": "PENDING",  
  "createdAt": "2024-03-29T12:41:57.314000-04:00",  
  "modifiedAt": "2024-03-29T13:52:23.115000-04:00",  
  "tags": {  
    "Name": "Probe1"  
  }  
}
```

For more information, see [How Amazon CloudWatch Network Monitor Works](#) in the *Amazon CloudWatch User Guide*.

- For API details, see [UpdateProbe](#) in *AWS CLI Command Reference*.

CodeArtifact examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with CodeArtifact.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-external-connection

The following code example shows how to use `associate-external-connection`.

AWS CLI

To add an external connection to a repository

The following `associate-external-connection` example adds an external connection to `npmjs.com` to a repository named `test-repo`.

```
aws codeartifact associate-external-connection \  
  --repository test-repo \  
  --domain test-domain \  
  --external-connection public:npmjs
```

Output:

```
{
  "repository": {
    "name": "test-repo",
    "administratorAccount": "111122223333",
    "domainName": "test-domain",
    "domainOwner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/test-domain/test-repo",
    "upstreams": [],
    "externalConnections": [
      {
        "externalConnectionName": "public:npmjs",
        "packageFormat": "npm",
        "status": "AVAILABLE"
      }
    ]
  }
}
```

For more information, see [Add an external connection](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [AssociateExternalConnection](#) in *AWS CLI Command Reference*.

copy-package-versions

The following code example shows how to use `copy-package-versions`.

AWS CLI**To copy package versions from one repository to another**

The following `copy-package-versions` moves versions 4.0.0 and 5.0.0 of a package named `test-package` from `my-repo` to `test-repo`.

```
aws codeartifact copy-package-versions \
  --domain test-domain \
  --source-repository my-repo \
  --destination-repository test-repo \
  --format npm \
  --package test-package \
  --versions '["4.0.0", "5.0.0"]'
```

Output:

```
{
  "format": "npm",
  "package": "test-package",
  "versions": [
    {
      "version": "5.0.0",
      "revision": "REVISION-1-SAMPLE-6C81EFF7DA55CC",
      "status": "Published"
    },
    {
      "version": "4.0.0",
      "revision": "REVISION-2-SAMPLE-55C752BEE772FC",
      "status": "Published"
    }
  ]
}
```

For more information, see [Copy packages between repositories](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [CopyPackageVersions](#) in *AWS CLI Command Reference*.

create-domain

The following code example shows how to use create-domain.

AWS CLI**To create a domain**

The following create-domain example creates a domain named test-domain.

```
aws codeartifact create-domain \
  --domain test-domain
```

Output:

```
{
  "domain": {
    "name": "test-domain",
```

```
    "owner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:domain/test-domain",
    "status": "Active",
    "createdTime": "2020-10-20T13:16:48.559000-04:00",
    "encryptionKey": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111",
    "repositoryCount": 0,
    "assetSizeBytes": 0
  }
}
```

For more information, see [Create a domain](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [CreateDomain](#) in *AWS CLI Command Reference*.

create-repository

The following code example shows how to use create-repository.

AWS CLI

To create a repository

The following create-repository example creates a repository named test-repo inside a domain named test-domain.

```
aws codeartifact create-repository \
  --domain test-domain \
  --domain-owner 111122223333 \
  --repository test-repo \
  --description "This is a test repository."
```

Output:

```
{
  "repository": {
    "name": "test-repo",
    "administratorAccount": "111122223333",
    "domainName": "test-domain",
    "domainOwner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/test-domain/
test-repo",
    "description": "This is a test repository.",
  }
}
```



```
    "upstreams": [],
    "externalConnections": []
  }
}
```

For more information, see [Create a domain](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [CreateRepository](#) in *AWS CLI Command Reference*.

delete-domain-permissions-policy

The following code example shows how to use `delete-domain-permissions-policy`.

AWS CLI

To delete the permissions policy document from a domain

The following `delete-domain-permissions-policy` example deletes the permission policy from a domain named `test-domain`.

```
aws codeartifact delete-domain-permissions-policy \
  --domain test-domain
```

Output:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "BasicDomainPolicy",
      "Action": [
        "codeartifact:GetDomainPermissionsPolicy",
        "codeartifact:ListRepositoriesInDomain",
        "codeartifact:GetAuthorizationToken",
        "codeartifact:CreateRepository"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      }
    }
  ]
}
```

```
}
```

For more information, see [Delete a domain policy](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [DeleteDomainPermissionsPolicy](#) in *AWS CLI Command Reference*.

delete-domain

The following code example shows how to use `delete-domain`.

AWS CLI

To delete a domain

The following `delete-domain` example deletes a domain named `test-domain`.

```
aws codeartifact delete-domain \  
  --domain test-domain
```

Output:

```
{  
  "domain": {  
    "name": "test-domain",  
    "owner": "417498243647",  
    "arn": "arn:aws:codeartifact:us-west-2:417498243647:domain/test-domain",  
    "status": "Deleted",  
    "createdTime": "2020-10-20T13:16:48.559000-04:00",  
    "encryptionKey": "arn:aws:kms:us-west-2:417498243647:key/c9fe2447-0795-4fda-  
afbe-8464574ae162",  
    "repositoryCount": 0,  
    "assetSizeBytes": 0  
  }  
}
```

For more information, see [Delete a domain](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [DeleteDomain](#) in *AWS CLI Command Reference*.

delete-package-versions

The following code example shows how to use `delete-package-versions`.

AWS CLI

To delete package versions

The following `delete-package-versions` example deletes version 4.0.0 of a package named `test-package`.

```
aws codeartifact delete-package-versions \  
  --domain test-domain \  
  --repo test-repo \  
  --format npm \  
  --package test-package \  
  --versions 4.0.0
```

Output:

```
{  
  "successfulVersions": {  
    "4.0.0": {  
      "revision": "Ciqe5/9yicvkJT13b5/LdLpCyE6fqA7poa9qp+FilPs=",  
      "status": "Deleted"  
    }  
  },  
  "failedVersions": {}  
}
```

For more information, see [Delete a package version](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [DeletePackageVersions](#) in *AWS CLI Command Reference*.

delete-repository-permissions-policy

The following code example shows how to use `delete-repository-permissions-policy`.

AWS CLI

To delete a permissions policy from a repository

The following `delete-repository-permissions-policy` example deletes the permission policy from a repository named `test-repo`.

```
aws codeartifact delete-repository-permissions-policy \  
  --domain test-domain \  
  --repo test-repo \  
  --policy test-policy
```

```
--domain test-domain \  
--repository test-repo
```

Output:

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Principal": {  
        "AWS": "arn:aws:iam::111122223333:root"  
      },  
      "Action": [  
        "codeartifact:DescribePackageVersion",  
        "codeartifact:DescribeRepository",  
        "codeartifact:GetPackageVersionReadme",  
        "codeartifact:GetRepositoryEndpoint",  
        "codeartifact:ListPackages",  
        "codeartifact:ListPackageVersions",  
        "codeartifact:ListPackageVersionAssets",  
        "codeartifact:ListPackageVersionDependencies",  
        "codeartifact:ReadFromRepository"  
      ],  
      "Resource": "*"   
    }  
  ]  
}
```

For more information, see [Delete a policy](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [DeleteRepositoryPermissionsPolicy](#) in *AWS CLI Command Reference*.

delete-repository

The following code example shows how to use `delete-repository`.

AWS CLI

To delete a repository

The following `delete-repository` example deletes a repository named `test-repo` in a domain named `test-domain`.

```
aws codeartifact delete-repository \  
  --domain test-domain \  
  --repository test-repo
```

Output:

```
{  
  "repository": {  
    "name": "test-repo",  
    "administratorAccount": "111122223333",  
    "domainName": "test-domain",  
    "domainOwner": "111122223333",  
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/test-domain/  
test-repo",  
    "description": "This is a test repository",  
    "upstreams": [],  
    "externalConnections": []  
  }  
}
```

For more information, see [Delete a repository](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [DeleteRepository](#) in *AWS CLI Command Reference*.

describe-domain

The following code example shows how to use `describe-domain`.

AWS CLI

To get information about a domain

The following `describe-domain` example returns a `DomainDescription` object for a domain named `test-domain`.

```
aws codeartifact describe-domain \  
  --domain test-domain
```

Output:

```
{  
  "domain": {
```

```

    "name": "test-domain",
    "owner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:domain/test-domain",
    "status": "Active",
    "createdTime": "2020-10-20T13:16:48.559000-04:00",
    "encryptionKey": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111",
    "repositoryCount": 2,
    "assetSizeBytes": 0,
    "s3BucketArn": "arn:aws:s3:::assets-111122223333-us-west-2"
  }
}

```

For more information, see [Domain overview](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [DescribeDomain](#) in *AWS CLI Command Reference*.

describe-repository

The following code example shows how to use `describe-repository`.

AWS CLI

To get information about a repository

The following `describe-repository` example returns a `RepositoryDescription` object for a repository named `test-repo`.

```

aws codeartifact describe-repository \
  --domain test-domain \
  --repository test-repo

```

Output:

```

{
  "repository": {
    "name": "test-repo",
    "administratorAccount": "111122223333",
    "domainName": "test-domain",
    "domainOwner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/test-domain/
test-repo",
    "description": "This is a test repository.",
  }
}

```

```
    "upstreams": [],
    "externalConnections": []
  }
}
```

For more information, see [Create a domain](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [DescribeRepository](#) in *AWS CLI Command Reference*.

disassociate-external-connection

The following code example shows how to use `disassociate-external-connection`.

AWS CLI

To remove an external connection from a repository

The following `disassociate-external-connection` example removes an external connection to `npmjs.com` from a repository named `test-repo`.

```
aws codeartifact disassociate-external-connection \
  --repository test-repo \
  --domain test-domain \
  --external-connection public:npmjs
```

Output:

```
{
  "repository": {
    "name": "test-repo",
    "administratorAccount": "111122223333",
    "domainName": "test-domain",
    "domainOwner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/test-domain/test-repo",
    "upstreams": [],
    "externalConnections": []
  }
}
```

For more information, see [Remove an external connection](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [DisassociateExternalConnection](#) in *AWS CLI Command Reference*.

dispose-package-versions

The following code example shows how to use `dispose-package-versions`.

AWS CLI

To delete a package version's assets and set its status to Disposed

The following `dispose-package-versions` example deletes the assets of `test-package` version `4.0.0` and sets its status to `Disposed`.

```
aws codeartifact dispose-package-versions \  
  --domain test-domain \  
  --repo test-repo \  
  --format npm \  
  --package test-package \  
  --versions 4.0.0
```

Output:

```
{  
  "successfulVersions": {  
    "4.0.0": {  
      "revision": "Ciqe5/9yicvkJT13b5/LdLpCyE6fqA7poa9qp+FilPs=",  
      "status": "Disposed"  
    }  
  },  
  "failedVersions": {}  
}
```

For more information, see [Working with packages in CodeArtifact](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [DisposePackageVersions](#) in *AWS CLI Command Reference*.

get-authorization-token

The following code example shows how to use `get-authorization-token`.

AWS CLI

To get an authorization token

The following `get-authorization-token` example retrieves a CodeArtifact authorization token.

```
aws codeartifact get-authorization-token \  
  --domain test-domain \  
  --query authorizationToken \  
  --output text
```

Output:

This command will return the authorization token. You can store the output in an environment variable when calling the command.

For more information, see [Configure pip without the login command](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [GetAuthorizationToken](#) in *AWS CLI Command Reference*.

get-domain-permissions-policy

The following code example shows how to use `get-domain-permissions-policy`.

AWS CLI

To get the permissions policy document for a domain

The following `get-domain-permissions-policy` example gets the permission policy attached to a domain named `test-domain`.

```
aws codeartifact get-domain-permissions-policy \  
  --domain test-domain
```

Output:

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "BasicDomainPolicy",  
      "Action": [  
        "codeartifact:GetDomainPermissionsPolicy",
```

```

        "codeartifact:ListRepositoriesInDomain",
        "codeartifact:GetAuthorizationToken",
        "codeartifact:CreateRepository"
    ],
    "Effect": "Allow",
    "Resource": "*",
    "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
    }
}
]
}

```

For more information, see [Read a domain policy](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [GetDomainPermissionsPolicy](#) in *AWS CLI Command Reference*.

get-package-version-asset

The following code example shows how to use `get-package-version-asset`.

AWS CLI

To get an asset from a package version

The following `get-package-version-asset` example retrieves the `package.tgz` asset for version 4.0.0 of an npm package named `test-package`.

```

aws codeartifact get-package-version-asset \
  --domain test-domain \
  --repository test-repo \
  --format npm \
  --package test-package \
  --package-version 4.0.0 \
  --asset 'package.tgz' \
  outfileName

```

Output:

The output for this command will also store the raw asset in the file provided in place of `outfileName`.

```
{
  "assetName": "package.tgz",
  "packageVersion": "4.0.0",
  "packageVersionRevision": "Ciqe5/9yicvkJT13b5/LdLpCyE6fqA7poa9qp+FilPs="
}
```

For more information, see [List package version assets](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [GetPackageVersionAsset](#) in *AWS CLI Command Reference*.

get-package-version-readme

The following code example shows how to use `get-package-version-readme`.

AWS CLI

To get a package version's readme file

The following `get-package-version-readme` example retrieves the readme file for version 4.0.0 of an npm package named `test-package`.

```
aws codeartifact get-package-version-readme \
  --domain test-domain \
  --repo test-repo \
  --format npm \
  --package test-package \
  --package-version 4.0.0
```

Output:

```
{
  "format": "npm",
  "package": "test-package",
  "version": "4.0.0",
  "readme": "<div align=\"center\">\n  <a href=\"https://github.com/test-package/testpack\"> ... more content ... \n",
  "versionRevision": "Ciqe5/9yicvkJT13b5/LdLpCyE6fqA7poa9qp+FilPs="
}
```

For more information, see [View package version readme file](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [GetPackageVersionReadme](#) in *AWS CLI Command Reference*.

get-repository-endpoint

The following code example shows how to use `get-repository-endpoint`.

AWS CLI

To get a repository's URL endpoint

The following `get-repository-endpoint` example returns the npm endpoint for the `test-repo` repository.

```
aws codeartifact get-repository-endpoint \  
  --domain test-domain \  
  --repository test-repo \  
  --format npm
```

Output:

```
{  
  "repositoryEndpoint": "https://test-domain-111122223333.d.codeartifact.us-  
west-2.amazonaws.com/npm/test-repo/"  
}
```

For more information, see [Connect to a repository](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [GetRepositoryEndpoint](#) in *AWS CLI Command Reference*.

get-repository-permissions-policy

The following code example shows how to use `get-repository-permissions-policy`.

AWS CLI

To get the permissions policy document for a repository

The following `get-repository-permissions-policy` example gets the permission policy attached to a repository named `test-repo`.

```
aws codeartifact get-repository-permissions-policy \  
  --domain test-domain \  
  --repository test-repo
```

Output:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": [
        "codeartifact:DescribePackageVersion",
        "codeartifact:DescribeRepository",
        "codeartifact:GetPackageVersionReadme",
        "codeartifact:GetRepositoryEndpoint",
        "codeartifact:ListPackages",
        "codeartifact:ListPackageVersions",
        "codeartifact:ListPackageVersionAssets",
        "codeartifact:ListPackageVersionDependencies",
        "codeartifact:ReadFromRepository"
      ],
      "Resource": "*"
    }
  ]
}
```

For more information, see [Read a policy](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [GetRepositoryPermissionsPolicy](#) in *AWS CLI Command Reference*.

list-domains

The following code example shows how to use `list-domains`.

AWS CLI**To list domains**

The following `list-domains` example returns a summary of all domains owned by the AWS account that makes the call.

```
aws codeartifact list-domains
```

Output:

```
{
  "domains": [
    {
      "name": "my-domain",
      "owner": "111122223333",
      "status": "Active",
      "encryptionKey": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    },
    {
      "name": "test-domain",
      "owner": "111122223333",
      "status": "Active",
      "encryptionKey": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
    }
  ]
}
```

For more information, see [Working with domains in CodeArtifact](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [ListDomains](#) in *AWS CLI Command Reference*.

list-package-version-assets

The following code example shows how to use `list-package-version-assets`.

AWS CLI**To view a package version's assets**

The following `list-package-version-assets` example retrieves the assets for version 4.0.0 of an npm package named `test-package`.

```
aws codeartifact list-package-version-assets \
  --domain test-domain \
  --repo test-repo \
  --format npm \
  --package test-package \
```

```
--package-version 4.0.0
```

Output:

```
{
  "format": "npm",
  "package": "test-package",
  "version": "4.0.0",
  "versionRevision": "Ciqe5/9yicvkJT13b5/LdLpCyE6fqA7poa9qp+FilPs=",
  "assets": [
    {
      "name": "package.tgz",
      "size": 316680,
      "hashes": {
        "MD5": "60078ec6d9e76b89fb55c860832742b2",
        "SHA-1": "b44a9b6297bcb698f1c51a3545a2b3b368d59c52",
        "SHA-256":
          "d2aa8c6afc3c8591765785a37d1c5acae482a8eb3ab9729ed28922692454f2e2",
        "SHA-512":
          "3e585d15c8a594e20d7de57b362ea81754c011acb2641a19f1b72c8531ea39825896bab344ae616a0a5a824cb9
      }
    }
  ]
}
```

For more information, see [List package version assets](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [ListPackageVersionAssets](#) in *AWS CLI Command Reference*.

list-package-version-dependencies

The following code example shows how to use `list-package-version-dependencies`.

AWS CLI

To view a package version's dependencies

The following `list-package-version-dependencies` example retrieves the dependencies for version 4.0.0 of an npm package named `test-package`.

```
aws codeartifact list-package-version-dependencies \
  --domain test-domain \
  --repo test-repo \
```

```
--format npm \  
--package test-package \  
--package-version 4.0.0
```

Output:

```
{  
  "format": "npm",  
  "package": "test-package",  
  "version": "4.0.0",  
  "versionRevision": "Ciqe5/9yicvkJT13b5/LdLpCyE6fqA7poa9qp+FilPs=",  
  "dependencies": [  
    {  
      "namespace": "testns",  
      "package": "testdep1",  
      "dependencyType": "regular",  
      "versionRequirement": "1.8.5"  
    },  
    {  
      "namespace": "testns",  
      "package": "testdep2",  
      "dependencyType": "regular",  
      "versionRequirement": "1.8.5"  
    }  
  ]  
}
```

For more information, see [View and update package version details and dependencies](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [ListPackageVersionDependencies](#) in *AWS CLI Command Reference*.

list-package-versions

The following code example shows how to use `list-package-versions`.

AWS CLI

To list package versions for a package

The following `list-package-versions` example returns a list of package versions for a package named `kind-of`.


```
aws codeartifact list-package-versions \  
  --package kind-of \  
  --domain test-domain \  
  --repository test-repo \  
  --format npm
```

Output:

```
{  
  "defaultDisplayVersion": "1.0.1",  
  "format": "npm",  
  "package": "kind-of",  
  "versions": [  
    {  
      "version": "1.0.1",  
      "revision": "REVISION-SAMPLE-1-C7F4S5E9B772FC",  
      "status": "Published"  
    },  
    {  
      "version": "1.0.0",  
      "revision": "REVISION-SAMPLE-2-C752BEEF6D2CFC",  
      "status": "Published"  
    },  
    {  
      "version": "0.1.2",  
      "revision": "REVISION-SAMPLE-3-654S65A5C5E1FC",  
      "status": "Published"  
    },  
    {  
      "version": "0.1.1",  
      "revision": "REVISION-SAMPLE-1-C7F4S5E9B772FC",  
      "status": "Published"  
    },  
    {  
      "version": "0.1.0",  
      "revision": "REVISION-SAMPLE-4-AF669139B772FC",  
      "status": "Published"  
    }  
  ]  
}
```

For more information, see [List package versions](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [ListPackageVersions](#) in *AWS CLI Command Reference*.

list-packages

The following code example shows how to use `list-packages`.

AWS CLI

To list packages in a repository

The following `list-packages` example list packages in a repository named `test-repo` in a domain named `test-domain`.

```
aws codeartifact list-packages \  
  --domain test-domain \  
  --repository test-repo
```

Output:

```
{  
  "packages": [  
    {  
      "format": "npm",  
      "package": "lodash"  
    },  
    {  
      "format": "python",  
      "package": "test-package"  
    }  
  ]  
}
```

For more information, see [List package names](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [ListPackages](#) in *AWS CLI Command Reference*.

list-repositories-in-domain

The following code example shows how to use `list-repositories-in-domain`.

AWS CLI

To list repositories in a domain

The following `list-repositories-in-domain` example returns a summary of all repositories in the `test-domain` domain.

```
aws codeartifact list-repositories-in-domain \
  --domain test-domain
```

Output:

```
{
  "repositories": [
    {
      "name": "test-repo",
      "administratorAccount": "111122223333",
      "domainName": "test-domain",
      "domainOwner": "111122223333",
      "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/test-domain/test-repo",
      "description": "This is a test repository."
    },
    {
      "name": "test-repo2",
      "administratorAccount": "111122223333",
      "domainName": "test-domain",
      "domainOwner": "111122223333",
      "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/test-domain/test-repo2",
      "description": "This is a test repository."
    }
  ]
}
```

For more information, see [List repositories](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [ListRepositoriesInDomain](#) in *AWS CLI Command Reference*.

list-repositories

The following code example shows how to use `list-repositories`.

AWS CLI

To list repositories

The following `list-repositories` example returns a summary of all repositories in domain owned by the AWS account that makes the call.

```
aws codeartifact list-repositories
```

Output:

```
{
  "repositories": [
    {
      "name": "npm-store",
      "administratorAccount": "111122223333",
      "domainName": "my-domain",
      "domainOwner": "111122223333",
      "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/my-domain/npm-store",
      "description": "Provides npm artifacts from npm, Inc."
    },
    {
      "name": "target-repo",
      "administratorAccount": "111122223333",
      "domainName": "my-domain",
      "domainOwner": "111122223333",
      "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/my-domain/target-repo",
      "description": "test target repo"
    },
    {
      "name": "test-repo2",
      "administratorAccount": "111122223333",
      "domainName": "test-domain",
      "domainOwner": "111122223333",
      "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/test-domain/test-repo2",
      "description": "This is a test repository."
    }
  ]
}
```

For more information, see [List repositories](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [ListRepositories](#) in *AWS CLI Command Reference*.

login

The following code example shows how to use `login`.

AWS CLI

To configure authentication to your repository with the `login` command

The following `login` example configures the `npm` package manager with a repository named `test-repo` in a domain named `test-domain`.

```
aws codeartifact login \  
  --domain test-domain \  
  --repository test-repo \  
  --tool npm
```

Output:

```
Successfully configured npm to use AWS CodeArtifact repository https://test-  
domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/npm/test-repo/  
Login expires in 12 hours at 2020-11-12 01:53:16-05:00
```

For more information, see [Getting started with the AWS CLI](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [Login](#) in *AWS CLI Command Reference*.

put-domain-permissions-policy

The following code example shows how to use `put-domain-permissions-policy`.

AWS CLI

To attach a permissions policy to a domain

The following `put-domain-permissions-policy` example attaches a permission policy that is defined in the `policy.json` file to a domain named `test-domain`.

```
aws codeartifact put-domain-permissions-policy \  
  --domain test-domain \  
  --policy-name test-policy \  
  --policy-file policy.json
```

```
--domain test-domain \  
--policy-document file://PATH/T0/policy.json
```

Output:

```
{  
  "policy": {  
    "resourceArn": "arn:aws:codeartifact:region-id:111122223333:domain/test-  
domain",  
    "document": "{ ...policy document content...}",  
    "revision": "MQ1yyTQRASRU3HB58gBtSDHXG7Q3hvxxxxxxxxx="
```

For more information, see [Set a domain policy](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [PutDomainPermissionsPolicy](#) in *AWS CLI Command Reference*.

put-repository-permissions-policy

The following code example shows how to use `put-repository-permissions-policy`.

AWS CLI

To attach a permissions policy to a repository

The following `put-repository-permissions-policy` example attaches a permission policy that is defined in the `policy.json` file to a repository named `test-repo`.

```
aws codeartifact put-repository-permissions-policy \  
  --domain test-domain \  
  --repository test-repo \  
  --policy-document file://PATH/T0/policy.json
```

Output:

```
{  
  "policy": {  
    "resourceArn": "arn:aws:codeartifact:region-id:111122223333:repository/test-  
domain/test-repo",  
    "document": "{ ...policy document content...}",
```

```

    "revision": "MQ1yyTQRASRU3HB58gBtSDHXG7Q3hvxxxxxxxx="
  }
}

```

For more information, see [Set a policy](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [PutRepositoryPermissionsPolicy](#) in *AWS CLI Command Reference*.

update-package-versions-status

The following code example shows how to use `update-package-versions-status`.

AWS CLI

To update package version status

The following `update-package-versions-status` example updates the status of version 4.0.0 of the `test-package` package to `Archived`.

```

aws codeartifact update-package-versions-status \
  --domain test-domain \
  --repo test-repo \
  --format npm \
  --package test-package \
  --versions 4.0.0 \
  --target-status Archived

```

Output:

```

{
  "successfulVersions": {
    "4.0.0": {
      "revision": "Ciqe5/9yicvkJT13b5/LdLpCyE6fqA7poa9qp+FilPs=",
      "status": "Archived"
    }
  },
  "failedVersions": {}
}

```

For more information, see [Update package version status](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [UpdatePackageVersionsStatus](#) in *AWS CLI Command Reference*.

update-repository

The following code example shows how to use `update-repository`.

AWS CLI

To update a repository

The following `update-repository` example updates the description of a repo named `test-repo` in a domain named `test-domain` to "this is an updated description".

```
aws codeartifact update-repository \  
  --domain test-domain \  
  --repository test-repo \  
  --description "this is an updated description"
```

Output:

```
{  
  "repository": {  
    "name": "test-repo",  
    "administratorAccount": "111122223333",  
    "domainName": "test-domain",  
    "domainOwner": "111122223333",  
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/test-domain/  
test-repo",  
    "description": "this is an updated description",  
    "upstreams": [],  
    "externalConnections": []  
  }  
}
```

For more information, see [View or modify a repository configuration](#) in the *AWS CodeArtifact User Guide*.

- For API details, see [UpdateRepository](#) in *AWS CLI Command Reference*.

CodeBuild examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with CodeBuild.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

batch-delete-builds

The following code example shows how to use `batch-delete-builds`.

AWS CLI

To delete builds in AWS CodeBuild.

The following `batch-delete-builds` example deletes builds in CodeBuild with the specified IDs.

```
aws codebuild batch-delete-builds --ids my-build-project-one:a1b2c3d4-5678-9012-
abcd-11111EXAMPLE my-build-project-two:a1b2c3d4-5678-9012-abcd-22222EXAMPLE
```

Output:

```
{
  "buildsNotDeleted": [
    {
      "id": "arn:aws:codebuild:us-west-2:123456789012:build/my-build-project-
one:a1b2c3d4-5678-9012-abcd-11111EXAMPLE",
      "statusCode": "BUILD_IN_PROGRESS"
    }
  ],
  "buildsDeleted": [
    "arn:aws:codebuild:us-west-2:123456789012:build/my-build-project-
two:a1b2c3d4-5678-9012-abcd-22222EXAMPLE"
```

```
]
}
```

For more information, see [Delete Builds \(AWS CLI\)](#) in the *AWS CodeBuild User Guide*.

- For API details, see [BatchDeleteBuilds](#) in *AWS CLI Command Reference*.

batch-get-build-batches

The following code example shows how to use `batch-get-build-batches`.

AWS CLI

To view details of builds in AWS CodeBuild.

The following `batch-get-build-batches` example gets information about build batches in CodeBuild with the specified IDs.

```
aws codebuild batch-get-build-batches \
  --ids codebuild-demo-project:e9c4f4df-3f43-41d2-ab3a-60fe2EXAMPLE
```

Output:

```
{
  "buildBatches": [
    {
      "id": "codebuild-demo-project:e9c4f4df-3f43-41d2-ab3a-60fe2EXAMPLE",
      "arn": "arn:aws:codebuild:us-west-2:123456789012:build-batch/codebuild-
demo-project:e9c4f4df-3f43-41d2-ab3a-60fe2EXAMPLE",
      "startTime": "2020-11-03T21:52:20.775000+00:00",
      "endTime": "2020-11-03T21:56:59.784000+00:00",
      "currentPhase": "SUCCEEDED",
      "buildBatchStatus": "SUCCEEDED",
      "resolvedSourceVersion": "0a6546f68309560d08a310daac92314c4d378f6b",
      "projectName": "codebuild-demo-project",
      "phases": [
        {
          "phaseType": "SUBMITTED",
          "phaseStatus": "SUCCEEDED",
          "startTime": "2020-11-03T21:52:20.775000+00:00",
          "endTime": "2020-11-03T21:52:20.976000+00:00",
          "durationInSeconds": 0
        }
      ],
    }
  ],
}
```

```
{
  "phaseType": "DOWNLOAD_BATCHSPEC",
  "phaseStatus": "SUCCEEDED",
  "startTime": "2020-11-03T21:52:20.976000+00:00",
  "endTime": "2020-11-03T21:52:57.401000+00:00",
  "durationInSeconds": 36
},
{
  "phaseType": "IN_PROGRESS",
  "phaseStatus": "SUCCEEDED",
  "startTime": "2020-11-03T21:52:57.401000+00:00",
  "endTime": "2020-11-03T21:56:59.751000+00:00",
  "durationInSeconds": 242
},
{
  "phaseType": "COMBINE_ARTIFACTS",
  "phaseStatus": "SUCCEEDED",
  "startTime": "2020-11-03T21:56:59.751000+00:00",
  "endTime": "2020-11-03T21:56:59.784000+00:00",
  "durationInSeconds": 0
},
{
  "phaseType": "SUCCEEDED",
  "startTime": "2020-11-03T21:56:59.784000+00:00"
}
],
"source": {
  "type": "GITHUB",
  "location": "https://github.com/my-repo/codebuild-demo-project.git",
  "gitCloneDepth": 1,
  "gitSubmodulesConfig": {
    "fetchSubmodules": false
  },
  "reportBuildStatus": false,
  "insecureSsl": false
},
"secondarySources": [],
"secondarySourceVersions": [],
"artifacts": {
  "location": ""
},
"secondaryArtifacts": [],
"cache": {
  "type": "NO_CACHE"
}
```

```
  },
  "environment": {
    "type": "LINUX_CONTAINER",
    "image": "aws/codebuild/amazonlinux2-x86_64-standard:3.0",
    "computeType": "BUILD_GENERAL1_SMALL",
    "environmentVariables": [],
    "privilegedMode": false,
    "imagePullCredentialsType": "CODEBUILD"
  },
  "logConfig": {
    "cloudWatchLogs": {
      "status": "ENABLED"
    },
    "s3Logs": {
      "status": "DISABLED",
      "encryptionDisabled": false
    }
  },
  "buildTimeoutInMinutes": 60,
  "queuedTimeoutInMinutes": 480,
  "complete": true,
  "initiator": "Strohm",
  "encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3",
  "buildBatchNumber": 6,
  "buildBatchConfig": {
    "serviceRole": "arn:aws:iam::123456789012:role/service-role/codebuild-demo-project",
    "restrictions": {
      "maximumBuildsAllowed": 100
    }
  },
  "timeoutInMins": 480
},
"buildGroups": [
  {
    "identifier": "DOWNLOAD_SOURCE",
    "ignoreFailure": false,
    "currentBuildSummary": {
      "arn": "arn:aws:codebuild:us-west-2:123456789012:build/codebuild-demo-project:379737d8-bc35-48ec-97fd-776d27545315",
      "requestedOn": "2020-11-03T21:52:21.394000+00:00",
      "buildStatus": "SUCCEEDED",
      "primaryArtifact": {
        "type": "no_artifacts",
        "identifier": "DOWNLOAD_SOURCE"
      }
    }
  }
]
```

```
    },
    "secondaryArtifacts": []
  }
},
{
  "identifier": "linux_small",
  "dependsOn": [],
  "ignoreFailure": false,
  "currentBuildSummary": {
    "arn": "arn:aws:codebuild:us-west-2:123456789012:build/
codebuild-demo-project:dd785171-ed84-4bb6-8ede-ceeb86e54bdb",
    "requestedOn": "2020-11-03T21:52:57.604000+00:00",
    "buildStatus": "SUCCEEDED",
    "primaryArtifact": {
      "type": "no_artifacts",
      "identifier": "linux_small"
    },
    "secondaryArtifacts": []
  },
},
{
  "identifier": "linux_medium",
  "dependsOn": [
    "linux_small"
  ],
  "ignoreFailure": false,
  "currentBuildSummary": {
    "arn": "arn:aws:codebuild:us-west-2:123456789012:build/
codebuild-demo-project:97cf7bd4-5313-4786-8243-4aef350a1267",
    "requestedOn": "2020-11-03T21:54:18.474000+00:00",
    "buildStatus": "SUCCEEDED",
    "primaryArtifact": {
      "type": "no_artifacts",
      "identifier": "linux_medium"
    },
    "secondaryArtifacts": []
  },
},
{
  "identifier": "linux_large",
  "dependsOn": [
    "linux_medium"
  ],
  "ignoreFailure": false,
```

```

        "currentBuildSummary": {
            "arn": "arn:aws:codebuild:us-west-2:123456789012:build/
codebuild-demo-project:60a194cd-0d03-4337-9db1-d41476a17d27",
            "requestedOn": "2020-11-03T21:55:39.203000+00:00",
            "buildStatus": "SUCCEEDED",
            "primaryArtifact": {
                "type": "no_artifacts",
                "identifier": "linux_large"
            },
            "secondaryArtifacts": []
        }
    ]
},
"buildBatchesNotFound": []
}

```

For more information, see Batch builds in AWS CodeBuild <<https://docs.aws.amazon.com/codebuild/latest/userguide/batch-build.html>>__ in the *AWS CodeBuild User Guide*.

- For API details, see [BatchGetBuildBatches](#) in *AWS CLI Command Reference*.

batch-get-builds

The following code example shows how to use batch-get-builds.

AWS CLI

To view details of builds in AWS CodeBuild.

The following batch-get-builds example gets information about builds in CodeBuild with the specified IDs.

```
aws codebuild batch-get-builds --ids codebuild-demo-project:e9c4f4df-3f43-41d2-
ab3a-60fe2EXAMPLE codebuild-demo-project:815e755f-bade-4a7e-80f0-efe51EXAMPLE
```

Output:

```
{
  "buildsNotFound": [],
  "builds": [
    {
```

```
"artifacts": {
  "md5sum": "0e95edf915048a0c22efe6d139fff837",
  "location": "arn:aws:s3:::codepipeline-us-west-2-820783811474/CodeBuild-Python-Pip/BuildArtif/6DJsqQa",
  "encryptionDisabled": false,
  "sha256sum":
"cfa0df33a090966a737f64ae4fe498969fdc842a0c9aec540bf93c37ac0d05a2"
},
"logs": {
  "cloudWatchLogs": {
    "status": "ENABLED"
  },
  "s3Logs": {
    "status": "DISABLED"
  },
  "streamName": "46472baf-8f6b-43c2-9255-b3b963af2732",
  "groupName": "/aws/codebuild/codebuild-demo-project",
  "deepLink": "https://console.aws.amazon.com/cloudwatch/home?region=us-west-2#logEvent:group=/aws/codebuild/codebuild-demo-project;stream=46472baf-8f6b-43c2-9255-b3b963af2732"
},
"timeoutInMinutes": 60,
"environment": {
  "privilegedMode": false,
  "computeType": "BUILD_GENERAL1_MEDIUM",
  "image": "aws/codebuild/windows-base:1.0",
  "environmentVariables": [],
  "type": "WINDOWS_CONTAINER"
},
"projectName": "codebuild-demo-project",
"buildComplete": true,
"source": {
  "gitCloneDepth": 1,
  "insecureSsl": false,
  "type": "CODEPIPELINE"
},
"buildStatus": "SUCCEEDED",
"secondaryArtifacts": [],
"phases": [
  {
    "durationInSeconds": 0,
    "startTime": 1548717462.122,
    "phaseType": "SUBMITTED",
    "endTime": 1548717462.484,
```

```
    "phaseStatus": "SUCCEEDED"
  },
  {
    "durationInSeconds": 0,
    "startTime": 1548717462.484,
    "phaseType": "QUEUED",
    "endTime": 1548717462.775,
    "phaseStatus": "SUCCEEDED"
  },
  {
    "durationInSeconds": 34,
    "endTime": 1548717496.909,
    "contexts": [
      {
        "statusCode": "",
        "message": ""
      }
    ],
    "startTime": 1548717462.775,
    "phaseType": "PROVISIONING",
    "phaseStatus": "SUCCEEDED"
  },
  {
    "durationInSeconds": 15,
    "endTime": 1548717512.555,
    "contexts": [
      {
        "statusCode": "",
        "message": ""
      }
    ],
    "startTime": 1548717496.909,
    "phaseType": "DOWNLOAD_SOURCE",
    "phaseStatus": "SUCCEEDED"
  },
  {
    "durationInSeconds": 0,
    "endTime": 1548717512.734,
    "contexts": [
      {
        "statusCode": "",
        "message": ""
      }
    ]
  },
],
```



```
    "startTime": 1548717512.555,  
    "phaseType": "INSTALL",  
    "phaseStatus": "SUCCEEDED"  
  },  
  {  
    "durationInSeconds": 0,  
    "endTime": 1548717512.924,  
    "contexts": [  
      {  
        "statusCode": "",  
        "message": ""  
      }  
    ],  
    "startTime": 1548717512.734,  
    "phaseType": "PRE_BUILD",  
    "phaseStatus": "SUCCEEDED"  
  },  
  {  
    "durationInSeconds": 9,  
    "endTime": 1548717522.254,  
    "contexts": [  
      {  
        "statusCode": "",  
        "message": ""  
      }  
    ],  
    "startTime": 1548717512.924,  
    "phaseType": "BUILD",  
    "phaseStatus": "SUCCEEDED"  
  },  
  {  
    "durationInSeconds": 3,  
    "endTime": 1548717525.498,  
    "contexts": [  
      {  
        "statusCode": "",  
        "message": ""  
      }  
    ],  
    "startTime": 1548717522.254,  
    "phaseType": "POST_BUILD",  
    "phaseStatus": "SUCCEEDED"  
  },  
  {
```

```
        "durationInSeconds": 9,
        "endTime": 1548717534.646,
        "contexts": [
            {
                "statusCode": "",
                "message": ""
            }
        ],
        "startTime": 1548717525.498,
        "phaseType": "UPLOAD_ARTIFACTS",
        "phaseStatus": "SUCCEEDED"
    },
    {
        "durationInSeconds": 2,
        "endTime": 1548717536.846,
        "contexts": [
            {
                "statusCode": "",
                "message": ""
            }
        ],
        "startTime": 1548717534.646,
        "phaseType": "FINALIZING",
        "phaseStatus": "SUCCEEDED"
    },
    {
        "startTime": 1548717536.846,
        "phaseType": "COMPLETED"
    }
],
"startTime": 1548717462.122,
"encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3",
"initiator": "codepipeline/CodeBuild-Pipeline",
"secondarySources": [],
"serviceRole": "arn:aws:iam::123456789012:role/service-role/my-codebuild-service-role",
"currentPhase": "COMPLETED",
"id": "codebuild-demo-project:e9c4f4df-3f43-41d2-ab3a-60fe2EXAMPLE",
"cache": {
    "type": "NO_CACHE"
},
"sourceVersion": "arn:aws:s3:::codepipeline-us-west-2-820783811474/CodeBuild-Python-Pip/SourceArti/1TspnN3.zip",
"endTime": 1548717536.846,
```

```
    "arn": "arn:aws:codebuild:us-west-2:123456789012:build/codebuild-demo-
project:e9c4f4df-3f43-41d2-ab3a-60fe2EXAMPLE",
    "queuedTimeoutInMinutes": 480,
    "resolvedSourceVersion": "f2194c1757bbdcb0f8f229254a4b3c8b27d43e0b"
  },
  {
    "artifacts": {
      "md5sum": "",
      "overrideArtifactName": false,
      "location": "arn:aws:s3:::my-artifacts/codebuild-demo-project",
      "encryptionDisabled": false,
      "sha256sum": ""
    },
    "logs": {
      "cloudWatchLogs": {
        "status": "ENABLED"
      },
      "s3Logs": {
        "status": "DISABLED"
      },
      "streamName": "4dea3ca4-20ec-4898-b22a-a9eb9292775d",
      "groupName": "/aws/codebuild/codebuild-demo-project",
      "deepLink": "https://console.aws.amazon.com/cloudwatch/
home?region=us-west-2#logEvent:group=/aws/codebuild/codebuild-demo-
project;stream=4dea3ca4-20ec-4898-b22a-a9eb9292775d"
    },
    "timeoutInMinutes": 60,
    "environment": {
      "privilegedMode": false,
      "computeType": "BUILD_GENERAL1_MEDIUM",
      "image": "aws/codebuild/windows-base:1.0",
      "environmentVariables": [],
      "type": "WINDOWS_CONTAINER"
    },
    "projectName": "codebuild-demo-project",
    "buildComplete": true,
    "source": {
      "gitCloneDepth": 1,
      "location": "https://github.com/my-repo/codebuild-demo-project.git",
      "insecureSsl": false,
      "reportBuildStatus": false,
      "type": "GITHUB"
    },
    "buildStatus": "SUCCEEDED",
```

```
"secondaryArtifacts": [],
"phases": [
  {
    "durationInSeconds": 0,
    "startTime": 1548716241.89,
    "phaseType": "SUBMITTED",
    "endTime": 1548716242.241,
    "phaseStatus": "SUCCEEDED"
  },
  {
    "durationInSeconds": 0,
    "startTime": 1548716242.241,
    "phaseType": "QUEUED",
    "endTime": 1548716242.536,
    "phaseStatus": "SUCCEEDED"
  },
  {
    "durationInSeconds": 33,
    "endTime": 1548716276.171,
    "contexts": [
      {
        "statusCode": "",
        "message": ""
      }
    ],
    "startTime": 1548716242.536,
    "phaseType": "PROVISIONING",
    "phaseStatus": "SUCCEEDED"
  },
  {
    "durationInSeconds": 15,
    "endTime": 1548716291.809,
    "contexts": [
      {
        "statusCode": "",
        "message": ""
      }
    ],
    "startTime": 1548716276.171,
    "phaseType": "DOWNLOAD_SOURCE",
    "phaseStatus": "SUCCEEDED"
  },
  {
    "durationInSeconds": 0,
```

```
"endTime": 1548716291.993,
"contexts": [
  {
    "statusCode": "",
    "message": ""
  }
],
"startTime": 1548716291.809,
"phaseType": "INSTALL",
"phaseStatus": "SUCCEEDED"
},
{
  "durationInSeconds": 0,
  "endTime": 1548716292.191,
  "contexts": [
    {
      "statusCode": "",
      "message": ""
    }
  ],
  "startTime": 1548716291.993,
  "phaseType": "PRE_BUILD",
  "phaseStatus": "SUCCEEDED"
},
{
  "durationInSeconds": 9,
  "endTime": 1548716301.622,
  "contexts": [
    {
      "statusCode": "",
      "message": ""
    }
  ],
  "startTime": 1548716292.191,
  "phaseType": "BUILD",
  "phaseStatus": "SUCCEEDED"
},
{
  "durationInSeconds": 3,
  "endTime": 1548716304.783,
  "contexts": [
    {
      "statusCode": "",
      "message": ""
    }
  ]
}
```

```
    }
  ],
  "startTime": 1548716301.622,
  "phaseType": "POST_BUILD",
  "phaseStatus": "SUCCEEDED"
},
{
  "durationInSeconds": 8,
  "endTime": 1548716313.775,
  "contexts": [
    {
      "statusCode": "",
      "message": ""
    }
  ],
  "startTime": 1548716304.783,
  "phaseType": "UPLOAD_ARTIFACTS",
  "phaseStatus": "SUCCEEDED"
},
{
  "durationInSeconds": 2,
  "endTime": 1548716315.935,
  "contexts": [
    {
      "statusCode": "",
      "message": ""
    }
  ],
  "startTime": 1548716313.775,
  "phaseType": "FINALIZING",
  "phaseStatus": "SUCCEEDED"
},
{
  "startTime": 1548716315.935,
  "phaseType": "COMPLETED"
}
],
"startTime": 1548716241.89,
"secondarySourceVersions": [],
"initiator": "my-codebuild-project",
"arn": "arn:aws:codebuild:us-west-2:123456789012:build/codebuild-demo-
project:815e755f-bade-4a7e-80f0-efe51EXAMPLE",
"encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3",
```

```

        "serviceRole": "arn:aws:iam::123456789012:role/service-role/my-
codebuild-service-role",
        "currentPhase": "COMPLETED",
        "id": "codebuild-demo-project:815e755f-bade-4a7e-80f0-efe51EXAMPLE",
        "cache": {
            "type": "NO_CACHE"
        },
        "endTime": 1548716315.935,
        "secondarySources": [],
        "queuedTimeoutInMinutes": 480,
        "resolvedSourceVersion": "f2194c1757bbdcb0f8f229254a4b3c8b27d43e0b"
    }
]
}

```

For more information, see [View Build Details \(AWS CLI\)](#) in the *AWS CodeBuild User Guide*.

- For API details, see [BatchGetBuilds](#) in *AWS CLI Command Reference*.

batch-get-projects

The following code example shows how to use batch-get-projects.

AWS CLI

To get a list of AWS CodeBuild build project names.

The following batch-get-projects example gets a list of CodeBuild build projects specified by name.

```
aws codebuild batch-get-projects --names codebuild-demo-project codebuild-demo-
project2 my-other-demo-project
```

In the following output, the projectsNotFound array lists any build project names that were specified, but not found. The projects array lists details for each build project where information was found.

```

{
  "projectsNotFound": [],
  "projects": [
    {
      "encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3",

```

```

    "name": "codebuild-demo-project2",
    "queuedTimeoutInMinutes": 480,
    "timeoutInMinutes": 60,
    "source": {
        "buildspec": "version: 0.2\n\n#env:\n #variables:\n    # key:\n\n    # key: \"value\"\n\n    # parameter-store:\n    # key: \"value\"\n\n    # key: \"value\"\n\n    # phases:\n    # install:\n    # commands:\n    # - command\n\n    # - command\n\n    # pre_build:\n    # commands:\n    # - command\n    # - command\n\n    # build:\n    # commands:\n    # - command\n    # - command\n\n    # post_build:\n    # commands:\n    # - command\n    # - command\n\n    # artifacts:\n    # files:\n    # - location\n    # - location\n\n    # name: $(date +%Y-%m-%d)\n\n    # discard-paths: yes\n\n    # base-directory: location\n\n    # cache:\n    # paths:\n    # - paths",
        "type": "NO_SOURCE",
        "insecureSsl": false,
        "gitCloneDepth": 1
    },
    "artifacts": {
        "type": "NO_ARTIFACTS"
    },
    "badge": {
        "badgeEnabled": false
    },
    "lastModified": 1540588091.108,
    "created": 1540588091.108,
    "arn": "arn:aws:codebuild:us-west-2:123456789012:project/test-for-sample",
    "secondarySources": [],
    "secondaryArtifacts": [],
    "cache": {
        "type": "NO_CACHE"
    },
    "serviceRole": "arn:aws:iam::123456789012:role/service-role/my-test-role",
    "environment": {
        "image": "aws/codebuild/java:openjdk-8",
        "privilegedMode": true,
        "type": "LINUX_CONTAINER",
        "computeType": "BUILD_GENERAL1_SMALL",
        "environmentVariables": []
    },
    "tags": []
},
{
    "encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3",

```



```
    "name": "my-other-demo-project",
    "queuedTimeoutInMinutes": 480,
    "timeoutInMinutes": 60,
    "source": {
      "location": "https://github.com/iversonic/codedeploy-sample.git",
      "reportBuildStatus": false,
      "buildspec": "buildspec.yml",
      "insecureSsl": false,
      "gitCloneDepth": 1,
      "type": "GITHUB",
      "auth": {
        "type": "OAUTH"
      }
    },
    "artifacts": {
      "type": "NO_ARTIFACTS"
    },
    "badge": {
      "badgeEnabled": false
    },
    "lastModified": 1523401711.73,
    "created": 1523401711.73,
    "arn": "arn:aws:codebuild:us-west-2:123456789012:project/Project2",
    "cache": {
      "type": "NO_CACHE"
    },
    "serviceRole": "arn:aws:iam::123456789012:role/service-role/codebuild-Project2-service-role",
    "environment": {
      "image": "aws/codebuild/nodejs:4.4.7",
      "privilegedMode": false,
      "type": "LINUX_CONTAINER",
      "computeType": "BUILD_GENERAL1_SMALL",
      "environmentVariables": []
    },
    "tags": []
  }
]
```

For more information, see [View a Build Project's Details \(AWS CLI\)](#) in the *AWS CodeBuild User Guide*.

- For API details, see [BatchGetProjects](#) in *AWS CLI Command Reference*.

batch-get-report-groups

The following code example shows how to use `batch-get-report-groups`.

AWS CLI

To get information about one or more report groups in AWS CodeBuild.

The following `batch-get-report-groups` example retrieves information about the report group with the specified ARN.

```
aws codebuild batch-get-report-groups \
  --report-group-arns arn:aws:codebuild:<region-ID>:<user-ID>:report-group/
  <report-group-name>
```

Output:

```
{
  "reportGroups": [
    {
      "arn": "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/<report-
group-name>",
      "name": "report-group-name",
      "type": "TEST",
      "exportConfig": {
        "exportConfigType": "NO_EXPORT"
      },
      "created": "2020-10-01T18:04:08.466000+00:00",
      "lastModified": "2020-10-01T18:04:08.466000+00:00",
      "tags": []
    }
  ],
  "reportGroupsNotFound": []
}
```

For more information, see [Working with report groups](#) in the *AWS CodeBuild User Guide*.

- For API details, see [BatchGetReportGroups](#) in *AWS CLI Command Reference*.

batch-get-reports

The following code example shows how to use `batch-get-reports`.

AWS CLI

To get information about one or more reports in AWS CodeBuild.

The following `batch-get-reports` example retrieves information about the reports with the specified ARNs.

```
aws codebuild batch-get-reports \  
  --report-arns arn:aws:codebuild:<region-ID>:<user-ID>:report/<report-group-  
name>:<report 1 ID> arn:aws:codebuild:<region-ID>:<user-ID>:report/<report-group-  
name>:<report 2 ID>
```

Output:

```
{  
  "reports": [  
    {  
      "arn": "arn:aws:codebuild:<region-ID>:<user-ID>:report/<report-group-  
name>:<report 1 ID>",  
      "type": "TEST",  
      "name": "<report-group-name>",  
      "reportGroupArn": "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/  
<report-group-name>",  
      "executionId": "arn:aws:codebuild:<region-ID>:<user-ID>:build/test-  
reports:<ID>",  
      "status": "FAILED",  
      "created": "2020-10-01T11:25:22.531000-07:00",  
      "expired": "2020-10-31T11:25:22-07:00",  
      "exportConfig": {  
        "exportConfigType": "NO_EXPORT"  
      },  
      "truncated": false,  
      "testSummary": {  
        "total": 28,  
        "statusCounts": {  
          "ERROR": 5,  
          "FAILED": 1,  
          "SKIPPED": 4,  
          "SUCCEEDED": 18,  
          "UNKNOWN": 0  
        },  
        "durationInNanoSeconds": 94000000  
      },  
    },  
  ]  
}
```

```

    },
    {
      "arn": "arn:aws:codebuild:<region-ID>:<user-ID>:report/<report-group-
name>:<report 2 ID>",
      "type": "TEST",
      "name": "<report-group-name>",
      "reportGroupArn": "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/
<report-group-name>",
      "executionId": "arn:aws:codebuild:<region-ID>:<user-ID>:build/test-
reports:<ID>",
      "status": "FAILED",
      "created": "2020-10-01T11:13:05.816000-07:00",
      "expired": "2020-10-31T11:13:05-07:00",
      "exportConfig": {
        "exportConfigType": "NO_EXPORT"
      },
      "truncated": false,
      "testSummary": {
        "total": 28,
        "statusCounts": {
          "ERROR": 5,
          "FAILED": 1,
          "SKIPPED": 4,
          "SUCCEEDED": 18,
          "UNKNOWN": 0
        },
        "durationInNanoSeconds": 94000000
      },
    }
  ],
  "reportsNotFound": []
}

```

For more information, see [Working with reports](#) in the *AWS CodeBuild User Guide*.

- For API details, see [BatchGetReports](#) in *AWS CLI Command Reference*.

create-project

The following code example shows how to use create-project.

AWS CLI

Example 1: To create an AWS CodeBuild build project

The following `create-project` example creates a CodeBuild build project using source files from an S3 bucket

```
aws codebuild create-project \
  --name "my-demo-project" \
  --source "{\"type\": \"S3\", \"location\": \"codebuild-us-west-2-123456789012-
input-bucket/my-source.zip\"}" \
  --artifacts {"\"type\": \"S3\", \"location\": \"codebuild-us-west-2-123456789012-
output-bucket\""} \
  --environment {"\"type\": \"LINUX_CONTAINER\", \"image\": \"aws/codebuild/
standard:1.0\", \"computeType\": \"BUILD_GENERAL1_SMALL\"}" \
  --service-role "arn:aws:iam::123456789012:role/service-role/my-codebuild-
service-role"
```

Output:

```
{
  "project": {
    "arn": "arn:aws:codebuild:us-west-2:123456789012:project/my-demo-project",
    "name": "my-cli-demo-project",
    "encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3",
    "serviceRole": "arn:aws:iam::123456789012:role/service-role/my-codebuild-
service-role",
    "lastModified": 1556839783.274,
    "badge": {
      "badgeEnabled": false
    },
    "queuedTimeoutInMinutes": 480,
    "environment": {
      "image": "aws/codebuild/standard:1.0",
      "computeType": "BUILD_GENERAL1_SMALL",
      "type": "LINUX_CONTAINER",
      "imagePullCredentialsType": "CODEBUILD",
      "privilegedMode": false,
      "environmentVariables": []
    },
    "artifacts": {
      "location": "codebuild-us-west-2-123456789012-output-bucket",
      "name": "my-cli-demo-project",
      "namespaceType": "NONE",
      "type": "S3",
      "packaging": "NONE",
      "encryptionDisabled": false
    }
  }
}
```

```
    },
    "source": {
      "type": "S3",
      "location": "codebuild-us-west-2-123456789012-input-bucket/my-
source.zip",
      "insecureSsl": false
    },
    "timeoutInMinutes": 60,
    "cache": {
      "type": "NO_CACHE"
    },
    "created": 1556839783.274
  }
}
```

Example 2: To create an AWS CodeBuild build project using a JSON input file for the parameters

The following `create-project` example creates a CodeBuild build project by passing all of the required parameters in a JSON input file. Create the input file template by running the command with only the `--generate-cli-skeleton` parameter.

```
aws codebuild create-project --cli-input-json file://create-project.json
```

The input JSON file `create-project.json` contains the following content:

```
{
  "name": "codebuild-demo-project",
  "source": {
    "type": "S3",
    "location": "codebuild-region-ID-account-ID-input-bucket/MessageUtil.zip"
  },
  "artifacts": {
    "type": "S3",
    "location": "codebuild-region-ID-account-ID-output-bucket"
  },
  "environment": {
    "type": "LINUX_CONTAINER",
    "image": "aws/codebuild/standard:1.0",
    "computeType": "BUILD_GENERAL1_SMALL"
  },
  "serviceRole": "serviceIAMRole"
}
```

```
}

```

Output:

```
{
  "project": {
    "name": "codebuild-demo-project",
    "serviceRole": "serviceIAMRole",
    "tags": [],
    "artifacts": {
      "packaging": "NONE",
      "type": "S3",
      "location": "codebuild-region-ID-account-ID-output-bucket",
      "name": "message-util.zip"
    },
    "lastModified": 1472661575.244,
    "timeoutInMinutes": 60,
    "created": 1472661575.244,
    "environment": {
      "computeType": "BUILD_GENERAL1_SMALL",
      "image": "aws/codebuild/standard:1.0",
      "type": "LINUX_CONTAINER",
      "environmentVariables": []
    },
    "source": {
      "type": "S3",
      "location": "codebuild-region-ID-account-ID-input-bucket/
MessageUtil.zip"
    },
    "encryptionKey": "arn:aws:kms:region-ID:account-ID:alias/aws/s3",
    "arn": "arn:aws:codebuild:region-ID:account-ID:project/codebuild-demo-
project"
  }
}
```

For more information, see [Create a Build Project \(AWS CLI\)](#) in the *AWS CodeBuild User Guide*.

- For API details, see [CreateProject](#) in *AWS CLI Command Reference*.

create-report-group

The following code example shows how to use create-report-group.

AWS CLI

To create a report group in AWS CodeBuild.

The following `create-report-group` example creates a new report group.

```
aws codebuild create-report-group \  
  --cli-input-json file://create-report-group-source.json
```

Contents of `create-report-group-source.json`:

```
{  
  "name": "cli-created-report-group",  
  "type": "TEST",  
  "exportConfig": {  
    "exportConfigType": "S3",  
    "s3Destination": {  
      "bucket": "my-s3-bucket",  
      "path": "",  
      "packaging": "ZIP",  
      "encryptionDisabled": true  
    }  
  }  
}
```

Output:

```
{  
  "reportGroup": {  
    "arn": "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/cli-created-report-group",  
    "name": "cli-created-report-group",  
    "type": "TEST",  
    "exportConfig": {  
      "exportConfigType": "S3",  
      "s3Destination": {  
        "bucket": "my-s3-bucket",  
        "path": "",  
        "packaging": "ZIP",  
        "encryptionDisabled": true  
      }  
    }  
  },  
}
```



```

    "created": 1602020026.775,
    "lastModified": 1602020026.775
  }
}

```

For more information, see [Working with report groups](#) in the *AWS CodeBuild User Guide*.

- For API details, see [CreateReportGroup](#) in *AWS CLI Command Reference*.

create-webhook

The following code example shows how to use `create-webhook`.

AWS CLI

To create webhook filters for an AWS CodeBuild project

The following `create-webhook` example creates a webhook for a CodeBuild project named `my-project` that has two filter groups. The first filter group specifies pull requests that are created, updated, or reopened on branches with Git reference names that match the regular expression `^refs/heads/master$` and head references that match `^refs/heads/myBranch$`. The second filter group specifies push requests on branches with Git reference names that do not match the regular expression `^refs/heads/myBranch$`.

```

aws codebuild create-webhook \
  --project-name my-project \
  --filter-groups "[[{"type":"EVENT","pattern":"PULL_REQUEST_CREATED,
  PULL_REQUEST_UPDATED, PULL_REQUEST_REOPENED"}, {"type":"HEAD_REF","pattern
  \":"^refs/heads/myBranch$","excludeMatchedPattern":true}, {"type":"BASE_REF
  \","pattern":"^refs/heads/master$","excludeMatchedPattern":true}], [{"type":
  "EVENT","pattern":"PUSH"}, {"type":"HEAD_REF","pattern":"^refs/heads/
  myBranch$","excludeMatchedPattern":true}]]]"

```

Output:

```

{
  "webhook": {
    "payloadUrl": "https://codebuild.us-west-2.amazonaws.com/webhooks?
    t=eyJlbnNyeXB0ZWREYXRhIjoiVVl5MGtoeGRwSzZFRXl2Wnh4bldlZ0tKZ291TVpQNEtFamQ3RDlDYWpRaGIreVFrdm
    "url": "https://api.github.com/repos/iversonic/codedeploy-sample/
    hooks/105190656",
    "lastModifiedSecret": 1556311319.069,

```

```

    "filterGroups": [
      [
        {
          "type": "EVENT",
          "pattern": "PULL_REQUEST_CREATED, PULL_REQUEST_UPDATED,
PULL_REQUEST_REOPENED",
          "excludeMatchedPattern": false
        },
        {
          "type": "HEAD_REF",
          "pattern": "refs/heads/myBranch$",
          "excludeMatchedPattern": true
        },
        {
          "type": "BASE_REF",
          "pattern": "refs/heads/master$",
          "excludeMatchedPattern": true
        }
      ],
      [
        {
          "type": "EVENT",
          "pattern": "PUSH",
          "excludeMatchedPattern": false
        },
        {
          "type": "HEAD_REF",
          "pattern": "refs/heads/myBranch$",
          "excludeMatchedPattern": true
        }
      ]
    ]
  }
}

```

For more information, see [Filter GitHub Webhook Events \(SDK\)](#) in the *AWS CodeBuild User Guide*.

- For API details, see [CreateWebhook](#) in *AWS CLI Command Reference*.

delete-build-batch

The following code example shows how to use delete-build-batch.

AWS CLI

To delete a batch build in AWS CodeBuild.

The following `delete-build-batch` example deletes the specified batch build.

```
aws codebuild delete-build-batch \  
  --id <project-name>:<batch-ID>
```

Output:

```
{  
  "statusCode": "BATCH_DELETED",  
  "buildsDeleted": [  
    "arn:aws:codebuild:<region-ID>:<account-ID>:build/<project-name>:<build-ID>",  
    "arn:aws:codebuild:<region-ID>:<account-ID>:build/<project-name>:<build-ID>",  
    "arn:aws:codebuild:<region-ID>:<account-ID>:build/<project-name>:<build-ID>",  
    "arn:aws:codebuild:<region-ID>:<account-ID>:build/<project-name>:<build-ID>"  
  ],  
  "buildsNotDeleted": []  
}
```

For more information, see [Batch builds in AWS CodeBuild](#) in the *AWS CodeBuild User Guide*.

- For API details, see [DeleteBuildBatch](#) in *AWS CLI Command Reference*.

delete-project

The following code example shows how to use `delete-project`.

AWS CLI

To delete an AWS CodeBuild build project

The following `delete-project` example deletes the specified CodeBuild build project.

```
aws codebuild delete-project --name my-project
```

This command produces no output.

For more information, see [Delete a Build Project \(AWS CLI\)](#) in the *AWS CodeBuild User Guide*.

- For API details, see [DeleteProject](#) in *AWS CLI Command Reference*.

delete-report-group

The following code example shows how to use `delete-report-group`.

AWS CLI

To delete a report groups in AWS CodeBuild.

The following `delete-report-group` example deletes the report group with the specified ARN.

```
aws codebuild delete-report-group \  
  --arn arn:aws:codebuild:<region-ID>:<user-ID>:report-group/<report-group-name>
```

This command produces no output.

For more information, see [Working with report groups](#) in the *AWS CodeBuild User Guide*.

- For API details, see [DeleteReportGroup](#) in *AWS CLI Command Reference*.

delete-report

The following code example shows how to use `delete-report`.

AWS CLI

To delete a report in AWS CodeBuild.

The following `delete-report` example deletes the specified report.

```
aws codebuild delete-report \  
  --arn arn:aws:codebuild:<region-ID>:<account-ID>:report/<report-group-  
name>:<report-ID>
```

This command produces no output.

For more information, see [Working with reports](#) in the *AWS CodeBuild User Guide*.

- For API details, see [DeleteReport](#) in *AWS CLI Command Reference*.

delete-source-credentials

The following code example shows how to use `delete-source-credentials`.

AWS CLI

To disconnect from a source provider and remove its access tokens.

The following `delete-source-credentials` example disconnects from a source provider and removes its tokens. The ARN of source credentials used to connect to the source provider determines which source credentials.

```
aws codebuild delete-source-credentials --arn arn-of-your-credentials
```

Output:

```
{
  "arn": "arn:aws:codebuild:your-region:your-account-id:token/your-server-type"
}
```

For more information, see [Connect Source Providers with Access Tokens \(CLI\)](#) in the *AWS CodeBuild User Guide*.

- For API details, see [DeleteSourceCredentials](#) in *AWS CLI Command Reference*.

delete-webhook

The following code example shows how to use `delete-webhook`.

AWS CLI

To delete a webhook filter from an AWS CodeBuild project

The following `delete-webhook` example deletes a webhook from the specified CodeBuild project.

```
aws codebuild delete-webhook --project-name my-project
```

This command produces no output.

For more information, see [Stop Running Builds Automatically \(AWS CLI\)](#) in the *AWS CodeBuild User Guide*.

- For API details, see [DeleteWebhook](#) in *AWS CLI Command Reference*.

describe-code-coverages

The following code example shows how to use `describe-code-coverages`.

AWS CLI

To get detailed information about code coverage test results in AWS CodeBuild.

The following `describe-code-coverages` example gets information about the code coverage test results in the specified report.

```
aws codebuild describe-code-coverages \
  --report-arn arn:aws:codebuild:<region-ID>:<account-ID>:report/<report-group-name>:<report-ID>
```

Output:

```
{
  "codeCoverages": [
    {
      "id": "20a0adcc-db13-4b66-804b-ecaf9f852855",
      "reportARN": "arn:aws:codebuild:<region-ID>:972506530580:report/<report-group-name>:<report-ID>",
      "filePath": "<source-file-1-path>",
      "lineCoveragePercentage": 83.33,
      "linesCovered": 5,
      "linesMissed": 1,
      "branchCoveragePercentage": 50.0,
      "branchesCovered": 1,
      "branchesMissed": 1,
      "expired": "2020-11-20T21:22:45+00:00"
    },
    {
      "id": "0887162d-bf57-4cf1-a164-e432373d1a83",
      "reportARN": "arn:aws:codebuild:<region-ID>:972506530580:report/<report-group-name>:<report-ID>",
      "filePath": "<source-file-2-path>",
```

```

        "lineCoveragePercentage": 90.9,
        "linesCovered": 10,
        "linesMissed": 1,
        "branchCoveragePercentage": 50.0,
        "branchesCovered": 1,
        "branchesMissed": 1,
        "expired": "2020-11-20T21:22:45+00:00"
    }
]
}

```

For more information, see [Code coverage reports](#) in the *AWS CodeBuild User Guide*.

- For API details, see [DescribeCodeCoverages](#) in *AWS CLI Command Reference*.

describe-test-cases

The following code example shows how to use `describe-test-cases`.

AWS CLI

To get detailed information about test cases in AWS CodeBuild.

The following `describe-test-cases` example gets information about the test cases in the specified report.

```

aws codebuild describe-test-cases \
  --report-arn arn:aws:codebuild:<region-ID>:<account-ID>:report/<report-group-name>:<report-ID>

```

Output:

```

{
  "testCases": [
    {
      "reportArn": "arn:aws:codebuild:<region-ID>:<account-ID>:report/<report-group-name>:<report-ID>",
      "testRawDataPath": "<test-report-path>",
      "prefix": "NUnit.Tests.Assemblies.MockTestFixture",
      "name": "NUnit.Tests.Assemblies.MockTestFixture.NotRunnableTest",
      "status": "ERROR",
      "durationInNanoSeconds": 0,
    }
  ]
}

```

```

        "message": "No arguments were provided\n",
        "expired": "2020-11-20T17:52:10+00:00"
    },
    {
        "reportArn": "arn:aws:codebuild:<region-ID>:<account-ID>:report/<report-
group-name>:<report-ID>",
        "testRawDataPath": "<test-report-path>",
        "prefix": "NUnit.Tests.Assemblies.MockTestFixture",
        "name": "NUnit.Tests.Assemblies.MockTestFixture.TestWithException",
        "status": "ERROR",
        "durationInNanoSeconds": 0,
        "message": "System.ApplicationException : Intentional Exception
\nat NUnit.Tests.Assemblies.MockTestFixture.MethodThrowsException()\nat
NUnit.Tests.Assemblies.MockTestFixture.TestWithException()\n\n",
        "expired": "2020-11-20T17:52:10+00:00"
    }
]
}

```

For more information, see [Working with test reporting in AWS CodeBuild](#) in the *AWS CodeBuild User Guide*.

- For API details, see [DescribeTestCases](#) in *AWS CLI Command Reference*.

import-source-credentials

The following code example shows how to use `import-source-credentials`.

AWS CLI

Connect an AWS CodeBuild user to a source provider by importing credentials for the source provider.

The following `import-source-credentials` example imports a token for a Bitbucket repository that uses `BASIC_AUTH` for its authentication type.

```
aws codebuild import-source-credentials --server-type BITBUCKET --auth-type
BASIC_AUTH --token my-Bitbucket-password --username my-Bitbucket-username
```

Output:

```
{
```



```
"arn": "arn:aws:codebuild:us-west-2:123456789012:token/bitbucket"
}
```

For more information, see [Connect Source Providers with Access Tokens \(CLI\)](#) in the *AWS CodeBuild User Guide*.

- For API details, see [ImportSourceCredentials](#) in *AWS CLI Command Reference*.

invalidate-project-cache

The following code example shows how to use `invalidate-project-cache`.

AWS CLI

To reset the cache for an AWS CodeBuild build project.

The following `invalidate-project-cache` example resets the cache for the specified CodeBuild project.

```
aws codebuild invalidate-project-cache --project-name my-project
```

This command produces no output.

For more information, see [Build Caching in CodeBuild](#) in the *AWS CodeBuild User Guide*.

- For API details, see [InvalidateProjectCache](#) in *AWS CLI Command Reference*.

list-build-batches-for-project

The following code example shows how to use `list-build-batches-for-project`.

AWS CLI

To list batch builds for a specific build project in AWS CodeBuild.

The following `list-build-batches-for-project` example lists the CodeBuild batch builds for the specified project.

```
aws codebuild list-build-batches-for-project \  
  --project-name "<project-name>"
```

Output:

```
{
  "ids": [
    "<project-name>:<batch-ID>",
    "<project-name>:<batch-ID>"
  ]
}
```

For more information, see [Batch builds in AWS CodeBuild](#) in the *AWS CodeBuild User Guide*.

- For API details, see [ListBuildBatchesForProject](#) in *AWS CLI Command Reference*.

list-build-batches

The following code example shows how to use `list-build-batches`.

AWS CLI

To list batch builds in AWS CodeBuild.

The following `list-build-batches` example lists the CodeBuild batch builds for the current account.

```
aws codebuild list-build-batches
```

Output:

```
{
  "ids": [
    "<project-name>:<batch-ID>",
    "<project-name>:<batch-ID>"
  ]
}
```

For more information, see Batch builds in AWS CodeBuild (<<https://docs.aws.amazon.com/codebuild/latest/userguide/batch-build.html>>)_ in the *AWS CodeBuild User Guide*.

- For API details, see [ListBuildBatches](#) in *AWS CLI Command Reference*.

list-builds-for-project

The following code example shows how to use `list-builds-for-project`.

AWS CLI

To view a list of builds for an AWS CodeBuild build project.

The following `list-builds-for-project` example lists the build IDs in descending order for the specified CodeBuild build project.

```
aws codebuild list-builds-for-project --project-name codebuild-demo-project --sort-order DESCENDING
```

Output:

```
{
  "ids": [
    "codebuild-demo-project:1a2b3c4d-5678-90ab-cdef-11111example",
    "codebuild-demo-project:1a2b3c4d-5678-90ab-cdef-22222example",
    "codebuild-demo-project:1a2b3c4d-5678-90ab-cdef-33333example",
    "codebuild-demo-project:1a2b3c4d-5678-90ab-cdef-44444example",
    "codebuild-demo-project:1a2b3c4d-5678-90ab-cdef-55555example"
  ]
}
```

For more information, see [View a List of Build IDs for a Build Project \(AWS CLI\)](#) in the *AWS CodeBuild User Guide*

- For API details, see [ListBuildsForProject](#) in *AWS CLI Command Reference*.

list-builds

The following code example shows how to use `list-builds`.

AWS CLI

To get a list of AWS CodeBuild builds IDs.

The following `list-builds` example gets a list of CodeBuild IDs sorted in ascending order.

```
aws codebuild list-builds --sort-order ASCENDING
```

The output includes a `nextToken` value which indicates that there is more output available.

```
{
  "nextToken": "4AEA6u7J...The full token has been omitted for
  brevity...MzY20A==",
  "ids": [
    "codebuild-demo-project:815e755f-bade-4a7e-80f0-efe51EXAMPLE"
    "codebuild-demo-project:84a7f3d1-d40e-4956-b4cf-7a9d4EXAMPLE"
    ... The full list of build IDs has been omitted for brevity ...
    "codebuild-demo-project:931d0b72-bf6f-4040-a472-5c707EXAMPLE"
  ]
}
```

Run this command again and provide the `nextToken` value in the previous response as a parameter to get the next part of the output. Repeat until you don't receive a `nextToken` value in the response.

```
aws codebuild list-builds --sort-order ASCENDING --next-token 4AEA6u7J...The full
token has been omitted for brevity...MzY20A==
```

Next part of the output:

```
{
  "ids": [
    "codebuild-demo-project:49015049-21cf-4b50-9708-df115EXAMPLE",
    "codebuild-demo-project:543e7206-68a3-46d6-a4da-759abEXAMPLE",
    ... The full list of build IDs has been omitted for brevity ...
    "codebuild-demo-project:c282f198-4582-4b38-bdc0-26f96EXAMPLE"
  ]
}
```

For more information, see [View a List of Build IDs \(AWS CLI\)](#) in the *AWS CodeBuild User Guide*

- For API details, see [ListBuilds](#) in *AWS CLI Command Reference*.

list-curated-environment-images

The following code example shows how to use `list-curated-environment-images`.

AWS CLI

To get a list of Docker images managed by AWS CodeBuild that you can use for your builds.

The following `list-curated-environment-images` example lists the Docker images managed by CodeBuild that can be used for builds.:

```
aws codebuild list-curated-environment-images
```

Output:

```
{
  "platforms": [
    {
      "platform": "AMAZON_LINUX",
      "languages": [
        {
          "language": "JAVA",
          "images": [
            {
              "description": "AWS ElasticBeanstalk - Java 7 Running on
Amazon Linux 64bit v2.1.3",
              "name": "aws/codebuild/eb-java-7-amazonlinux-64:2.1.3",
              "versions": [
                "aws/codebuild/eb-java-7-amazonlinux-64:2.1.3-1.0.0"
              ]
            },
            {
              "description": "AWS ElasticBeanstalk - Java 8 Running on
Amazon Linux 64bit v2.1.3",
              "name": "aws/codebuild/eb-java-8-amazonlinux-64:2.1.3",
              "versions": [
                "aws/codebuild/eb-java-8-amazonlinux-64:2.1.3-1.0.0"
              ]
            },
            ... LIST TRUNCATED FOR BREVITY ...
          ]
        }
      ]
    }
  ]
}
```

For more information, see [Docker Images Provided by CodeBuild](#) in the *AWS CodeBuild User Guide*

- For API details, see [ListCuratedEnvironmentImages](#) in *AWS CLI Command Reference*.

list-projects

The following code example shows how to use `list-projects`.

AWS CLI

To get a list of AWS CodeBuild build project names.

The following `list-projects` example gets a list of CodeBuild build projects sorted by name in ascending order.

```
aws codebuild list-projects --sort-by NAME --sort-order ASCENDING
```

The output includes a `nextToken` value which indicates that there is more output available.

```
{
  "nextToken": "Ci33ACF6...The full token has been omitted for brevity...U
+AkMx8=",
  "projects": [
    "codebuild-demo-project",
    "codebuild-demo-project2",
    ... The full list of build project names has been omitted for
brevity ...
    "codebuild-demo-project99"
  ]
}
```

Run this command again and provide the `nextToken` value from the previous response as a parameter to get the next part of the output. Repeat until you don't receive a `nextToken` value in the response.

```
aws codebuild list-projects --sort-by NAME --sort-order ASCENDING --next-token
Ci33ACF6...The full token has been omitted for brevity...U+AkMx8=

{
  "projects": [
    "codebuild-demo-project100",
    "codebuild-demo-project101",
    ... The full list of build project names has been omitted for
brevity ...
    "codebuild-demo-project122"
  ]
}
```

```
}
```

For more information, see [View a List of Build Project Names \(AWS CLI\)](#) in the *AWS CodeBuild User Guide*.

- For API details, see [ListProjects](#) in *AWS CLI Command Reference*.

list-report-groups

The following code example shows how to use `list-report-groups`.

AWS CLI

To get a list of the report group ARNs in AWS CodeBuild.

The following `list-report-groups` example retrieves the report group ARNs for the account in the region.

```
aws codebuild list-report-groups
```

Output:

```
{
  "reportGroups": [
    "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/report-group-1",
    "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/report-group-2",
    "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/report-group-3"
  ]
}
```

For more information, see [Working with report groups](#) in the *AWS CodeBuild User Guide*.

- For API details, see [ListReportGroups](#) in *AWS CLI Command Reference*.

list-reports-for-report-group

The following code example shows how to use `list-reports-for-report-group`.

AWS CLI

To get a list of the reports in a report group in AWS CodeBuild.

The following `list-reports-for-report-groups` example retrieves the reports in the specified report group for the account in the region.

```
aws codebuild list-reports-for-report-group \  
  --report-group-arn arn:aws:codebuild:<region-ID>:<user-ID>:report-group/<report-  
group-name>
```

Output:

```
{  
  "reports": [  
    "arn:aws:codebuild:<region-ID>:<user-ID>:report/report-1",  
    "arn:aws:codebuild:<region-ID>:<user-ID>:report/report-2",  
    "arn:aws:codebuild:<region-ID>:<user-ID>:report/report-3"  
  ]  
}
```

For more information, see [Working with report groups](#) in the *AWS CodeBuild User Guide*.

- For API details, see [ListReportsForReportGroup](#) in *AWS CLI Command Reference*.

list-reports

The following code example shows how to use `list-reports`.

AWS CLI

To get a list of the reports for the current account in AWS CodeBuild.

The following `list-reports` example retrieves the ARNs of the reports for the current account.

```
aws codebuild list-reports
```

Output:

```
{  
  "reports": [  
    "arn:aws:codebuild:<region-ID>:<user-ID>:report/<report-group-name>:<report  
ID>",  
    "arn:aws:codebuild:<region-ID>:<user-ID>:report/<report-group-name>:<report  
ID>",
```



```
        "arn:aws:codebuild:<region-ID>:<user-ID>:report/<report-group-name>:<report
ID>"
    ]
}
```

For more information, see [Working with reports](#) in the *AWS CodeBuild User Guide*.

- For API details, see [ListReports](#) in *AWS CLI Command Reference*.

list-shared-projects

The following code example shows how to use `list-shared-projects`.

AWS CLI

To list the shared project in AWS CodeBuild.

The following `list-shared-projects` example lists the CodeBuild shared projects that are available to the current account.

```
aws codebuild list-shared-projects
```

Output:

```
{
  "projects": [
    "arn:aws:codebuild:<region-ID>:<account-ID>:project/<shared-project-
name-1>",
    "arn:aws:codebuild:<region-ID>:<account-ID>:project/<shared-project-name-2>"
  ]
}
```

For more information, see [Working with shared projects](#) in the *AWS CodeBuild User Guide*.

- For API details, see [ListSharedProjects](#) in *AWS CLI Command Reference*.

list-shared-report-groups

The following code example shows how to use `list-shared-report-groups`.

AWS CLI

To get a list of the shared report group ARNs in AWS CodeBuild.

The following `list-shared-report-groups` example retrieves the report group ARNs for the account in the region.

```
aws codebuild list-shared-report-groups
```

Output:

```
{
  "reportGroups": [
    "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/report-group-1",
    "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/report-group-2",
    "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/report-group-3"
  ]
}
```

For more information, see [Working with report groups](#) in the *AWS CodeBuild User Guide*.

- For API details, see [ListSharedReportGroups](#) in *AWS CLI Command Reference*.

list-source-credentials

The following code example shows how to use `list-source-credentials`.

AWS CLI

To view a list of `sourceCredentialsObjects`

The following `list-source-credentials` example lists tokens for an AWS account connected to one Bitbucket account and one GitHub account. Each `sourceCredentialsInfos` object in the response contains connected source credentials information.

```
aws codebuild list-source-credentials
```

Output:

```
{
  "sourceCredentialsInfos": [
    {
      "serverType": "BITBUCKET",
      "arn": "arn:aws:codebuild:us-west-2:123456789012:token/bitbucket",
      "authType": "BASIC_AUTH"
    }
  ]
}
```

```

    },
    {
      "serverType": "GITHUB",
      "arn": "arn:aws:codebuild:us-west-2:123456789012:token/github",
      "authType": "OAUTH"
    }
  ]
}

```

For more information, see [Connect Source Providers with Access Tokens \(CLI\)](#) in the *AWS CodeBuild User Guide*.

- For API details, see [ListSourceCredentials](#) in *AWS CLI Command Reference*.

retry-build-batch

The following code example shows how to use `retry-build-batch`.

AWS CLI

To retry a failed batch build in AWS CodeBuild.

The following `retry-build-batch` example restarts the specified batch build.

```

aws codebuild retry-build-batch \
  --id <project-name>:<batch-ID>

```

Output:

```

{
  "buildBatch": {
    "id": "<project-name>:<batch-ID>",
    "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build-batch/<project-name>:<batch-ID>",
    "startTime": "2020-10-21T17:26:23.099000+00:00",
    "currentPhase": "SUBMITTED",
    "buildBatchStatus": "IN_PROGRESS",
    "resolvedSourceVersion": "3a9e11cb419e8fff14b03883dc4e64f6155aaa7e",
    "projectName": "<project-name>",
    "phases": [
      {
        "phaseType": "SUBMITTED",
        "phaseStatus": "SUCCEEDED",

```

```

    "startTime": "2020-10-21T17:26:23.099000+00:00",
    "endTime": "2020-10-21T17:26:23.457000+00:00",
    "durationInSeconds": 0
  },
  {
    "phaseType": "DOWNLOAD_BATCHSPEC",
    "phaseStatus": "SUCCEEDED",
    "startTime": "2020-10-21T17:26:23.457000+00:00",
    "endTime": "2020-10-21T17:26:54.902000+00:00",
    "durationInSeconds": 31
  },
  {
    "phaseType": "IN_PROGRESS",
    "phaseStatus": "CLIENT_ERROR",
    "startTime": "2020-10-21T17:26:54.902000+00:00",
    "endTime": "2020-10-21T17:28:16.060000+00:00",
    "durationInSeconds": 81
  },
  {
    "phaseType": "FAILED",
    "phaseStatus": "RETRY",
    "startTime": "2020-10-21T17:28:16.060000+00:00",
    "endTime": "2020-10-21T17:29:39.709000+00:00",
    "durationInSeconds": 83
  },
  {
    "phaseType": "SUBMITTED",
    "startTime": "2020-10-21T17:29:39.709000+00:00"
  }
],
"source": {
  "type": "GITHUB",
  "location": "https://github.com/strohm-a/<project-name>-graph.git",
  "gitCloneDepth": 1,
  "gitSubmodulesConfig": {
    "fetchSubmodules": false
  },
  "reportBuildStatus": false,
  "insecureSsl": false
},
"secondarySources": [],
"secondarySourceVersions": [],
"artifacts": {
  "location": ""
}

```

```

    },
    "secondaryArtifacts": [],
    "cache": {
      "type": "NO_CACHE"
    },
    },
    "environment": {
      "type": "LINUX_CONTAINER",
      "image": "aws/codebuild/amazonlinux2-x86_64-standard:3.0",
      "computeType": "BUILD_GENERAL1_SMALL",
      "environmentVariables": [],
      "privilegedMode": false,
      "imagePullCredentialsType": "CODEBUILD"
    },
    },
    "logConfig": {
      "cloudWatchLogs": {
        "status": "ENABLED"
      },
      "s3Logs": {
        "status": "DISABLED",
        "encryptionDisabled": false
      }
    },
    },
    "buildTimeoutInMinutes": 60,
    "queuedTimeoutInMinutes": 480,
    "complete": false,
    "initiator": "<username>",
    "encryptionKey": "arn:aws:kms:<region-ID>:<account-ID>:alias/aws/s3",
    "buildBatchNumber": 4,
    "buildBatchConfig": {
      "serviceRole": "arn:aws:iam::<account-ID>:role/service-role/<project-
name>",
      "restrictions": {
        "maximumBuildsAllowed": 100
      },
      "timeoutInMins": 480
    },
    },
    "buildGroups": [
      {
        "identifier": "DOWNLOAD_SOURCE",
        "ignoreFailure": false,
        "currentBuildSummary": {
          "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build/
<project-name>:<build-ID>",
          "requestedOn": "2020-10-21T17:26:23.889000+00:00",

```

```

        "buildStatus": "SUCCEEDED",
        "primaryArtifact": {
            "type": "no_artifacts",
            "identifier": "DOWNLOAD_SOURCE"
        },
        "secondaryArtifacts": []
    }
},
{
    "identifier": "linux_small",
    "dependsOn": [],
    "ignoreFailure": false,
    "currentBuildSummary": {
        "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build/
<project-name>:<build-ID>",
        "requestedOn": "2020-10-21T17:26:55.115000+00:00",
        "buildStatus": "FAILED",
        "primaryArtifact": {
            "type": "no_artifacts",
            "identifier": "linux_small"
        },
        "secondaryArtifacts": []
    }
},
{
    "identifier": "linux_medium",
    "dependsOn": [
        "linux_small"
    ],
    "ignoreFailure": false,
    "currentBuildSummary": {
        "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build/
<project-name>:<build-ID>",
        "requestedOn": "2020-10-21T17:26:54.594000+00:00",
        "buildStatus": "STOPPED"
    }
},
{
    "identifier": "linux_large",
    "dependsOn": [
        "linux_medium"
    ],
    "ignoreFailure": false,
    "currentBuildSummary": {

```

```

        "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build/
<project-name>:<build-ID>",
        "requestedOn": "2020-10-21T17:26:54.701000+00:00",
        "buildStatus": "STOPPED"
    }
}
]
}
}

```

For more information, see [Batch builds in AWS CodeBuild](#) in the *AWS CodeBuild User Guide*.

- For API details, see [RetryBuildBatch](#) in *AWS CLI Command Reference*.

retry-build

The following code example shows how to use `retry-build`.

AWS CLI

To retry a failed build in AWS CodeBuild.

The following `retry-build` example restarts the specified build.

```

aws codebuild retry-build \
  --id <project-name>:<build-ID>

```

Output:

```

{
  "build": {
    "id": "<project-name>:<build-ID>",
    "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build/<project-
name>:<build-ID>",
    "buildNumber": 9,
    "startTime": "2020-10-21T17:51:38.161000+00:00",
    "currentPhase": "QUEUED",
    "buildStatus": "IN_PROGRESS",
    "projectName": "<project-name>",
    "phases": [
      {
        "phaseType": "SUBMITTED",
        "phaseStatus": "SUCCEEDED",

```

```
        "startTime": "2020-10-21T17:51:38.161000+00:00",
        "endTime": "2020-10-21T17:51:38.210000+00:00",
        "durationInSeconds": 0
    },
    {
        "phaseType": "QUEUED",
        "startTime": "2020-10-21T17:51:38.210000+00:00"
    }
],
"source": {
    "type": "GITHUB",
    "location": "<GitHub-repo-URL>",
    "gitCloneDepth": 1,
    "gitSubmodulesConfig": {
        "fetchSubmodules": false
    },
    "reportBuildStatus": false,
    "insecureSsl": false
},
"secondarySources": [],
"secondarySourceVersions": [],
"artifacts": {
    "location": ""
},
"secondaryArtifacts": [],
"cache": {
    "type": "NO_CACHE"
},
"environment": {
    "type": "LINUX_CONTAINER",
    "image": "aws/codebuild/amazonlinux2-x86_64-standard:3.0",
    "computeType": "BUILD_GENERAL1_SMALL",
    "environmentVariables": [],
    "privilegedMode": false,
    "imagePullCredentialsType": "CODEBUILD"
},
"serviceRole": "arn:aws:iam::<account-ID>:role/service-role/<service-role-name>",
"logs": {
    "deepLink": "https://console.aws.amazon.com/cloudwatch/home?region=<region-ID>#logEvent:group=null;stream=null",
    "cloudWatchLogsArn": "arn:aws:logs:<region-ID>:<account-ID>:log-group:null:log-stream:null",
    "cloudWatchLogs": {
```



```

        "status": "ENABLED"
    },
    "s3Logs": {
        "status": "DISABLED",
        "encryptionDisabled": false
    }
},
"timeoutInMinutes": 60,
"queuedTimeoutInMinutes": 480,
"buildComplete": false,
"initiator": "<username>",
"encryptionKey": "arn:aws:kms:<region-ID>:<account-ID>:alias/aws/s3"
}
}

```

For more information, see [Batch builds in AWS CodeBuild](#) in the *AWS CodeBuild User Guide*.

- For API details, see [RetryBuild](#) in *AWS CLI Command Reference*.

start-build-batch

The following code example shows how to use start-build-batch.

AWS CLI

To start a batch build in AWS CodeBuild.

The following start-build-batch example starts a batch build of the specified project.

```
aws codebuild start-build-batch \
  --project-name <project-name>
```

Output:

```
{
  "buildBatch": {
    "id": "<project-name>:<batch-ID>",
    "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build-batch/<project-name>:<batch-ID>",
    "startTime": "2020-10-21T16:54:24.740000+00:00",
    "currentPhase": "SUBMITTED",
    "buildBatchStatus": "IN_PROGRESS",
    "projectName": "<project-name>",
  }
}
```

```
"source": {
  "type": "GITHUB",
  "location": "<GitHub-repo-URL>",
  "gitCloneDepth": 1,
  "gitSubmodulesConfig": {
    "fetchSubmodules": false
  },
  "reportBuildStatus": false,
  "insecureSsl": false
},
"secondarySources": [],
"secondarySourceVersions": [],
"artifacts": {
  "location": ""
},
"secondaryArtifacts": [],
"cache": {
  "type": "NO_CACHE"
},
"environment": {
  "type": "LINUX_CONTAINER",
  "image": "aws/codebuild/amazonlinux2-x86_64-standard:3.0",
  "computeType": "BUILD_GENERAL1_SMALL",
  "environmentVariables": [],
  "privilegedMode": false,
  "imagePullCredentialsType": "CODEBUILD"
},
"logConfig": {
  "cloudWatchLogs": {
    "status": "ENABLED"
  },
  "s3Logs": {
    "status": "DISABLED",
    "encryptionDisabled": false
  }
},
"buildTimeoutInMinutes": 60,
"queuedTimeoutInMinutes": 480,
"complete": false,
"initiator": "<username>",
"encryptionKey": "arn:aws:kms:<region-ID>:<account-ID>:alias/aws/s3",
"buildBatchNumber": 3,
"buildBatchConfig": {
```

```

        "serviceRole": "arn:aws:iam::<account-ID>:role/service-role/<service-
role-name>",
        "restrictions": {
            "maximumBuildsAllowed": 100
        },
        "timeoutInMins": 480
    }
}

```

For more information, see [Batch builds in AWS CodeBuild](#) in the *AWS CodeBuild User Guide*.

- For API details, see [StartBuildBatch](#) in *AWS CLI Command Reference*.

start-build

The following code example shows how to use `start-build`.

AWS CLI

To start running a build of an AWS CodeBuild build project.

The following `start-build` example starts a build for the specified CodeBuild project. The build overrides both the project's setting for the number of minutes the build is allowed to be queued before it times out and the project's artifact settings.

```

aws codebuild start-build \
  --project-name "my-demo-project" \
  --queued-timeout-in-minutes-override 5 \
  --artifacts-override {"type": "S3","location": "arn:aws:s3:::artifacts-
override","overrideArtifactName":true}

```

Output:

```

{
  "build": {
    "serviceRole": "arn:aws:iam::123456789012:role/service-role/my-codebuild-
service-role",
    "buildStatus": "IN_PROGRESS",
    "buildComplete": false,
    "projectName": "my-demo-project",
    "timeoutInMinutes": 60,
    "source": {

```

```
    "insecureSsl": false,
    "type": "S3",
    "location": "codebuild-us-west-2-123456789012-input-bucket/my-
source.zip"
  },
  "queuedTimeoutInMinutes": 5,
  "encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3",
  "currentPhase": "QUEUED",
  "startTime": 1556905683.568,
  "environment": {
    "computeType": "BUILD_GENERAL1_MEDIUM",
    "environmentVariables": [],
    "type": "LINUX_CONTAINER",
    "privilegedMode": false,
    "image": "aws/codebuild/standard:1.0",
    "imagePullCredentialsType": "CODEBUILD"
  },
  "phases": [
    {
      "phaseStatus": "SUCCEEDED",
      "startTime": 1556905683.568,
      "phaseType": "SUBMITTED",
      "durationInSeconds": 0,
      "endTime": 1556905684.524
    },
    {
      "startTime": 1556905684.524,
      "phaseType": "QUEUED"
    }
  ],
  "logs": {
    "deepLink": "https://console.aws.amazon.com/cloudwatch/home?region=us-
west-2#logEvent:group=null;stream=null"
  },
  "artifacts": {
    "encryptionDisabled": false,
    "location": "arn:aws:s3:::artifacts-override/my-demo-project",
    "overrideArtifactName": true
  },
  "cache": {
    "type": "NO_CACHE"
  },
  "id": "my-demo-project::12345678-a1b2-c3d4-e5f6-11111EXAMPLE",
  "initiator": "my-aws-account-name",
```

```

    "arn": "arn:aws:codebuild:us-west-2:123456789012:build/my-demo-
project::12345678-a1b2-c3d4-e5f6-11111EXAMPLE"
  }
}

```

For more information, see [Run a Build \(AWS CLI\)](#) in the *AWS CodeBuild User Guide*.

- For API details, see [StartBuild](#) in *AWS CLI Command Reference*.

stop-build-batch

The following code example shows how to use stop-build-batch.

AWS CLI

To stop an in-progress batch build in AWS CodeBuild.

The following stop-build-batch example stops the specified batch build.

```

aws codebuild stop-build-batch \
  --id <project-name>:<batch-ID>

```

Output:

```

{
  "buildBatch": {
    "id": "<project-name>:<batch-ID>",
    "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build-batch/<project-
name>:<batch-ID>",
    "startTime": "2020-10-21T16:54:24.740000+00:00",
    "endTime": "2020-10-21T16:56:05.152000+00:00",
    "currentPhase": "STOPPED",
    "buildBatchStatus": "STOPPED",
    "resolvedSourceVersion": "aef7744ed069c51098e15c360f4102cd2cd1ad64",
    "projectName": "<project-name>",
    "phases": [
      {
        "phaseType": "SUBMITTED",
        "phaseStatus": "SUCCEEDED",
        "startTime": "2020-10-21T16:54:24.740000+00:00",
        "endTime": "2020-10-21T16:54:25.039000+00:00",
        "durationInSeconds": 0
      }
    ],
  },
}

```

```
{
  "phaseType": "DOWNLOAD_BATCHSPEC",
  "phaseStatus": "SUCCEEDED",
  "startTime": "2020-10-21T16:54:25.039000+00:00",
  "endTime": "2020-10-21T16:54:56.583000+00:00",
  "durationInSeconds": 31
},
{
  "phaseType": "IN_PROGRESS",
  "phaseStatus": "STOPPED",
  "startTime": "2020-10-21T16:54:56.583000+00:00",
  "endTime": "2020-10-21T16:56:05.152000+00:00",
  "durationInSeconds": 68
},
{
  "phaseType": "STOPPED",
  "startTime": "2020-10-21T16:56:05.152000+00:00"
}
],
"source": {
  "type": "GITHUB",
  "location": "<GitHub-repo-URL>",
  "gitCloneDepth": 1,
  "gitSubmodulesConfig": {
    "fetchSubmodules": false
  },
  "reportBuildStatus": false,
  "insecureSsl": false
},
"secondarySources": [],
"secondarySourceVersions": [],
"artifacts": {
  "location": ""
},
"secondaryArtifacts": [],
"cache": {
  "type": "NO_CACHE"
},
"environment": {
  "type": "LINUX_CONTAINER",
  "image": "aws/codebuild/amazonlinux2-x86_64-standard:3.0",
  "computeType": "BUILD_GENERAL1_SMALL",
  "environmentVariables": [],
  "privilegedMode": false,
```

```

    "imagePullCredentialsType": "CODEBUILD"
  },
  "logConfig": {
    "cloudWatchLogs": {
      "status": "ENABLED"
    },
    "s3Logs": {
      "status": "DISABLED",
      "encryptionDisabled": false
    }
  },
  "buildTimeoutInMinutes": 60,
  "queuedTimeoutInMinutes": 480,
  "complete": true,
  "initiator": "Strohm",
  "encryptionKey": "arn:aws:kms:<region-ID>:<account-ID>:alias/aws/s3",
  "buildBatchNumber": 3,
  "buildBatchConfig": {
    "serviceRole": "arn:aws:iam::<account-ID>:role/service-role/<project-
name>",
    "restrictions": {
      "maximumBuildsAllowed": 100
    },
    "timeoutInMins": 480
  },
  "buildGroups": [
    {
      "identifier": "DOWNLOAD_SOURCE",
      "ignoreFailure": false,
      "currentBuildSummary": {
        "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build/
<project-name>:<build-ID>",
        "requestedOn": "2020-10-21T16:54:25.468000+00:00",
        "buildStatus": "SUCCEEDED",
        "primaryArtifact": {
          "type": "no_artifacts",
          "identifier": "DOWNLOAD_SOURCE"
        },
        "secondaryArtifacts": []
      }
    },
    {
      "identifier": "linux_small",
      "dependsOn": [],

```

```

        "ignoreFailure": false,
        "currentBuildSummary": {
            "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build/
<project-name>:<build-ID>",
            "requestedOn": "2020-10-21T16:54:56.833000+00:00",
            "buildStatus": "IN_PROGRESS"
        }
    },
    {
        "identifier": "linux_medium",
        "dependsOn": [
            "linux_small"
        ],
        "ignoreFailure": false,
        "currentBuildSummary": {
            "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build/
<project-name>:<build-ID>",
            "requestedOn": "2020-10-21T16:54:56.211000+00:00",
            "buildStatus": "PENDING"
        }
    },
    {
        "identifier": "linux_large",
        "dependsOn": [
            "linux_medium"
        ],
        "ignoreFailure": false,
        "currentBuildSummary": {
            "arn": "arn:aws:codebuild:<region-ID>:<account-ID>:build/
<project-name>:<build-ID>",
            "requestedOn": "2020-10-21T16:54:56.330000+00:00",
            "buildStatus": "PENDING"
        }
    }
]
}
}

```

For more information, see [Batch builds in AWS CodeBuild](#) in the *AWS CodeBuild User Guide*.

- For API details, see [StopBuildBatch](#) in *AWS CLI Command Reference*.

stop-build

The following code example shows how to use stop-build.

AWS CLI

To stop a build of an AWS CodeBuild build project.

The following stop-build example stops the specified CodeBuild build.

```
aws codebuild stop-build --id my-demo-project:12345678-a1b2-c3d4-e5f6-11111EXAMPLE
```

Output:

```
{
  "build": {
    "startTime": 1556906956.318,
    "initiator": "my-aws-account-name",
    "projectName": "my-demo-project",
    "currentPhase": "COMPLETED",
    "cache": {
      "type": "NO_CACHE"
    },
    "source": {
      "insecureSsl": false,
      "location": "codebuild-us-west-2-123456789012-input-bucket/my-source.zip",
      "type": "S3"
    },
    "id": "my-demo-project:1a2b3c4d-5678-90ab-cdef-11111EXAMPLE",
    "endTime": 1556906974.781,
    "phases": [
      {
        "durationInSeconds": 0,
        "phaseType": "SUBMITTED",
        "endTime": 1556906956.935,
        "phaseStatus": "SUCCEEDED",
        "startTime": 1556906956.318
      },
      {
        "durationInSeconds": 1,
        "phaseType": "QUEUED",
        "endTime": 1556906958.272,
        "phaseStatus": "SUCCEEDED",
      }
    ]
  }
}
```

```
    "startTime": 1556906956.935
  },
  {
    "phaseType": "PROVISIONING",
    "phaseStatus": "SUCCEEDED",
    "durationInSeconds": 14,
    "contexts": [
      {
        "message": "",
        "statusCode": ""
      }
    ],
    "endTime": 1556906972.847,
    "startTime": 1556906958.272
  },
  {
    "phaseType": "DOWNLOAD_SOURCE",
    "phaseStatus": "SUCCEEDED",
    "durationInSeconds": 0,
    "contexts": [
      {
        "message": "",
        "statusCode": ""
      }
    ],
    "endTime": 1556906973.552,
    "startTime": 1556906972.847
  },
  {
    "phaseType": "INSTALL",
    "phaseStatus": "SUCCEEDED",
    "durationInSeconds": 0,
    "contexts": [
      {
        "message": "",
        "statusCode": ""
      }
    ],
    "endTime": 1556906973.75,
    "startTime": 1556906973.552
  },
  {
    "phaseType": "PRE_BUILD",
    "phaseStatus": "SUCCEEDED",
```

```
        "durationInSeconds": 0,
        "contexts": [
            {
                "message": "",
                "statusCode": ""
            }
        ],
        "endTime": 1556906973.937,
        "startTime": 1556906973.75
    },
    {
        "durationInSeconds": 0,
        "phaseType": "BUILD",
        "endTime": 1556906974.781,
        "phaseStatus": "STOPPED",
        "startTime": 1556906973.937
    },
    {
        "phaseType": "COMPLETED",
        "startTime": 1556906974.781
    }
],
"artifacts": {
    "location": "arn:aws:s3:::artifacts-override/my-demo-project",
    "encryptionDisabled": false,
    "overrideArtifactName": true
},
"buildComplete": true,
"buildStatus": "STOPPED",
"encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3",
"serviceRole": "arn:aws:iam::123456789012:role/service-role/my-codebuild-
service-role",
"queuedTimeoutInMinutes": 5,
"timeoutInMinutes": 60,
"environment": {
    "type": "LINUX_CONTAINER",
    "environmentVariables": [],
    "computeType": "BUILD_GENERAL1_MEDIUM",
    "privilegedMode": false,
    "image": "aws/codebuild/standard:1.0",
    "imagePullCredentialsType": "CODEBUILD"
},
"logs": {
    "streamName": "1a2b3c4d-5678-90ab-cdef-11111EXAMPLE",
```

```

        "deepLink": "https://console.aws.amazon.com/cloudwatch/home?region=us-
west-2#logEvent:group=/aws/codebuild/my-demo-project;stream=1a2b3c4d-5678-90ab-
cdef-11111EXAMPLE",
        "groupName": "/aws/codebuild/my-demo-project"
    },
    "arn": "arn:aws:codebuild:us-west-2:123456789012:build/my-demo-
project:1a2b3c4d-5678-90ab-cdef-11111EXAMPLE"
}
}

```

For more information, see [Stop a Build \(AWS CLI\)](#) in the *AWS CodeBuild User Guide*.

- For API details, see [StopBuild](#) in *AWS CLI Command Reference*.

update-project

The following code example shows how to use `update-project`.

AWS CLI

To change an AWS CodeBuild build project's settings.

The following `update-project` example changes the settings of the specified CodeBuild build project named `my-demo-project`.

```

aws codebuild update-project --name "my-demo-project" \
  --description "This project is updated" \
  --source "{\"type\": \"S3\", \"location\": \"codebuild-us-west-2-123456789012-
input-bucket/my-source-2.zip\"}" \
  --artifacts "{\"type\": \"S3\", \"location\": \"codebuild-us-west-2-123456789012-
output-bucket-2\"}" \
  --environment "{\"type\": \"LINUX_CONTAINER\", \"image\": \"aws/codebuild/
standard:1.0\", \"computeType\": \"BUILD_GENERAL1_MEDIUM\"}" \
  --service-role "arn:aws:iam::123456789012:role/service-role/my-codebuild-
service-role"

```

The output displays the updated settings.

```

{
  "project": {
    "arn": "arn:aws:codebuild:us-west-2:123456789012:project/my-demo-project",
    "environment": {

```

```

        "privilegedMode": false,
        "environmentVariables": [],
        "type": "LINUX_CONTAINER",
        "image": "aws/codebuild/standard:1.0",
        "computeType": "BUILD_GENERAL1_MEDIUM",
        "imagePullCredentialsType": "CODEBUILD"
    },
    "queuedTimeoutInMinutes": 480,
    "description": "This project is updated",
    "artifacts": {
        "packaging": "NONE",
        "name": "my-demo-project",
        "type": "S3",
        "namespaceType": "NONE",
        "encryptionDisabled": false,
        "location": "codebuild-us-west-2-123456789012-output-bucket-2"
    },
    "encryptionKey": "arn:aws:kms:us-west-2:123456789012:alias/aws/s3",
    "badge": {
        "badgeEnabled": false
    },
    "serviceRole": "arn:aws:iam::123456789012:role/service-role/my-codebuild-
service-role",
    "lastModified": 1556840545.967,
    "tags": [],
    "timeoutInMinutes": 60,
    "created": 1556839783.274,
    "name": "my-demo-project",
    "cache": {
        "type": "NO_CACHE"
    },
    "source": {
        "type": "S3",
        "insecureSsl": false,
        "location": "codebuild-us-west-2-123456789012-input-bucket/my-
source-2.zip"
    }
}
}
}

```

For more information, see [Change a Build Project's Settings \(AWS CLI\)](#) in the *AWS CodeBuild User Guide*

- For API details, see [UpdateProject](#) in *AWS CLI Command Reference*.

update-report-group

The following code example shows how to use `update-report-group`.

AWS CLI

To update a report group in AWS CodeBuild.

The following `update-report-group` example changes the export type of the report group to "NO_EXPORT".

```
aws codebuild update-report-group \  
  --arn arn:aws:codebuild:<region-ID>:<user-ID>:report-group/cli-created-report-  
group \  
  --export-config="exportConfigType=NO_EXPORT"
```

Output:

```
{  
  "reportGroup": {  
    "arn": "arn:aws:codebuild:<region-ID>:<user-ID>:report-group/cli-created-  
report-group",  
    "name": "cli-created-report-group",  
    "type": "TEST",  
    "exportConfig": {  
      "exportConfigType": "NO_EXPORT"  
    },  
    "created": 1602020686.009,  
    "lastModified": 1602021033.454,  
    "tags": []  
  }  
}
```

For more information, see [Working with report groups](#) in the *AWS CodeBuild User Guide*.

- For API details, see [UpdateReportGroup](#) in *AWS CLI Command Reference*.

update-webhook

The following code example shows how to use `update-webhook`.

AWS CLI

To update the webhook for an AWS CodeBuild project

The following `update-webhook` example updates a webhook for the specified CodeBuild project with two filter groups. The `--rotate-secret` parameter specifies that GitHub rotate the project's secret key every time a code change triggers a build. The first filter group specifies pull requests that are created, updated, or reopened on branches with Git reference names that match the regular expression `^refs/heads/master$` and head references that match `^refs/heads/myBranch$`. The second filter group specifies push requests on branches with Git reference names that do not match the regular expression `^refs/heads/myBranch$`.

```
aws codebuild update-webhook \
  --project-name Project2 \
  --rotate-secret \
  --filter-groups "[[{"type":"EVENT","pattern":"PULL_REQUEST_CREATED,
PULL_REQUEST_UPDATED, PULL_REQUEST_REOPENED"}, {"type":"HEAD_REF","pattern
":"^refs/heads/myBranch$"},"excludeMatchedPattern":true}, {"type":"BASE_REF
","pattern":"^refs/heads/master$"},"excludeMatchedPattern":true}], [{"type":
"EVENT","pattern":"PUSH"}, {"type":"HEAD_REF","pattern":"^refs/heads/
myBranch$"},"excludeMatchedPattern":true}]]"
```

Output:

```
{
  "webhook": {
    "filterGroups": [
      [
        {
          "pattern": "PULL_REQUEST_CREATED, PULL_REQUEST_UPDATED,
PULL_REQUEST_REOPENED",
          "type": "EVENT"
        },
        {
          "excludeMatchedPattern": true,
          "pattern": "refs/heads/myBranch$",
          "type": "HEAD_REF"
        },
        {
          "excludeMatchedPattern": true,
          "pattern": "refs/heads/master$",
          "type": "BASE_REF"
        }
      ]
    ]
  }
}
```

```
    }
  ],
  [
    {
      "pattern": "PUSH",
      "type": "EVENT"
    },
    {
      "excludeMatchedPattern": true,
      "pattern": "refs/heads/myBranch$",
      "type": "HEAD_REF"
    }
  ]
],
"lastModifiedSecret": 1556312220.133
}
```

For more information, see [Change a Build Project's Settings \(AWS CLI\)](#) in the *AWS CodeBuild User Guide*

- For API details, see [UpdateWebhook](#) in *AWS CLI Command Reference*.

CodeCommit examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with CodeCommit.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-approval-rule-template-with-repository

The following code example shows how to use `associate-approval-rule-template-with-repository`.

AWS CLI

To associate an approval rule template with a repository

The following `associate-approval-rule-template-with-repository` example associates the specified approval rule template with a repository named `MyDemoRepo`.

```
aws codecommit associate-approval-rule-template-with-repository \  
  --repository-name MyDemoRepo \  
  --approval-rule-template-name 2-approver-rule-for-main
```

This command produces no output.

For more information, see [Associate an Approval Rule Template with a Repository](#) in the *AWS CodeCommit User Guide*.

- For API details, see [AssociateApprovalRuleTemplateWithRepository](#) in *AWS CLI Command Reference*.

batch-associate-approval-rule-template-with-repositories

The following code example shows how to use `batch-associate-approval-rule-template-with-repositories`.

AWS CLI

To associate an approval rule template with multiple repositories in a single operation

The following `batch-associate-approval-rule-template-with-repositories` example associates the specified approval rule template with repositories named `MyDemoRepo` and `MyOtherDemoRepo`.

Note: Approval rule templates are specific to the AWS Region where they are created. They can only be associated with repositories in that AWS Region.

```
aws codecommit batch-associate-approval-rule-template-with-repositories \  
  --repository-names MyDemoRepo, MyOtherDemoRepo \  
  --approval-rule-template-name 2-approver-rule-for-main
```

Output:

```
{  
  "associatedRepositoryNames": [  
    "MyDemoRepo",  
    "MyOtherDemoRepo"  
  ],  
  "errors": []  
}
```

For more information, see [Associate an Approval Rule Template with a Repository](#) in the *AWS CodeCommit User Guide*.

- For API details, see [BatchAssociateApprovalRuleTemplateWithRepositories](#) in *AWS CLI Command Reference*.

batch-describe-merge-conflicts

The following code example shows how to use `batch-describe-merge-conflicts`.

AWS CLI

To get information about merge conflicts in all files or a subset of files in a merge between two commit specifiers

The following `batch-describe-merge-conflicts` example determines the merge conflicts for merging a source branch named `feature-randomizationfeature` with a destination branch named `main` using the `THREE_WAY_MERGE` strategy in a repository named `MyDemoRepo`.

```
aws codecommit batch-describe-merge-conflicts \  
  --source-commit-specifier feature-randomizationfeature \  
  --destination-commit-specifier main \  
  --merge-option THREE_WAY_MERGE \  
  --repository-name MyDemoRepo
```

Output:

```
{
  "conflicts": [
    {
      "conflictMetadata": {
        "filePath": "readme.md",
        "fileSizes": {
          "source": 139,
          "destination": 230,
          "base": 85
        },
        "fileModes": {
          "source": "NORMAL",
          "destination": "NORMAL",
          "base": "NORMAL"
        },
        "objectTypes": {
          "source": "FILE",
          "destination": "FILE",
          "base": "FILE"
        },
        "numberOfConflicts": 1,
        "isBinaryFile": {
          "source": false,
          "destination": false,
          "base": false
        },
        "contentConflict": true,
        "fileModeConflict": false,
        "objectTypeConflict": false,
        "mergeOperations": {
          "source": "M",
          "destination": "M"
        }
      },
      "mergeHunks": [
        {
          "isConflict": true,
          "source": {
            "startLine": 0,
            "endLine": 3,
            "hunkContent": "VGhpcyBpEXAMPLE=="
          }
        }
      ]
    }
  ]
}
```



```

    "MyOtherDemoRepo"
  ],
  "errors": []
}

```

For more information, see [Disassociate an Approval Rule Template](#) in the *AWS CodeCommit User Guide*.

- For API details, see [BatchDisassociateApprovalRuleTemplateFromRepositories](#) in *AWS CLI Command Reference*.

batch-get-commits

The following code example shows how to use `batch-get-commits`.

AWS CLI

To view information about multiple commits

The following `batch-get-commits` example displays details about the specified commits.

```

aws codecommit batch-get-commits \
  --repository-name MyDemoRepo \
  --commit-ids 317f8570EXAMPLE 4c925148EXAMPLE

```

Output:

```

{
  "commits": [
    {
      "additionalData": "",
      "committer": {
        "date": "1508280564 -0800",
        "name": "Mary Major",
        "email": "mary_major@example.com"
      },
      "author": {
        "date": "1508280564 -0800",
        "name": "Mary Major",
        "email": "mary_major@example.com"
      },
      "commitId": "317f8570EXAMPLE",

```

```

    "treeId": "1f330709EXAMPLE",
    "parents": [
      "6e147360EXAMPLE"
    ],
    "message": "Change variable name and add new response element"
  },
  {
    "additionalData": "",
    "committer": {
      "date": "1508280542 -0800",
      "name": "Li Juan",
      "email": "li_juan@example.com"
    },
    "author": {
      "date": "1508280542 -0800",
      "name": "Li Juan",
      "email": "li_juan@example.com"
    },
    "commitId": "4c925148EXAMPLE",
    "treeId": "1f330709EXAMPLE",
    "parents": [
      "317f8570EXAMPLE"
    ],
    "message": "Added new class"
  }
}

```

For more information, see [View Commit Details](#) in the *AWS CodeCommit User Guide*.

- For API details, see [BatchGetCommits](#) in *AWS CLI Command Reference*.

batch-get-repositories

The following code example shows how to use `batch-get-repositories`.

AWS CLI

To view details about multiple repositories

This example shows details about multiple AWS CodeCommit repositories.

```

aws codecommit batch-get-repositories \
  --repository-names MyDemoRepo MyOtherDemoRepo

```

Output:

```
{
  "repositoriesNotFound": [],
  "repositories": [
    {
      "creationDate": 1429203623.625,
      "defaultBranch": "main",
      "repositoryName": "MyDemoRepo",
      "cloneUrlSsh": "ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo",
      "lastModifiedDate": 1430783812.0869999,
      "repositoryDescription": "My demonstration repository",
      "cloneUrlHttp": "https://codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo",
      "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE",
      "Arn": "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
      "accountId": "111111111111"
    },
    {
      "creationDate": 1429203623.627,
      "defaultBranch": "main",
      "repositoryName": "MyOtherDemoRepo",
      "cloneUrlSsh": "ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyOtherDemoRepo",
      "lastModifiedDate": 1430783812.0889999,
      "repositoryDescription": "My other demonstration repository",
      "cloneUrlHttp": "https://codecommit.us-east-2.amazonaws.com/v1/repos/MyOtherDemoRepo",
      "repositoryId": "cfc29ac4-b0cb-44dc-9990-f6f51EXAMPLE",
      "Arn": "arn:aws:codecommit:us-east-2:111111111111:MyOtherDemoRepo",
      "accountId": "111111111111"
    }
  ],
  "repositoriesNotFound": []
}
```

- For API details, see [BatchGetRepositories](#) in *AWS CLI Command Reference*.

create-approval-rule-template

The following code example shows how to use create-approval-rule-template.

AWS CLI

To create an approval rule template

The following `create-approval-rule-template` example creates an approval rule template named `2-approver-rule-for-main`. The template requires two users who assume the role of `CodeCommitReview` to approve any pull request before it can be merged to the main branch.

```
aws codecommit create-approval-rule-template \
  --approval-rule-template-name 2-approver-rule-for-main \
  --approval-rule-template-description "Requires two developers from the team to
  approve the pull request if the destination branch is main" \
  --approval-rule-template-content "{\"Version\": \"2018-11-08\",
  \"DestinationReferences\": [\"refs/heads/main\"],\"Statements\": [{\"Type
  \": \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers\":
  [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}"
```

Output:

```
{
  "approvalRuleTemplate": {
    "approvalRuleTemplateName": "2-approver-rule-for-main",
    "creationDate": 1571356106.936,
    "approvalRuleTemplateId": "dd8b17fe-EXAMPLE",
    "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\",
    \"DestinationReferences\": [\"refs/heads/main\"],\"Statements\": [{\"Type
    \": \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers\":
    [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
    "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
    "approvalRuleTemplateDescription": "Requires two developers from the team to
    approve the pull request if the destination branch is main",
    "lastModifiedDate": 1571356106.936,
    "ruleContentSha256": "4711b576EXAMPLE"
  }
}
```

For more information, see [Create an Approval Rule Template](#) in the *AWS CodeCommit User Guide*.

- For API details, see [CreateApprovalRuleTemplate](#) in *AWS CLI Command Reference*.

create-branch

The following code example shows how to use `create-branch`.

AWS CLI

To create a branch

This example creates a branch in an AWS CodeCommit repository. This command produces output only if there are errors.

Command:

```
aws codecommit create-branch --repository-name MyDemoRepo --branch-name MyNewBranch
--commit-id 317f8570EXAMPLE
```

Output:

```
None.
```

- For API details, see [CreateBranch](#) in *AWS CLI Command Reference*.

create-commit

The following code example shows how to use `create-commit`.

AWS CLI

To create a commit

The following `create-commit` example demonstrates how to create an initial commit for a repository that adds a `readme.md` file to a repository named `MyDemoRepo` in the `main` branch.

```
aws codecommit create-commit \
  --repository-name MyDemoRepo \
  --branch-name main \
  --put-files "filePath=readme.md,fileContent='Welcome to our team repository.'"
```

Output:

```
{
```

```

    "filesAdded": [
      {
        "blobId": "5e1c309d-EXAMPLE",
        "absolutePath": "readme.md",
        "fileMode": "NORMAL"
      }
    ],
    "commitId": "4df8b524-EXAMPLE",
    "treeId": "55b57003-EXAMPLE",
    "filesDeleted": [],
    "filesUpdated": []
  }

```

For more information, see [Create a Commit in AWS CodeCommit](#) in the *AWS CodeCommit User Guide*.

- For API details, see [CreateCommit](#) in *AWS CLI Command Reference*.

create-pull-request-approval-rule

The following code example shows how to use `create-pull-request-approval-rule`.

AWS CLI

To create an approval rule for a pull request

The following `create-pull-request-approval-rule` example creates an approval rule named `Require two approved approvers` for the specified pull request. The rule specifies that two approvals are required from an approval pool. The pool includes all users who access CodeCommit by assuming the role of `CodeCommitReview` in the `123456789012` AWS account. It also includes either an IAM user or federated user named `Nikhil_Jayashankar` from the same AWS account.

```

aws codecommit create-pull-request-approval-rule \
  --approval-rule-name "Require two approved approvers" \
  --approval-rule-content "{\"Version\": \"2018-11-08\", \"Statements\":
  [{\"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers
  \": [\"CodeCommitApprovers:123456789012:Nikhil_Jayashankar\",
  \"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}"

```

Output:

```
{
  "approvalRule": {
    "approvalRuleName": "Require two approved approvers",
    "lastModifiedDate": 1570752871.932,
    "ruleContentSha256": "7c44e6ebEXAMPLE",
    "creationDate": 1570752871.932,
    "approvalRuleId": "aac33506-EXAMPLE",
    "approvalRuleContent": "{\"Version\": \"2018-11-08\", \"Statements\":
  [ { \"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers
  \": [ \"CodeCommitApprovers:123456789012:Nikhil_Jayashankar\",
  \"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\" ] } ] }",
    "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major"
  }
}
```

For more information, see [Create an Approval Rule](#) in the *AWS CodeCommit User Guide*.

- For API details, see [CreatePullRequestApprovalRule](#) in *AWS CLI Command Reference*.

create-pull-request

The following code example shows how to use `create-pull-request`.

AWS CLI

To create a pull request

The following `create-pull-request` example creates a pull request named 'Pronunciation difficulty analyzer' with a description of 'Please review these changes by Tuesday' that targets the 'jane-branch' source branch and is to be merged to the default branch 'main' in an AWS CodeCommit repository named 'MyDemoRepo'.

```
aws codecommit create-pull-request \
  --title "My Pull Request" \
  --description "Please review these changes by Tuesday" \
  --client-request-token 123Example \
  --targets repositoryName=MyDemoRepo,sourceReference=MyNewBranch
```

Output:

```
{
```

```

"pullRequest": {
  "approvalRules": [
    {
      "approvalRuleContent": "{\"Version\": \"2018-11-08\",
\\\"DestinationReferences\": [\\\"refs/heads/main\\\"],\\\"Statements\": [{\\\"Type
\\\": \\\"Approvers\\\",\\\"NumberOfApprovalsNeeded\": 2,\\\"ApprovalPoolMembers\":
[\\\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\\\"]}}]",
      "approvalRuleId": "dd8b17fe-EXAMPLE",
      "approvalRuleName": "2-approver-rule-for-main",
      "creationDate": 1571356106.936,
      "lastModifiedDate": 571356106.936,
      "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
      "originApprovalRuleTemplate": {
        "approvalRuleTemplateId": "dd3d22fe-EXAMPLE",
        "approvalRuleTemplateName": "2-approver-rule-for-main"
      },
      "ruleContentSha256": "4711b576EXAMPLE"
    }
  ],
  "authorArn": "arn:aws:iam::111111111111:user/Jane_Doe",
  "description": "Please review these changes by Tuesday",
  "title": "Pronunciation difficulty analyzer",
  "pullRequestTargets": [
    {
      "destinationCommit": "5d036259EXAMPLE",
      "destinationReference": "refs/heads/main",
      "repositoryName": "MyDemoRepo",
      "sourceCommit": "317f8570EXAMPLE",
      "sourceReference": "refs/heads/jane-branch",
      "mergeMetadata": {
        "isMerged": false
      }
    }
  ],
  "lastActivityDate": 1508962823.285,
  "pullRequestId": "42",
  "clientRequestToken": "123Example",
  "pullRequestStatus": "OPEN",
  "creationDate": 1508962823.285
}
}

```

- For API details, see [CreatePullRequest](#) in *AWS CLI Command Reference*.

create-repository

The following code example shows how to use `create-repository`.

AWS CLI

To create a repository

This example creates a repository and associates it with the user's AWS account.

Command:

```
aws codecommit create-repository --repository-name MyDemoRepo --repository-  
description "My demonstration repository"
```

Output:

```
{  
  "repositoryMetadata": {  
    "repositoryName": "MyDemoRepo",  
    "cloneUrlSsh": "ssh://git-codecommit.us-east-1.amazonaws.com/v1/  
repos/MyDemoRepo",  
    "lastModifiedDate": 1444766838.027,  
    "repositoryDescription": "My demonstration repository",  
    "cloneUrlHttp": "https://git-codecommit.us-east-1.amazonaws.com/v1/  
repos/MyDemoRepo",  
    "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE",  
    "Arn": "arn:aws:codecommit:us-  
east-1:111111111111EXAMPLE:MyDemoRepo",  
    "accountId": "111111111111"  
  }  
}
```

- For API details, see [CreateRepository](#) in *AWS CLI Command Reference*.

create-unreferenced-merge-commit

The following code example shows how to use `create-unreferenced-merge-commit`.

AWS CLI

To create an unreferenced commit that represents the result of merging two commit specifiers

The following `create-unreferenced-merge-commit` example creates a commit that represents the results of a merge between a source branch named `bugfix-1234` with a destination branch named `main` using the `THREE_WAY_MERGE` strategy in a repository named `MyDemoRepo`.

```
aws codecommit create-unreferenced-merge-commit \  
  --source-commit-specifier bugfix-1234 \  
  --destination-commit-specifier main \  
  --merge-option THREE_WAY_MERGE \  
  --repository-name MyDemoRepo \  
  --name "Maria Garcia" \  
  --email "maria_garcia@example.com" \  
  --commit-message "Testing the results of this merge."
```

Output:

```
{  
  "commitId": "4f178133EXAMPLE",  
  "treeId": "389765daEXAMPLE"  
}
```

For more information, see [Resolve Conflicts in a Pull Request](#) in the *AWS CodeCommit User Guide*.

- For API details, see [CreateUnreferencedMergeCommit](#) in *AWS CLI Command Reference*.

credential-helper

The following code example shows how to use `credential-helper`.

AWS CLI

To set up the credential helper included in the AWS CLI with AWS CodeCommit

The `credential-helper` utility is not designed to be called directly from the AWS CLI. Instead it is intended to be used as a parameter with the `git config` command to set up your local computer. It enables Git to use HTTPS and a cryptographically signed version of your IAM user credentials or Amazon EC2 instance role whenever Git needs to authenticate with AWS to interact with CodeCommit repositories.

```
git config --global credential.helper '!aws codecommit credential-helper $@'
```

```
git config --global credential.UseHttpPath true
```

Output:

```
[credential]
  helper = !aws codecommit credential-helper $@
  UseHttpPath = true
```

For more information, see [Setting up for AWS CodeCommit Using Other Methods](#) in the *AWS CodeCommit User Guide*. Review the content carefully, and then follow the procedures in one of the following topics: [For HTTPS Connections on Linux, macOS, or Unix](#) or [For HTTPS Connections on Windows](#) in the *AWS CodeCommit User Guide*.

- For API details, see [CredentialHelper](#) in *AWS CLI Command Reference*.

delete-approval-rule-template

The following code example shows how to use `delete-approval-rule-template`.

AWS CLI

To delete an approval rule template

The following `delete-approval-rule-template` example deletes the specified approval rule template.

```
aws codecommit delete-approval-rule-template \
  --approval-rule-template-name 1-approver-for-all-pull-requests
```

Output:

```
{
  "approvalRuleTemplateId": "41de97b7-EXAMPLE"
}
```

For more information, see [Delete an Approval Rule Template](#) in the *AWS CodeCommit User Guide*.

- For API details, see [DeleteApprovalRuleTemplate](#) in *AWS CLI Command Reference*.

delete-branch

The following code example shows how to use `delete-branch`.

AWS CLI

To delete a branch

This example shows how to delete a branch in an AWS CodeCommit repository.

Command:

```
aws codecommit delete-branch --repository-name MyDemoRepo --branch-name MyNewBranch
```

Output:

```
{
  "branch": {
    "commitId": "317f8570EXAMPLE",
    "branchName": "MyNewBranch"
  }
}
```

- For API details, see [DeleteBranch](#) in *AWS CLI Command Reference*.

delete-comment-content

The following code example shows how to use `delete-comment-content`.

AWS CLI

To delete the content of a comment

You can only delete the content of a comment if you created the comment. This example demonstrates how to delete the content of a comment with the system-generated ID of `ff30b348EXAMPLEb9aa670f`.

```
aws codecommit delete-comment-content \
  --comment-id ff30b348EXAMPLEb9aa670f
```

Output:


```
{
  "comment": {
    "creationDate": 1508369768.142,
    "deleted": true,
    "lastModifiedDate": 1508369842.278,
    "clientRequestToken": "123Example",
    "commentId": "ff30b348EXAMPLEeb9aa670f",
    "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
    "callerReactions": [],
    "reactionCounts":
    {
      "CLAP" : 1
    }
  }
}
```

- For API details, see [DeleteCommentContent](#) in *AWS CLI Command Reference*.

delete-file

The following code example shows how to use `delete-file`.

AWS CLI

To delete a file

The following `delete-file` example demonstrates how to delete a file named `README.md` from a branch named `main` with a most recent commit ID of `c5709475EXAMPLE` in a repository named `MyDemoRepo`.

```
aws codecommit delete-file \
  --repository-name MyDemoRepo \
  --branch-name main \
  --file-path README.md \
  --parent-commit-id c5709475EXAMPLE
```

Output:

```
{
  "blobId": "559b44fEXAMPLE",
  "commitId": "353cf655EXAMPLE",
  "filePath": "README.md",
```

```
"treeId": "6bc824cEXAMPLE"
}
```

For more information, see [Edit or Delete a File in AWS CodeCommit](#) in the *AWS CodeCommit API Reference* guide.

- For API details, see [DeleteFile](#) in *AWS CLI Command Reference*.

delete-pull-request-approval-rule

The following code example shows how to use `delete-pull-request-approval-rule`.

AWS CLI

To delete an approval rule for a pull request

The following `delete-pull-request-approval-rule` example deletes the approval rule named `My Approval Rule` for the specified pull request.

```
aws codecommit delete-pull-request-approval-rule \
  --approval-rule-name "My Approval Rule" \
  --pull-request-id 15
```

Output:

```
{
  "approvalRuleId": "077d8e8a8-EXAMPLE"
}
```

For more information, see [Edit or Delete an Approval Rule](#) in the *AWS CodeCommit User Guide*.

- For API details, see [DeletePullRequestApprovalRule](#) in *AWS CLI Command Reference*.

delete-repository

The following code example shows how to use `delete-repository`.

AWS CLI

To delete a repository

This example shows how to delete an AWS CodeCommit repository.

Command:

```
aws codecommit delete-repository --repository-name MyDemoRepo
```

Output:

```
{
  "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE"
}
```

- For API details, see [DeleteRepository](#) in *AWS CLI Command Reference*.

describe-merge-conflicts

The following code example shows how to use `describe-merge-conflicts`.

AWS CLI**To get detailed information about merge conflicts**

The following `describe-merge-conflicts` example determines the merge conflicts for a file named `readme.md` in the specified source branch and destination branch using the `THREE_WAY_MERGE` strategy.

```
aws codecommit describe-merge-conflicts \
  --source-commit-specifier feature-randomizationfeature \
  --destination-commit-specifier main \
  --merge-option THREE_WAY_MERGE \
  --file-path readme.md \
  --repository-name MyDemoRepo
```

Output:

```
{
  "conflictMetadata": {
    "filePath": "readme.md",
    "fileSizes": {
      "source": 139,
      "destination": 230,
      "base": 85
    }
  },
}
```

```
    "fileModes": {
      "source": "NORMAL",
      "destination": "NORMAL",
      "base": "NORMAL"
    },
    "objectTypes": {
      "source": "FILE",
      "destination": "FILE",
      "base": "FILE"
    },
    "numberOfConflicts": 1,
    "isBinaryFile": {
      "source": false,
      "destination": false,
      "base": false
    },
    "contentConflict": true,
    "fileModeConflict": false,
    "objectTypeConflict": false,
    "mergeOperations": {
      "source": "M",
      "destination": "M"
    }
  },
  "mergeHunks": [
    {
      "isConflict": true,
      "source": {
        "startLine": 0,
        "endLine": 3,
        "hunkContent": "VGhpcyBpEXAMPLE="
      },
      "destination": {
        "startLine": 0,
        "endLine": 1,
        "hunkContent": "VXNlIHRoEXAMPLE="
      }
    }
  ],
  "destinationCommitId": "86958e0aEXAMPLE",
  "sourceCommitId": "6ccd57fdEXAMPLE",
  "baseCommitId": "767b69580EXAMPLE"
}
```

For more information, see [Resolve Conflicts in a Pull Request](#) in the *AWS CodeCommit User Guide*.

- For API details, see [DescribeMergeConflicts](#) in *AWS CLI Command Reference*.

describe-pull-request-events

The following code example shows how to use `describe-pull-request-events`.

AWS CLI

To view events in a pull request

The following `describe-pull-request-events` example retrieves the events for a pull request with the ID of '8'.

```
aws codecommit describe-pull-request-events --pull-request-id 8
```

Output:

```
{
  "pullRequestEvents": [
    {
      "pullRequestId": "8",
      "pullRequestEventType": "PULL_REQUEST_CREATED",
      "eventDate": 1510341779.53,
      "actor": "arn:aws:iam::111111111111:user/Zhang_Wei"
    },
    {
      "pullRequestStatusChangedEventMetadata": {
        "pullRequestStatus": "CLOSED"
      },
      "pullRequestId": "8",
      "pullRequestEventType": "PULL_REQUEST_STATUS_CHANGED",
      "eventDate": 1510341930.72,
      "actor": "arn:aws:iam::111111111111:user/Jane_Doe"
    }
  ]
}
```

- For API details, see [DescribePullRequestEvents](#) in *AWS CLI Command Reference*.

disassociate-approval-rule-template-from-repository

The following code example shows how to use `disassociate-approval-rule-template-from-repository`.

AWS CLI

To disassociate an approval rule template from a repository

The following `disassociate-approval-rule-template-from-repository` example disassociates the specified approval rule template from a repository named `MyDemoRepo`.

```
aws codecommit disassociate-approval-rule-template-from-repository \  
  --repository-name MyDemoRepo \  
  --approval-rule-template-name 1-approver-rule-for-all-pull-requests
```

This command produces no output.

For more information, see [Disassociate an Approval Rule Template](#) in the *AWS CodeCommit User Guide*.

- For API details, see [DisassociateApprovalRuleTemplateFromRepository](#) in *AWS CLI Command Reference*.

evaluate-pull-request-approval-rules

The following code example shows how to use `evaluate-pull-request-approval-rules`.

AWS CLI

To evaluate whether a pull request has all of its approval rules satisfied

The following `evaluate-pull-request-approval-rules` example evaluates the state of approval rules on the specified pull request. In this example, an approval rule has not been satisfied for the pull request, so the output of the command shows an approved value of `false`.

```
aws codecommit evaluate-pull-request-approval-rules \  
  --pull-request-id 27 \  
  --revision-id 9f29d167EXAMPLE
```

Output:

```
{
  "evaluation": {
    "approved": false,
    "approvalRulesNotSatisfied": [
      "Require two approved approvers"
    ],
    "overridden": false,
    "approvalRulesSatisfied": []
  }
}
```

For more information, see [Merge a Pull Request](#) in the *AWS CodeCommit User Guide*.

- For API details, see [EvaluatePullRequestApprovalRules](#) in *AWS CLI Command Reference*.

get-approval-rule-template

The following code example shows how to use `get-approval-rule-template`.

AWS CLI

To get the content of an approval rule template

The following `get-approval-rule-template` example gets the content of an approval rule template named `1-approver-rule-for-all-pull-requests`.

```
aws codecommit get-approval-rule-template \
  --approval-rule-template-name 1-approver-rule-for-all-pull-requests
```

Output:

```
{
  "approvalRuleTemplate": {
    "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\", \"Statements\": [\"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 1, \"ApprovalPoolMembers\": [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]]}\",
    \"ruleContentSha256\": \"621181bbEXAMPLE\",
    \"lastModifiedDate\": 1571356106.936,
    \"creationDate\": 1571356106.936,
    \"approvalRuleTemplateName\": \"1-approver-rule-for-all-pull-requests\",
    \"lastModifiedUser\": \"arn:aws:iam::123456789012:user/Li_Juan\",
    \"approvalRuleTemplateId\": \"a29abb15-EXAMPLE\",
```

```
        "approvalRuleTemplateDescription": "All pull requests must be approved by
one developer on the team."
    }
}
```

For more information, see [Manage Approval Rule Templates](#) in the *AWS CodeCommit User Guide*.

- For API details, see [GetApprovalRuleTemplate](#) in *AWS CLI Command Reference*.

get-blob

The following code example shows how to use `get-blob`.

AWS CLI

To view information about a Git blob object

The following `get-blob` example retrieves information about a Git blob with the ID of '2eb4af3bEXAMPLE' in an AWS CodeCommit repository named 'MyDemoRepo'.

```
aws codecommit get-blob --repository-name MyDemoRepo --blob-id 2eb4af3bEXAMPLE
```

Output:

```
{
  "content": "QSBcCaW5hcnkgTGFyToEXAMPLE="
}
```

- For API details, see [GetBlob](#) in *AWS CLI Command Reference*.

get-branch

The following code example shows how to use `get-branch`.

AWS CLI

To get information about a branch

This example gets information about a branch in an AWS CodeCommit repository.

Command:


```
aws codecommit get-branch --repository-name MyDemoRepo --branch-name MyNewBranch
```

Output:

```
{
  "BranchInfo": {
    "commitID": "317f8570EXAMPLE",
    "branchName": "MyNewBranch"
  }
}
```

- For API details, see [GetBranch](#) in *AWS CLI Command Reference*.

get-comment-reactions

The following code example shows how to use `get-comment-reactions`.

AWS CLI

To view emoji reactions to a comment

The following `get-comment-reactions` example lists all emoji reactions to a comment with the ID of `abcd1234EXAMPLEb5678efgh`. If the font for your shell supports displaying Emoji Version 1.0, then in the output for `emoji` the emoji is displayed.

```
aws codecommit get-comment-reactions \
  --comment-id abcd1234EXAMPLEb5678efgh
```

Output:

```
{
  "reactionsForComment": {
    [
      {
        "reaction": {
          "emoji": "??",
          "shortCode": "thumbsup",
          "unicode": "U+1F44D"
        },
        "users": [
          "arn:aws:iam::123456789012:user/Li_Juan",

```

```

        "arn:aws:iam::123456789012:user/Mary_Major",
        "arn:aws:iam::123456789012:user/Jorge_Souza"
    ]
  },
  {
    "reaction": {
      "emoji": "??",
      "shortCode": "thumbsdown",
      "unicode": "U+1F44E"
    },
    "users": [
      "arn:aws:iam::123456789012:user/Nikhil_Jayashankar"
    ]
  },
  {
    "reaction": {
      "emoji": "??",
      "shortCode": "confused",
      "unicode": "U+1F615"
    },
    "users": [
      "arn:aws:iam::123456789012:user/Saanvi_Sarkar"
    ]
  }
]
}
}

```

For more information, see [Comment on a commit in AWS CodeCommit](#) in the *AWS CodeCommit User Guide*.

- For API details, see [GetCommentReactions](#) in *AWS CLI Command Reference*.

get-comment

The following code example shows how to use `get-comment`.

AWS CLI

To view details of a comment

This example demonstrates how to view details of a comment with the system-generated comment ID of `ff30b348EXAMPLEb9aa670f`.

```
aws codecommit get-comment \  
  --comment-id ff30b348EXAMPLEb9aa670f
```

Output:

```
{  
  "comment": {  
    "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",  
    "clientRequestToken": "123Example",  
    "commentId": "ff30b348EXAMPLEb9aa670f",  
    "content": "Whoops - I meant to add this comment to the line, but I don't  
see how to delete it.",  
    "creationDate": 1508369768.142,  
    "deleted": false,  
    "commentId": "",  
    "lastModifiedDate": 1508369842.278,  
    "callerReactions": [],  
    "reactionCounts":  
    {  
      "SMILE" : 6,  
      "THUMBSUP" : 1  
    }  
  }  
}
```

- For API details, see [GetComment](#) in *AWS CLI Command Reference*.

get-comments-for-compared-commit

The following code example shows how to use `get-comments-for-compared-commit`.

AWS CLI

To view comments on a commit

This example demonstrates how to view view comments made on the comparison between two commits in a repository named `MyDemoRepo`.

```
aws codecommit get-comments-for-compared-commit \  
  --repository-name MyDemoRepo \  
  --before-commit-ID 6e147360EXAMPLE \  
  --after-commit-ID 7e147360EXAMPLE
```

```
--after-commit-id 317f8570EXAMPLE
```

Output:

```
{
  "commentsForComparedCommitData": [
    {
      "afterBlobId": "1f330709EXAMPLE",
      "afterCommitId": "317f8570EXAMPLE",
      "beforeBlobId": "80906a4cEXAMPLE",
      "beforeCommitId": "6e147360EXAMPLE",
      "comments": [
        {
          "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
          "clientRequestToken": "123Example",
          "commentId": "ff30b348EXAMPLEb9aa670f",
          "content": "Whoops - I meant to add this comment to the line,
not the file, but I don't see how to delete it.",
          "creationDate": 1508369768.142,
          "deleted": false,
          "CommentId": "123abc-EXAMPLE",
          "lastModifiedDate": 1508369842.278,
          "callerReactions": [],
          "reactionCounts":
            {
              "SMILE" : 6,
              "THUMBSUP" : 1
            }
        },
        {
          "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
          "clientRequestToken": "123Example",
          "commentId": "553b509bEXAMPLE56198325",
          "content": "Can you add a test case for this?",
          "creationDate": 1508369612.240,
          "deleted": false,
          "commentId": "456def-EXAMPLE",
          "lastModifiedDate": 1508369612.240,
          "callerReactions": [],
          "reactionCounts":
            {
              "THUMBSUP" : 2
            }
        }
      ]
    }
  ]
}
```

```

        }
      ],
      "location": {
        "filePath": "cl_sample.js",
        "filePosition": 1232,
        "relativeFileVersion": "after"
      },
      "repositoryName": "MyDemoRepo"
    }
  ],
  "nextToken": "exampleToken"
}

```

- For API details, see [GetCommentsForComparedCommit](#) in *AWS CLI Command Reference*.

get-comments-for-pull-request

The following code example shows how to use `get-comments-for-pull-request`.

AWS CLI

To view comments for a pull request

This example demonstrates how to view comments for a pull request in a repository named `MyDemoRepo`.

```

aws codecommit get-comments-for-pull-request \
  --repository-name MyDemoRepo \
  --before-commit-ID 317f8570EXAMPLE \
  --after-commit-id 5d036259EXAMPLE

```

Output:

```

{
  "commentsForPullRequestData": [
    {
      "afterBlobId": "1f330709EXAMPLE",
      "afterCommitId": "5d036259EXAMPLE",
      "beforeBlobId": "80906a4cEXAMPLE",
      "beforeCommitId": "317f8570EXAMPLE",
      "comments": [

```

```

    {
      "authorArn": "arn:aws:iam::111111111111:user/Saanvi_Sarkar",
      "clientRequestToken": "",
      "commentId": "abcd1234EXAMPLEb5678efgh",
      "content": "These don't appear to be used anywhere. Can we
remove them?",
      "creationDate": 1508369622.123,
      "deleted": false,
      "lastModifiedDate": 1508369622.123,
      "callerReactions": [],
      "reactionCounts":
      {
        "THUMBSUP" : 6,
        "CONFUSED" : 1
      }
    },
    {
      "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
      "clientRequestToken": "",
      "commentId": "442b498bEXAMPLE5756813",
      "content": "Good catch. I'll remove them.",
      "creationDate": 1508369829.104,
      "deleted": false,
      "lastModifiedDate": 150836912.273,
      "callerReactions": ["THUMBSUP"]
      "reactionCounts":
      {
        "THUMBSUP" : 14
      }
    }
  ],
  "location": {
    "filePath": "ahs_count.py",
    "filePosition": 367,
    "relativeFileVersion": "AFTER"
  },
  "repositoryName": "MyDemoRepo",
  "pullRequestId": "42"
}
"nextToken": "exampleToken"
}

```

- For API details, see [GetCommentsForPullRequest](#) in *AWS CLI Command Reference*.

get-commit

The following code example shows how to use `get-commit`.

AWS CLI

To view information about a commit in a repository

This example shows details about a commit with the system-generated ID of '7e9fd3091thisisanexamplethisisanexample1' in an AWS CodeCommit repository named 'MyDemoRepo'.

Command:

```
aws codecommit get-commit --repository-name MyDemoRepo --commit-id
7e9fd3091thisisanexamplethisisanexample1
```

Output:

```
{
  "commit": {
    "additionalData": "",
    "committer": {
      "date": "1484167798 -0800",
      "name": "Mary Major",
      "email": "mary_major@example.com"
    },
    "author": {
      "date": "1484167798 -0800",
      "name": "Mary Major",
      "email": "mary_major@example.com"
    },
    "treeId": "347a3408thisisanexampletreeidexample",
    "parents": [
      "7aa87a031thisisanexamplethisisanexample1"
    ],
    "message": "Fix incorrect variable name"
  }
}
```

- For API details, see [GetCommit](#) in *AWS CLI Command Reference*.

get-differences

The following code example shows how to use `get-differences`.

AWS CLI

To get information about differences for a commit specifier in a repository

This example shows view metadata information about changes between two commit specifiers (branch, tag, HEAD, or other fully qualified references, such as commit IDs) in a renamed folder in AWS CodeCommit repository named MyDemoRepo. The example includes several options that are not required, including `--before-commit-specifier`, `--before-path`, and `--after-path`, in order to more fully illustrate how you can use these options to limit the results. The response includes file mode permissions.

Command:

```
aws codecommit get-differences --repository-name MyDemoRepo --before-commit-specifier 955bba12thisisanexamplethisisanexample --after-commit-specifier 14a95463thisisanexamplethisisanexample --before-path tmp/example-folder --after-path tmp/renamed-folder
```

Output:

```
{
  "differences": [
    {
      "afterBlob": {
        "path": "blob.txt",
        "blobId": "2eb4af3b1thisisanexamplethisisanexample1",
        "mode": "100644"
      },
      "changeType": "M",
      "beforeBlob": {
        "path": "blob.txt",
        "blobId": "bf7fcf281thisisanexamplethisisanexample1",
        "mode": "100644"
      }
    }
  ]
}
```


- For API details, see [GetDifferences](#) in *AWS CLI Command Reference*.

get-file

The following code example shows how to use `get-file`.

AWS CLI

To get the base-64 encoded contents of a file in an AWS CodeCommit repository

The following `get-file` example demonstrates how to get the base-64 encoded contents of a file named `README.md` from a branch named `main` in a repository named `MyDemoRepo`.

```
aws codecommit get-file \  
  --repository-name MyDemoRepo \  
  --commit-specifier main \  
  --file-path README.md
```

Output:

```
{  
  "blobId": "559b44fEXAMPLE",  
  "commitId": "c5709475EXAMPLE",  
  "fileContent": "IyBQaHVzEXAMPLE",  
  "filePath": "README.md",  
  "fileMode": "NORMAL",  
  "fileSize": 1563  
}
```

For more information, see [GetFile](#) in the *AWS CodeCommit API Reference* guide.

- For API details, see [GetFile](#) in *AWS CLI Command Reference*.

get-folder

The following code example shows how to use `get-folder`.

AWS CLI

To get the contents of a folder in an AWS CodeCommit repository

The following `get-folder` example demonstrates how to get the contents of a top-level folder from a repository named `MyDemoRepo`.

```
aws codecommit get-folder --repository-name MyDemoRepo --folder-path ""
```

Output:

```
{
  "commitId":"c5709475EXAMPLE",
  "files":[
    {
      "absolutePath": ".gitignore",
      "blobId": "74094e8bEXAMPLE",
      "fileMode": "NORMAL",
      "relativePath": ".gitignore"
    },
    {
      "absolutePath": "Gemfile",
      "blobId": "9ceb72f6EXAMPLE",
      "fileMode": "NORMAL",
      "relativePath": "Gemfile"
    },
    {
      "absolutePath": "Gemfile.lock",
      "blobId": "795c4a2aEXAMPLE",
      "fileMode": "NORMAL",
      "relativePath": "Gemfile.lock"
    },
    {
      "absolutePath": "LICENSE.txt",
      "blobId": "0c7932c8EXAMPLE",
      "fileMode": "NORMAL",
      "relativePath": "LICENSE.txt"
    },
    {
      "absolutePath": "README.md",
      "blobId": "559b44feEXAMPLE",
      "fileMode": "NORMAL",
      "relativePath": "README.md"
    }
  ],
  "folderPath": "",
  "subFolders": [
```

```
{
  "absolutePath":"public",
  "relativePath":"public",
  "treeId":"d5e92ae3aEXAMPLE"
},
{
  "absolutePath":"tmp",
  "relativePath":"tmp",
  "treeId":"d564d0bcEXAMPLE"
}
],
"subModules":[],
"symbolicLinks":[],
"treeId":"7b3c4dadEXAMPLE"
}
```

For more information, see `GetFolder` in the *AWS CodeCommit API Reference* guide.

- For API details, see [GetFolder](#) in *AWS CLI Command Reference*.

get-merge-commit

The following code example shows how to use `get-merge-commit`.

AWS CLI

To get detailed information about a merge commit

The following `get-merge-commit` example displays details about a merge commit for the source branch named `bugfix-bug1234` with a destination branch named `main` using the `THREE_WAY_MERGE` strategy in a repository named `MyDemoRepo`.

```
aws codecommit get-merge-commit \
  --source-commit-specifier bugfix-bug1234 \
  --destination-commit-specifier main \
  --merge-option THREE_WAY_MERGE \
  --repository-name MyDemoRepo
```

Output:

```
{
  "sourceCommitId": "c5709475EXAMPLE",
```

```
"destinationCommitId": "317f8570EXAMPLE",
"baseCommitId": "fb12a539EXAMPLE",
"mergeCommitId": "ffc4d608eEXAMPLE"
}
```

For more information, see [View Commit Details](#) in the *AWS CodeCommit User Guide*.

- For API details, see [GetMergeCommit](#) in *AWS CLI Command Reference*.

get-merge-conflicts

The following code example shows how to use `get-merge-conflicts`.

AWS CLI

To view whether there are any merge conflicts for a pull request

The following `get-merge-conflicts` example displays whether there are any merge conflicts between the tip of a source branch named `feature-randomizationfeature` and a destination branch named `'main'` in a repository named `MyDemoRepo`.

```
aws codecommit get-merge-conflicts \
  --repository-name MyDemoRepo \
  --source-commit-specifier feature-randomizationfeature \
  --destination-commit-specifier main \
  --merge-option THREE_WAY_MERGE
```

Output:

```
{
  "mergeable": false,
  "destinationCommitId": "86958e0aEXAMPLE",
  "sourceCommitId": "6ccd57fdEXAMPLE",
  "baseCommitId": "767b6958EXAMPLE",
  "conflictMetadataList": [
    {
      "filePath": "readme.md",
      "fileSizes": {
        "source": 139,
        "destination": 230,
        "base": 85
      }
    }
  ],
}
```

```
    "fileModes": {
      "source": "NORMAL",
      "destination": "NORMAL",
      "base": "NORMAL"
    },
    "objectTypes": {
      "source": "FILE",
      "destination": "FILE",
      "base": "FILE"
    },
    "numberOfConflicts": 1,
    "isBinaryFile": {
      "source": false,
      "destination": false,
      "base": false
    },
    "contentConflict": true,
    "fileModeConflict": false,
    "objectTypeConflict": false,
    "mergeOperations": {
      "source": "M",
      "destination": "M"
    }
  }
]
}
```

- For API details, see [GetMergeConflicts](#) in *AWS CLI Command Reference*.

get-merge-options

The following code example shows how to use `get-merge-options`.

AWS CLI

To get information about the merge options available for merging two specified branches

The following `get-merge-options` example determines the merge options available for merging a source branch named `bugfix-bug1234` with a destination branch named `main` in a repository named `MyDemoRepo`.

```
aws codecommit get-merge-options \
```

```
--source-commit-specifier bugfix-bug1234 \  
--destination-commit-specifier main \  
--repository-name MyDemoRepo
```

Output:

```
{  
  "mergeOptions": [  
    "FAST_FORWARD_MERGE",  
    "SQUASH_MERGE",  
    "THREE_WAY_MERGE"  
  ],  
  "sourceCommitId": "18059494EXAMPLE",  
  "destinationCommitId": "ffd3311dEXAMPLE",  
  "baseCommitId": "ffd3311dEXAMPLE"  
}
```

For more information, see [Resolve Conflicts in a Pull Request](#) in the *AWS CodeCommit User Guide*.

- For API details, see [GetMergeOptions](#) in *AWS CLI Command Reference*.

get-pull-request-approval-states

The following code example shows how to use `get-pull-request-approval-states`.

AWS CLI

To view approvals on a pull request

The following `get-pull-request-approval-states` example returns approvals for the specified pull request.

```
aws codecommit get-pull-request-approval-states \  
  --pull-request-id 8 \  
  --revision-id 9f29d167EXAMPLE
```

Output:

```
{  
  "approvals": [  
    {
```

```
        "userArn": "arn:aws:iam::123456789012:user/Mary_Major",
        "approvalState": "APPROVE"
    }
]
}
```

For more information, see [View Pull Requests](#) in the *AWS CodeCommit User Guide*.

- For API details, see [GetPullRequestApprovalStates](#) in *AWS CLI Command Reference*.

get-pull-request-override-state

The following code example shows how to use `get-pull-request-override-state`.

AWS CLI

To get information about the override status of a pull request

The following `get-pull-request-override-state` example returns the override state for the specified pull request. In this example, the approval rules for the pull request were overridden by a user named Mary Major, so the output returns a value of `true`:

```
aws codecommit get-pull-request-override-state \
  --pull-request-id 34 \
  --revision-id 9f29d167EXAMPLE
```

Output:

```
{
  "overridden": true,
  "overrider": "arn:aws:iam::123456789012:user/Mary_Major"
}
```

For more information, see [Override Approval Rules on a Pull Request](#) in the *AWS CodeCommit User Guide*.

- For API details, see [GetPullRequestOverrideState](#) in *AWS CLI Command Reference*.

get-pull-request

The following code example shows how to use `get-pull-request`.

AWS CLI

To view details of a pull request

This example demonstrates how to view information about a pull request with the ID of 27.

```
aws codecommit get-pull-request \  
  --pull-request-id 27
```

Output:

```
{  
  "pullRequest": {  
    "approvalRules": [  
      {  
        "approvalRuleContent": "{\"Version\": \"2018-11-08\", \"Statements\":  
[{\n\"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers\":  
[\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",  
        "approvalRuleId": "dd8b17fe-EXAMPLE",  
        "approvalRuleName": "2-approver-rule-for-main",  
        "creationDate": 1571356106.936,  
        "lastModifiedDate": 571356106.936,  
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",  
        "ruleContentSha256": "4711b576EXAMPLE"  
      }  
    ],  
    "lastActivityDate": 1562619583.565,  
    "pullRequestTargets": [  
      {  
        "sourceCommit": "ca45e279EXAMPLE",  
        "sourceReference": "refs/heads/bugfix-1234",  
        "mergeBase": "a99f5ddbEXAMPLE",  
        "destinationReference": "refs/heads/main",  
        "mergeMetadata": {  
          "isMerged": false  
        },  
        "destinationCommit": "2abfc6beEXAMPLE",  
        "repositoryName": "MyDemoRepo"  
      }  
    ],  
    "revisionId": "e47def21EXAMPLE",  
    "title": "Quick fix for bug 1234",  
    "authorArn": "arn:aws:iam::123456789012:user/Nikhil_Jayashankar",
```



```
    "clientRequestToken": "d8d7612e-EXAMPLE",
    "creationDate": 1562619583.565,
    "pullRequestId": "27",
    "pullRequestStatus": "OPEN"
  }
}
```

- For API details, see [GetPullRequest](#) in *AWS CLI Command Reference*.

get-repository-triggers

The following code example shows how to use `get-repository-triggers`.

AWS CLI

To get information about triggers in a repository

This example shows details about triggers configured for an AWS CodeCommit repository named `MyDemoRepo`.

```
aws codecommit get-repository-triggers \
  --repository-name MyDemoRepo
```

Output:

```
{
  "configurationId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE",
  "triggers": [
    {
      "destinationArn": "arn:aws:sns:us-
east-1:111111111111:MyCodeCommitTopic",
      "branches": [
        "main",
        "preprod"
      ],
      "name": "MyFirstTrigger",
      "customData": "",
      "events": [
        "all"
      ]
    },
    {
```

```

        "destinationArn": "arn:aws:lambda:us-
east-1:111111111111:function:MyCodeCommitPythonFunction",
        "branches": [],
        "name": "MySecondTrigger",
        "customData": "EXAMPLE",
        "events": [
            "all"
        ]
    }
]
}

```

- For API details, see [GetRepositoryTriggers](#) in *AWS CLI Command Reference*.

get-repository

The following code example shows how to use `get-repository`.

AWS CLI

To get information about a repository

This example shows details about an AWS CodeCommit repository.

```

aws codecommit get-repository \
  --repository-name MyDemoRepo

```

Output:

```

{
  "repositoryMetadata": {
    "creationDate": 1429203623.625,
    "defaultBranch": "main",
    "repositoryName": "MyDemoRepo",
    "cloneUrlSsh": "ssh://git-codecommit.us-east-1.amazonaws.com/v1/repos/v1/
repos/MyDemoRepo",
    "lastModifiedDate": 1430783812.0869999,
    "repositoryDescription": "My demonstration repository",
    "cloneUrlHttp": "https://codecommit.us-east-1.amazonaws.com/v1/repos/
MyDemoRepo",
    "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE",
    "Arn": "arn:aws:codecommit:us-east-1:80398EXAMPLE:MyDemoRepo"
  }
}

```

```
    "accountId": "111111111111"  
  }  
}
```

- For API details, see [GetRepository](#) in *AWS CLI Command Reference*.

list-approval-rule-templates

The following code example shows how to use `list-approval-rule-templates`.

AWS CLI

To list all approval rule templates in an AWS Region

The following `list-approval-rule-templates` example lists all approval rule templates in the specified Region. If no AWS Region is specified as a parameter, the command returns approval rule templates for the region specified in the AWS CLI profile used to run the command.

```
aws codecommit list-approval-rule-templates \  
  --region us-east-2
```

Output:

```
{  
  "approvalRuleTemplateName": [  
    "2-approver-rule-for-main",  
    "1-approver-rule-for-all-pull-requests"  
  ]  
}
```

For more information, see [Manage Approval Rule Templates](#) in the *AWS CodeCommit User Guide*.

- For API details, see [ListApprovalRuleTemplates](#) in *AWS CLI Command Reference*.

list-associated-approval-rule-templates-for-repository

The following code example shows how to use `list-associated-approval-rule-templates-for-repository`.

AWS CLI

To list all templates associated with a repository

The following `list-associated-approval-rule-templates-for-repository` example lists all approval rule templates associated with a repository named `MyDemoRepo`.

```
aws codecommit list-associated-approval-rule-templates-for-repository \  
  --repository-name MyDemoRepo
```

Output:

```
{  
  "approvalRuleTemplateName": [  
    "2-approver-rule-for-main",  
    "1-approver-rule-for-all-pull-requests"  
  ]  
}
```

For more information, see [Manage Approval Rule Templates](#) in the *AWS CodeCommit User Guide*.

- For API details, see [ListAssociatedApprovalRuleTemplatesForRepository](#) in *AWS CLI Command Reference*.

list-branches

The following code example shows how to use `list-branches`.

AWS CLI

To view a list of branch names

This example lists all branch names in an AWS CodeCommit repository.

```
aws codecommit list-branches \  
  --repository-name MyDemoRepo
```

Output:

```
{
```

```
"branches": [  
  "MyNewBranch",  
  "main"  
]  
}
```

- For API details, see [ListBranches](#) in *AWS CLI Command Reference*.

list-pull-requests

The following code example shows how to use `list-pull-requests`.

AWS CLI

To view a list of pull requests in a repository

This example demonstrates how to list pull requests created by an IAM user with the ARN 'arn:aws:iam::111111111111:user/Li_Juan' and the status of 'CLOSED' in an AWS CodeCommit repository named 'MyDemoRepo':

```
aws codecommit list-pull-requests --author-arn arn:aws:iam::111111111111:user/  
Li_Juan --pull-request-status CLOSED --repository-name MyDemoRepo
```

Output:

```
{  
  "nextToken": "",  
  "pullRequestIds": ["2", "12", "16", "22", "23", "35", "30", "39", "47"]  
}
```

- For API details, see [ListPullRequests](#) in *AWS CLI Command Reference*.

list-repositories-for-approval-rule-template

The following code example shows how to use `list-repositories-for-approval-rule-template`.

AWS CLI

To list all repositories associated with a template

The following `list-repositories-for-approval-rule-template` example lists all repositories associated with the specified approval rule template.

```
aws codecommit list-repositories-for-approval-rule-template \  
  --approval-rule-template-name 2-approver-rule-for-main
```

Output:

```
{  
  "repositoryNames": [  
    "MyDemoRepo",  
    "MyClonedRepo"  
  ]  
}
```

For more information, see [Manage Approval Rule Templates](#) in the *AWS CodeCommit User Guide*.

- For API details, see [ListRepositoriesForApprovalRuleTemplate](#) in *AWS CLI Command Reference*.

list-repositories

The following code example shows how to use `list-repositories`.

AWS CLI

To view a list of repositories

This example lists all AWS CodeCommit repositories associated with the user's AWS account.

Command:

```
aws codecommit list-repositories
```

Output:

```
{  
  "repositories": [  
    {  
      "repositoryName": "MyDemoRepo"  
    }  
  ]  
}
```

```
    "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE",
  },
  {
    "repositoryName": "MyOtherDemoRepo"
    "repositoryId": "cfc29ac4-b0cb-44dc-9990-f6f51EXAMPLE"
  }
]
}
```

- For API details, see [ListRepositories](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To view the AWS tags for a repository

The following `list-tags-for-resource` example lists tag keys and tag values for the specified repository.

```
aws codecommit list-tags-for-resource \
  --resource-arn arn:aws:codecommit:us-west-2:111111111111:MyDemoRepo
```

Output:

```
{
  "tags": {
    "Status": "Secret",
    "Team": "Saanvi"
  }
}
```

For more information, see [View Tags for a Repository](#) in the *AWS CodeCommit User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

merge-branches-by-fast-forward

The following code example shows how to use `merge-branches-by-fast-forward`.

AWS CLI

To merge two branches using the fast-forward merge strategy

The following `merge-branches-by-fast-forward` example merges the specified source branch with the specified destination branch in a repository named `MyDemoRepo`.

```
aws codecommit merge-branches-by-fast-forward \  
  --source-commit-specifier bugfix-bug1234 \  
  --destination-commit-specifier bugfix-bug1233 \  
  --repository-name MyDemoRepo
```

Output:

```
{  
  "commitId": "4f178133EXAMPLE",  
  "treeId": "389765daEXAMPLE"  
}
```

For more information, see [Compare and Merge Branches](#) in the *AWS CodeCommit User Guide*.

- For API details, see [MergeBranchesByFastForward](#) in *AWS CLI Command Reference*.

`merge-branches-by-squash`

The following code example shows how to use `merge-branches-by-squash`.

AWS CLI

To merge two branches using the squash merge strategy

The following `merge-branches-by-squash` example merges the specified source branch with the specified destination branch in a repository named `MyDemoRepo`.

```
aws codecommit merge-branches-by-squash \  
  --source-commit-specifier bugfix-bug1234 \  
  --destination-commit-specifier bugfix-bug1233 \  
  --author-name "Maria Garcia" \  
  --email "maria_garcia@example.com" \  
  --commit-message "Merging two fix branches to prepare for a general patch." \  
  --repository-name MyDemoRepo
```


Output:

```
{
  "commitId": "4f178133EXAMPLE",
  "treeId": "389765daEXAMPLE"
}
```

For more information, see [Compare and Merge Branches](#) in the *AWS CodeCommit User Guide*.

- For API details, see [MergeBranchesBySquash](#) in *AWS CLI Command Reference*.

merge-branches-by-three-way

The following code example shows how to use `merge-branches-by-three-way`.

AWS CLI**To merge two branches using the three-way merge strategy**

The following `merge-branches-by-three-way` example merges the specified source branch with the specified destination branch in a repository named `MyDemoRepo`.

```
aws codecommit merge-branches-by-three-way \
  --source-commit-specifier main \
  --destination-commit-specifier bugfix-bug1234 \
  --author-name "Jorge Souza" --email "jorge_souza@example.com" \
  --commit-message "Merging changes from main to bugfix branch before additional
testing." \
  --repository-name MyDemoRepo
```

Output:

```
{
  "commitId": "4f178133EXAMPLE",
  "treeId": "389765daEXAMPLE"
}
```

For more information, see [Compare and Merge Branches](#) in the *AWS CodeCommit User Guide*.

- For API details, see [MergeBranchesByThreeWay](#) in *AWS CLI Command Reference*.

merge-pull-request-by-fast-forward

The following code example shows how to use merge-pull-request-by-fast-forward.

AWS CLI

To merge and close a pull request

This example demonstrates how to merge and close a pull request with the ID of '47' and a source commit ID of '99132ab0EXAMPLE' in a repository named MyDemoRepo.

```
aws codecommit merge-pull-request-by-fast-forward \  
  --pull-request-id 47 \  
  --source-commit-id 99132ab0EXAMPLE \  
  --repository-name MyDemoRepo
```

Output:

```
{  
  "pullRequest": {  
    "approvalRules": [  
      {  
        "approvalRuleContent": "{\"Version\": \"2018-11-08\", \"Statements\":  
[{\n\"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 1, \"ApprovalPoolMembers\":  
[\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]]\"",  
        "approvalRuleId": "dd8b17fe-EXAMPLE",  
        "approvalRuleName": "I want one approver for this pull request",  
        "creationDate": 1571356106.936,  
        "lastModifiedDate": 571356106.936,  
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",  
        "ruleContentSha256": "4711b576EXAMPLE"  
      }  
    ],  
    "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",  
    "clientRequestToken": "",  
    "creationDate": 1508530823.142,  
    "description": "Review the latest changes and updates to the global  
variables",  
    "lastActivityDate": 1508887223.155,  
    "pullRequestId": "47",  
    "pullRequestStatus": "CLOSED",  
    "pullRequestTargets": [  
      {
```

```

    "destinationCommit": "9f31c968EXAMPLE",
    "destinationReference": "refs/heads/main",
    "mergeMetadata": {
      "isMerged": true,
      "mergedBy": "arn:aws:iam::123456789012:user/Mary_Major"
    },
    "repositoryName": "MyDemoRepo",
    "sourceCommit": "99132ab0EXAMPLE",
    "sourceReference": "refs/heads/variables-branch"
  }
],
"title": "Consolidation of global variables"
}
}

```

For more information, see [Merge a Pull Request](#) in the *AWS CodeCommit User Guide*.

- For API details, see [MergePullRequestByFastForward](#) in *AWS CLI Command Reference*.

merge-pull-request-by-squash

The following code example shows how to use `merge-pull-request-by-squash`.

AWS CLI

To merge a pull request using the squash merge strategy

The following `merge-pull-request-by-squash` example merges and closes the specified pull request using the conflict resolution strategy of `ACCEPT_SOURCE` in a repository named `MyDemoRepo`.

```

aws codecommit merge-pull-request-by-squash \
  --pull-request-id 47 \
  --source-commit-id 99132ab0EXAMPLE \
  --repository-name MyDemoRepo \
  --conflict-detail-level LINE_LEVEL \
  --conflict-resolution-strategy ACCEPT_SOURCE \
  --name "Jorge Souza" --email "jorge_souza@example.com" \
  --commit-message "Merging pull request 47 by squash and accepting source in
merge conflicts"

```

Output:

```

{
  "pullRequest": {
    "approvalRules": [
      {
        "approvalRuleContent": "{\"Version\": \"2018-11-08\",
\\\"DestinationReferences\": [\\\"refs/heads/main\\\"],\\\"Statements\": [{\\\"Type
\\\": \\\"Approvers\\\",\\\"NumberOfApprovalsNeeded\": 2,\\\"ApprovalPoolMembers\\\":
[\\\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\\\"]}}]",
        "approvalRuleId": "dd8b17fe-EXAMPLE",
        "approvalRuleName": "2-approver-rule-for-main",
        "creationDate": 1571356106.936,
        "lastModifiedDate": 571356106.936,
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
        "originApprovalRuleTemplate": {
          "approvalRuleTemplateId": "dd8b17fe-EXAMPLE",
          "approvalRuleTemplateName": "2-approver-rule-for-main"
        },
        "ruleContentSha256": "4711b576EXAMPLE"
      }
    ],
    "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",
    "clientRequestToken": "",
    "creationDate": 1508530823.142,
    "description": "Review the latest changes and updates to the global
variables",
    "lastActivityDate": 1508887223.155,
    "pullRequestId": "47",
    "pullRequestStatus": "CLOSED",
    "pullRequestTargets": [
      {
        "destinationCommit": "9f31c968EXAMPLE",
        "destinationReference": "refs/heads/main",
        "mergeMetadata": {
          "isMerged": true,
          "mergedBy": "arn:aws:iam::123456789012:user/Mary_Major"
        },
        "repositoryName": "MyDemoRepo",
        "sourceCommit": "99132ab0EXAMPLE",
        "sourceReference": "refs/heads/variables-branch"
      }
    ],
    "title": "Consolidation of global variables"
  }
}

```

```
}

```

For more information, see [Merge a Pull Request](#) in the *AWS CodeCommit User Guide*.

- For API details, see [MergePullRequestBySquash](#) in *AWS CLI Command Reference*.

merge-pull-request-by-three-way

The following code example shows how to use `merge-pull-request-by-three-way`.

AWS CLI

To merge a pull request using the three-way merge strategy

The following `merge-pull-request-by-three-way` example merges and closes the specified pull request using the default options for conflict detail and conflict resolution strategy in a repository named `MyDemoRepo`.

```
aws codecommit merge-pull-request-by-three-way \
  --pull-request-id 47 \
  --source-commit-id 99132ab0EXAMPLE \
  --repository-name MyDemoRepo \
  --name "Maria Garcia" \
  --email "maria_garcia@example.com" \
  --commit-message "Merging pull request 47 by three-way with default options"
```

Output:

```
{
  "pullRequest": {
    "approvalRules": [
      {
        "approvalRuleContent": "{\"Version\": \"2018-11-08\",
        \"DestinationReferences\": [\"refs/heads/main\"], \"Statements\": [{\"Type
        \": \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers\":
        [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
        "approvalRuleId": "dd8b17fe-EXAMPLE",
        "approvalRuleName": "2-approver-rule-for-main",
        "creationDate": 1571356106.936,
        "lastModifiedDate": 571356106.936,
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
```

```

        "originApprovalRuleTemplate": {
            "approvalRuleTemplateId": "dd8b17fe-EXAMPLE",
            "approvalRuleTemplateName": "2-approver-rule-for-main"
        },
        "ruleContentSha256": "4711b576EXAMPLE"
    }
],
"authorArn": "arn:aws:iam::123456789012:user/Li_Juan",
"clientRequestToken": "",
"creationDate": 1508530823.142,
"description": "Review the latest changes and updates to the global
variables",
"lastActivityDate": 1508887223.155,
"pullRequestId": "47",
"pullRequestStatus": "CLOSED",
"pullRequestTargets": [
    {
        "destinationCommit": "9f31c968EXAMPLE",
        "destinationReference": "refs/heads/main",
        "mergeMetadata": {
            "isMerged": true,
            "mergedBy": "arn:aws:iam::123456789012:user/Mary_Major"
        },
        "repositoryName": "MyDemoRepo",
        "sourceCommit": "99132ab0EXAMPLE",
        "sourceReference": "refs/heads/variables-branch"
    }
],
"title": "Consolidation of global variables"
}
}

```

For more information, see [Merge a Pull Request](#) in the *AWS CodeCommit User Guide*.

- For API details, see [MergePullRequestByThreeWay](#) in *AWS CLI Command Reference*.

override-pull-request-approval-rules

The following code example shows how to use `override-pull-request-approval-rules`.

AWS CLI

To override approval rule requirements on a pull request

The following `override-pull-request-approval-rules` example overrides approval rules on the specified pull request. To revoke an override instead, set the `--override-status` parameter value to `REVOKE`.

```
aws codecommit override-pull-request-approval-rules \  
  --pull-request-id 34 \  
  --revision-id 927df8d8EXAMPLE \  
  --override-status OVERRIDE
```

This command produces no output.

For more information, see [Override Approval Rules on a Pull Request](#) in the *AWS CodeCommit User Guide*.

- For API details, see [OverridePullRequestApprovalRules](#) in *AWS CLI Command Reference*.

post-comment-for-compared-commit

The following code example shows how to use `post-comment-for-compared-commit`.

AWS CLI

To create a comment on a commit

This example demonstrates how to add the comment "Can you add a test case for this?" on the change to the `cl_sample.js` file in the comparison between two commits in a repository named `MyDemoRepo`.

```
aws codecommit post-comment-for-compared-commit \  
  --repository-name MyDemoRepo \  
  --before-commit-id 317f8570EXAMPLE \  
  --after-commit-id 5d036259EXAMPLE \  
  --client-request-token 123Example \  
  --content "Can you add a test case for this?" \  
  --location filePath=cl_sample.js,filePosition=1232,relativeFileVersion=AFTER
```

Output:

```
{  
  "afterBlobId": "1f330709EXAMPLE",  
  "afterCommitId": "317f8570EXAMPLE",
```

```

"beforeBlobId": "80906a4cEXAMPLE",
"beforeCommitId": "6e147360EXAMPLE",
"comment": {
  "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
  "clientRequestToken": "",
  "commentId": "553b509bEXAMPLE56198325",
  "content": "Can you add a test case for this?",
  "creationDate": 1508369612.203,
  "deleted": false,
  "commentId": "abc123-EXAMPLE",
  "lastModifiedDate": 1508369612.203,
  "callerReactions": [],
  "reactionCounts": []
},
"location": {
  "filePath": "cl_sample.js",
  "filePosition": 1232,
  "relativeFileVersion": "AFTER"
,
"repositoryName": "MyDemoRepo"
}
}

```

- For API details, see [PostCommentForComparedCommit](#) in *AWS CLI Command Reference*.

post-comment-for-pull-request

The following code example shows how to use `post-comment-for-pull-request`.

AWS CLI

To add a comment to a pull request

The following `post-comment-for-pull-request` example adds the comment "These don't appear to be used anywhere. Can we remove them?" on the change to the `ahs_count.py` file in a pull request with the ID of 47 in a repository named `MyDemoRepo`.

```

aws codecommit post-comment-for-pull-request \
  --pull-request-id "47" \
  --repository-name MyDemoRepo \
  --before-commit-id 317f8570EXAMPLE \
  --after-commit-id 5d036259EXAMPLE \

```



```
--client-request-token 123Example \  
--content "These don't appear to be used anywhere. Can we remove them?" \  
--location filePath=ahs_count.py,filePosition=367,relativeFileVersion=AFTER
```

Output:

```
{  
  "afterBlobId": "1f330709EXAMPLE",  
  "afterCommitId": "5d036259EXAMPLE",  
  "beforeBlobId": "80906a4cEXAMPLE",  
  "beforeCommitId": "317f8570EXAMPLE",  
  "comment": {  
    "authorArn": "arn:aws:iam::111111111111:user/Saanvi_Sarkar",  
    "clientRequestToken": "123Example",  
    "commentId": "abcd1234EXAMPLEeb5678efgh",  
    "content": "These don't appear to be used anywhere. Can we remove  
them?",  
    "creationDate": 1508369622.123,  
    "deleted": false,  
    "CommentId": "",  
    "lastModifiedDate": 1508369622.123,  
    "callerReactions": [],  
    "reactionCounts": []  
  },  
  "location": {  
    "filePath": "ahs_count.py",  
    "filePosition": 367,  
    "relativeFileVersion": "AFTER"  
  },  
  "repositoryName": "MyDemoRepo",  
  "pullRequestId": "47"  
}
```

- For API details, see [PostCommentForPullRequest](#) in *AWS CLI Command Reference*.

post-comment-reply

The following code example shows how to use `post-comment-reply`.

AWS CLI**To reply to a comment on a commit or in a pull request**

This example demonstrates how to add the reply "Good catch. I'll remove them." to the comment with the system-generated ID of abcd1234EXAMPLEb5678efgh.

```
aws codecommit post-comment-reply \  
  --in-reply-to abcd1234EXAMPLEb5678efgh \  
  --content "Good catch. I'll remove them." \  
  --client-request-token 123Example
```

Output:

```
{  
  "comment": {  
    "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",  
    "clientRequestToken": "123Example",  
    "commentId": "442b498bEXAMPLE5756813",  
    "content": "Good catch. I'll remove them.",  
    "creationDate": 1508369829.136,  
    "deleted": false,  
    "CommentId": "abcd1234EXAMPLEb5678efgh",  
    "lastModifiedDate": 150836912.221,  
    "callerReactions": [],  
    "reactionCounts": []  
  }  
}
```

- For API details, see [PostCommentReply](#) in *AWS CLI Command Reference*.

put-comment-reaction

The following code example shows how to use `put-comment-reaction`.

AWS CLI

To reply to a comment on a commit with an emoji

The following `put-comment-reaction` example replies to a comment with the ID of abcd1234EXAMPLEb5678efgh with an emoji reaction value of `:thumbsup:`.

```
aws codecommit put-comment-reaction \  
  --comment-id abcd1234EXAMPLEb5678efgh \  
  --reaction-value :thumbsup:
```

This command produces no output.

For more information, see [Comment on a commit in AWS CodeCommit](#) in the *AWS CodeCommit User Guide*.

- For API details, see [PutCommentReaction](#) in *AWS CLI Command Reference*.

put-file

The following code example shows how to use `put-file`.

AWS CLI

To add a file to a repository

The following `put-file` example adds a file named 'ExampleSolution.py' to a repository named 'MyDemoRepo' to a branch named 'feature-randomizationfeature' whose most recent commit has an ID of '4c925148EXAMPLE'.

```
aws codecommit put-file \  
  --repository-name MyDemoRepo \  
  --branch-name feature-randomizationfeature \  
  --file-content file://MyDirectory/ExampleSolution.py \  
  --file-path /solutions/ExampleSolution.py \  
  --parent-commit-id 4c925148EXAMPLE \  
  --name "Maria Garcia" \  
  --email "maria_garcia@example.com" \  
  --commit-message "I added a third randomization routine."
```

Output:

```
{  
  "blobId": "2eb4af3bEXAMPLE",  
  "commitId": "317f8570EXAMPLE",  
  "treeId": "347a3408EXAMPLE"  
}
```

- For API details, see [PutFile](#) in *AWS CLI Command Reference*.

put-repository-triggers

The following code example shows how to use `put-repository-triggers`.

AWS CLI

To add or update a trigger in a repository

This example demonstrates how to update triggers named 'MyFirstTrigger' and 'MySecondTrigger' using an already-created JSON file (here named MyTriggers.json) that contains the structure of all the triggers for a repository named MyDemoRepo. To learn how to get the JSON for existing triggers, see the `get-repository-triggers` command.

```
aws codecommit put-repository-triggers \  
  --repository-name MyDemoRepo file://MyTriggers.json
```

Contents of MyTriggers.json:

```
{  
  "repositoryName": "MyDemoRepo",  
  "triggers": [  
    {  
      "destinationArn": "arn:aws:sns:us-  
east-1:80398EXAMPLE:MyCodeCommitTopic",  
      "branches": [  
        "main",  
        "preprod"  
      ],  
      "name": "MyFirstTrigger",  
      "customData": "",  
      "events": [  
        "all"  
      ]  
    },  
    {  
      "destinationArn": "arn:aws:lambda:us-  
east-1:111111111111:function:MyCodeCommitPythonFunction",  
      "branches": [],  
      "name": "MySecondTrigger",  
      "customData": "EXAMPLE",  
      "events": [  
        "all"  
      ]  
    }  
  ]  
}
```

Output:

```
{
  "configurationId": "6fa51cd8-35c1-EXAMPLE"
}
```

- For API details, see [PutRepositoryTriggers](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI**To add AWS tags to an existing repository**

The following `tag-resource` example tags the specified repository with two tags.

```
aws codecommit tag-resource \
  --resource-arn arn:aws:codecommit:us-west-2:111111111111:MyDemoRepo \
  --tags Status=Secret,Team=Saarvi
```

This command produces no output.

For more information, see [Add a Tag to a Repository](#) in the *AWS CodeCommit User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

test-repository-triggers

The following code example shows how to use `test-repository-triggers`.

AWS CLI**To test triggers in a repository**

This example demonstrates how to test a trigger named 'MyFirstTrigger' in an AWS CodeCommit repository named MyDemoRepo. In this example, events in the repository trigger notifications from an Amazon Simple Notification Service (Amazon SNS) topic.

Command:

```
aws codecommit test-repository-triggers --repository-name MyDemoRepo
--triggers name=MyFirstTrigger,destinationArn=arn:aws:sns:us-
east-1:111111111111:MyCodeCommitTopic,branches=mainline,preprod,events=all
```

Output:

```
{
  "successfulExecutions": [
    "MyFirstTrigger"
  ],
  "failedExecutions": []
}
```

- For API details, see [TestRepositoryTriggers](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove AWS tags from a repository

The following `untag-resource` example removes the tag with the specified key from the repository named `MyDemoRepo`.

```
aws codecommit untag-resource \
--resource-arn arn:aws:codecommit:us-west-2:111111111111:MyDemoRepo \
--tag-keys Status
```

This command produces no output.

For more information, see [Remove a Tag from a Repository](#) in the *AWS CodeCommit User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-approval-rule-template-content

The following code example shows how to use `update-approval-rule-template-content`.

AWS CLI

To update the content of an approval rule template

The following `update-approval-rule-template-content` example changes the content of the specified approval rule template to redefine the approval pool to users who assume the role of `CodeCommitReview`.

```
aws codecommit update-approval-rule-template-content \
  --approval-rule-template-name 1-approver-rule \
  --new-rule-content "{\"Version\": \"2018-11-08\", \"DestinationReferences\": [\"refs/heads/main\"], \"Statements\": [{\"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers\": [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}"
```

Output:

```
{
  "approvalRuleTemplate": {
    "creationDate": 1571352720.773,
    "approvalRuleTemplateDescription": "Requires 1 approval for all pull requests from the CodeCommitReview pool",
    "lastModifiedDate": 1571358728.41,
    "approvalRuleTemplateId": "41de97b7-EXAMPLE",
    "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\", \"Statements\": [{\"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 1, \"ApprovalPoolMembers\": [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
    "approvalRuleTemplateName": "1-approver-rule-for-all-pull-requests",
    "ruleContentSha256": "2f6c21a5EXAMPLE",
    "lastModifiedUser": "arn:aws:iam::123456789012:user/Li_Juan"
  }
}
```

For more information, see [Manage Approval Rule Templates](#) in the *AWS CodeCommit User Guide*.

- For API details, see [UpdateApprovalRuleTemplateContent](#) in *AWS CLI Command Reference*.

update-approval-rule-template-description

The following code example shows how to use `update-approval-rule-template-description`.

AWS CLI

To update the description of an approval rule template

The following `update-approval-rule-template-description` example changes the description of the specified approval rule template to Requires 1 approval for all pull requests from the CodeCommitReview pool.:

```
aws codecommit update-approval-rule-template-description \  
  --approval-rule-template-name 1-approver-rule-for-all-pull-requests \  
  --approval-rule-template-description "Requires 1 approval for all pull requests  
from the CodeCommitReview pool"
```

Output:

```
{  
  "approvalRuleTemplate": {  
    "creationDate": 1571352720.773,  
    "approvalRuleTemplateDescription": "Requires 1 approval for all pull requests  
from the CodeCommitReview pool",  
    "lastModifiedDate": 1571358728.41,  
    "approvalRuleTemplateId": "41de97b7-EXAMPLE",  
    "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\", \"Statements\":  
[{\": \"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 1, \"ApprovalPoolMembers\":  
[\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",  
    "approvalRuleTemplateName": "1-approver-rule-for-all-pull-requests",  
    "ruleContentSha256": "2f6c21a5EXAMPLE",  
    "lastModifiedUser": "arn:aws:iam::123456789012:user/Li_Juan"  
  }  
}
```

For more information, see [Manage Approval Rule Templates](#) in the *AWS CodeCommit User Guide*.

- For API details, see [UpdateApprovalRuleTemplateDescription](#) in *AWS CLI Command Reference*.

`update-approval-rule-template-name`

The following code example shows how to use `update-approval-rule-template-name`.

AWS CLI

To update the name of an approval rule template

The following `update-approval-rule-template-name` example changes the name of an approval rule template from `1-approver-rule` to `1-approver-rule-for-all-pull-requests``.

```
aws codecommit update-approval-rule-template-name \  
  --old-approval-rule-template-name 1-approver-rule \  
  --new-approval-rule-template-name 1-approver-rule-for-all-pull-requests
```

Output:

```
{  
  "approvalRuleTemplate": {  
    "approvalRuleTemplateName": "1-approver-rule-for-all-pull-requests",  
    "lastModifiedDate": 1571358241.619,  
    "approvalRuleTemplateId": "41de97b7-EXAMPLE",  
    "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\", \"Statements\":  
  [{\"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 1, \"ApprovalPoolMembers\":  
  [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",  
    "creationDate": 1571352720.773,  
    "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",  
    "approvalRuleTemplateDescription": "All pull requests must be approved by one  
  developer on the team.",  
    "ruleContentSha256": "2f6c21a5cEXAMPLE"  
  }  
}
```

For more information, see [Manage Approval Rule Templates](#) in the *AWS CodeCommit User Guide*.

- For API details, see [UpdateApprovalRuleTemplateName](#) in *AWS CLI Command Reference*.

update-comment

The following code example shows how to use `update-comment`.

AWS CLI

To update a comment on a commit

This example demonstrates how to add the content "Fixed as requested. I'll update the pull request." to a comment with an ID of 442b498bEXAMPLE5756813.

```
aws codecommit update-comment \  
  --comment-id 442b498bEXAMPLE5756813 \  
  --content "Fixed as requested. I'll update the pull request."
```

Output:

```
{  
  "comment": {  
    "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",  
    "clientRequestToken": "",  
    "commentId": "442b498bEXAMPLE5756813",  
    "content": "Fixed as requested. I'll update the pull request.",  
    "creationDate": 1508369929.783,  
    "deleted": false,  
    "lastModifiedDate": 1508369929.287,  
    "callerReactions": [],  
    "reactionCounts":  
      {  
        "THUMBSUP" : 2  
      }  
  }  
}
```

- For API details, see [UpdateComment](#) in *AWS CLI Command Reference*.

update-default-branch

The following code example shows how to use `update-default-branch`.

AWS CLI

To change the default branch for a repository

This example changes the default branch for an AWS CodeCommit repository. This command produces output only if there are errors.

Command:

```
aws codecommit update-default-branch --repository-name MyDemoRepo --default-branch-name MyNewBranch
```

Output:

```
None.
```

- For API details, see [UpdateDefaultBranch](#) in *AWS CLI Command Reference*.

update-pull-request-approval-rule-content

The following code example shows how to use `update-pull-request-approval-rule-content`.

AWS CLI**To edit an approval rule for a pull request**

The following `update-pull-request-approval-rule-content` example updates the specified approval rule to require one user approval from an approval pool that includes any IAM user in the 123456789012 AWS account.

```
aws codecommit update-pull-request-approval-rule-content \
  --pull-request-id 27 \
  --approval-rule-name "Require two approved approvers" \
  --approval-rule-content "{Version: 2018-11-08, Statements: [{Type:
  \"Approvers\", NumberOfApprovalsNeeded: 1, ApprovalPoolMembers:
  [\"CodeCommitApprovers:123456789012:user/*\"]}]}"
```

Output:

```
{
  "approvalRule": {
    "approvalRuleContent": "{Version: 2018-11-08, Statements:
    [{Type: \"Approvers\", NumberOfApprovalsNeeded: 1, ApprovalPoolMembers:
    [\"CodeCommitApprovers:123456789012:user/*\"]}]}",
    "approvalRuleId": "aac33506-EXAMPLE",
    "originApprovalRuleTemplate": {},
    "creationDate": 1570752871.932,
```

```
"lastModifiedDate": 1570754058.333,  
"approvalRuleName": "Require two approved approvers",  
"lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",  
"ruleContentSha256": "cd93921cEXAMPLE",  
  }  
}
```

For more information, see [Edit or Delete an Approval Rule](#) in the *AWS CodeCommit User Guide*.

- For API details, see [UpdatePullRequestApprovalRuleContent](#) in *AWS CLI Command Reference*.

update-pull-request-approval-state

The following code example shows how to use `update-pull-request-approval-state`.

AWS CLI

To approve or revoke approval for a pull request

The following `update-pull-request-approval-state` example approves a pull request with the ID of 27 and a revision ID of 9f29d167EXAMPLE. If you wanted to revoke approval instead, then set the `--approval-state` parameter value to `REVOKE`.

```
aws codecommit update-pull-request-approval-state \  
  --pull-request-id 27 \  
  --revision-id 9f29d167EXAMPLE \  
  --approval-state "APPROVE"
```

This command produces no output.

For more information, see [Review a Pull Request](#) in the *AWS CodeCommit User Guide*.

- For API details, see [UpdatePullRequestApprovalState](#) in *AWS CLI Command Reference*.

update-pull-request-description

The following code example shows how to use `update-pull-request-description`.

AWS CLI

To change the description of a pull request

This example demonstrates how to change the description of a pull request with the ID of 47.

```
aws codecommit update-pull-request-description \  
  --pull-request-id 47 \  
  --description "Updated the pull request to remove unused global variable."
```

Output:

```
{  
  "pullRequest": {  
    "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",  
    "clientRequestToken": "",  
    "creationDate": 1508530823.155,  
    "description": "Updated the pull request to remove unused global variable.",  
    "lastActivityDate": 1508372423.204,  
    "pullRequestId": "47",  
    "pullRequestStatus": "OPEN",  
    "pullRequestTargets": [  
      {  
        "destinationCommit": "9f31c968EXAMPLE",  
        "destinationReference": "refs/heads/main",  
        "mergeMetadata": {  
          "isMerged": false,  
        },  
        "repositoryName": "MyDemoRepo",  
        "sourceCommit": "99132ab0EXAMPLE",  
        "sourceReference": "refs/heads/variables-branch"  
      }  
    ],  
    "title": "Consolidation of global variables"  
  }  
}
```

- For API details, see [UpdatePullRequestDescription](#) in *AWS CLI Command Reference*.

update-pull-request-status

The following code example shows how to use `update-pull-request-status`.

AWS CLI

To change the status of a pull request

This example demonstrates how to change the status of a pull request with the ID of 42 to a status of CLOSED in an AWS CodeCommit repository named MyDemoRepo.

```
aws codecommit update-pull-request-status \  
  --pull-request-id 42 \  
  --pull-request-status CLOSED
```

Output:

```
{  
  "pullRequest": {  
    "approvalRules": [  
      {  
        "approvalRuleContent": "{\"Version\": \"2018-11-08\", \"Statements\":  
[{\n\"Type\": \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers\":  
[\n\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}}]\"",  
        "approvalRuleId": "dd8b17fe-EXAMPLE",  
        "approvalRuleName": "2-approvers-needed-for-this-change",  
        "creationDate": 1571356106.936,  
        "lastModifiedDate": 571356106.936,  
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",  
        "ruleContentSha256": "4711b576EXAMPLE"  
      }  
    ],  
    "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",  
    "clientRequestToken": "",  
    "creationDate": 1508530823.165,  
    "description": "Updated the pull request to remove unused global variable.",  
    "lastActivityDate": 1508372423.12,  
    "pullRequestId": "47",  
    "pullRequestStatus": "CLOSED",  
    "pullRequestTargets": [  
      {  
        "destinationCommit": "9f31c968EXAMPLE",  
        "destinationReference": "refs/heads/main",  
        "mergeMetadata": {  
          "isMerged": false,  
        },  
        "repositoryName": "MyDemoRepo",  
        "sourceCommit": "99132ab0EXAMPLE",  
        "sourceReference": "refs/heads/variables-branch"  
      }  
    ],  
  },  
}
```

```

    "title": "Consolidation of global variables"
  }
}

```

- For API details, see [UpdatePullRequestStatus](#) in *AWS CLI Command Reference*.

update-pull-request-title

The following code example shows how to use `update-pull-request-title`.

AWS CLI

To change the title of a pull request

This example demonstrates how to change the title of a pull request with the ID of 47.

```

aws codecommit update-pull-request-title \
  --pull-request-id 47 \
  --title "Consolidation of global variables - updated review"

```

Output:

```

{
  "pullRequest": {
    "approvalRules": [
      {
        "approvalRuleContent": "{\"Version\": \"2018-11-08\",
        \"DestinationReferences\": [\"refs/heads/main\"], \"Statements\": [{\"Type
        \": \"Approvers\", \"NumberOfApprovalsNeeded\": 2, \"ApprovalPoolMembers\":
        [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
        "approvalRuleId": "dd8b17fe-EXAMPLE",
        "approvalRuleName": "2-approver-rule-for-main",
        "creationDate": 1571356106.936,
        "lastModifiedDate": 571356106.936,
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
        "originApprovalRuleTemplate": {
          "approvalRuleTemplateId": "dd8b26gr-EXAMPLE",
          "approvalRuleTemplateName": "2-approver-rule-for-main"
        },
        "ruleContentSha256": "4711b576EXAMPLE"
      }
    ],
  },
}

```

```
    "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",
    "clientRequestToken": "",
    "creationDate": 1508530823.12,
    "description": "Review the latest changes and updates to the global
variables. I have updated this request with some changes, including removing some
unused variables.",
    "lastActivityDate": 1508372657.188,
    "pullRequestId": "47",
    "pullRequestStatus": "OPEN",
    "pullRequestTargets": [
      {
        "destinationCommit": "9f31c968EXAMPLE",
        "destinationReference": "refs/heads/main",
        "mergeMetadata": {
          "isMerged": false,
        },
        "repositoryName": "MyDemoRepo",
        "sourceCommit": "99132ab0EXAMPLE",
        "sourceReference": "refs/heads/variables-branch"
      }
    ],
    "title": "Consolidation of global variables - updated review"
  }
}
```

- For API details, see [UpdatePullRequestTitle](#) in *AWS CLI Command Reference*.

update-repository-description

The following code example shows how to use `update-repository-description`.

AWS CLI

To change the description for a repository

This example changes the description for an AWS CodeCommit repository. This command produces output only if there are errors.

Command:

```
aws codecommit update-repository-description --repository-name MyDemoRepo --
repository-description "This description was changed"
```


Output:

```
None .
```

- For API details, see [UpdateRepositoryDescription](#) in *AWS CLI Command Reference*.

update-repository-name

The following code example shows how to use `update-repository-name`.

AWS CLI**To change the name of a repository**

This example changes the name of an AWS CodeCommit repository. This command produces output only if there are errors. Changing the name of the AWS CodeCommit repository will change the SSH and HTTPS URLs that users need to connect to the repository. Users will not be able to connect to this repository until they update their connection settings. Also, because the repository's ARN will change, changing the repository name will invalidate any IAM user policies that rely on this repository's ARN.

Command:

```
aws codecommit update-repository-name --old-name MyDemoRepo --new-name  
MyRenamedDemoRepo
```

Output:

```
None .
```

- For API details, see [UpdateRepositoryName](#) in *AWS CLI Command Reference*.

CodeDeploy examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with CodeDeploy.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-tags-to-on-premises-instances

The following code example shows how to use `add-tags-to-on-premises-instances`.

AWS CLI

To add tags to on-premises instances

The following `add-tags-to-on-premises-instances` example associates in AWS CodeDeploy the same on-premises instance tag to two on-premises instances. It does not register the on-premises instances with AWS CodeDeploy.

```
aws deploy add-tags-to-on-premises-instances \  
  --instance-names AssetTag12010298EX AssetTag23121309EX \  
  --tags Key=Name,Value=CodeDeployDemo-OnPrem
```

This command produces no output.

- For API details, see [AddTagsToOnPremisesInstances](#) in *AWS CLI Command Reference*.

batch-get-application-revisions

The following code example shows how to use `batch-get-application-revisions`.

AWS CLI

To retrieve information about application revisions

The following `batch-get-application-revisions` example retrieves information about the specified revision stored in a GitHub repository.

```
aws deploy batch-get-application-revisions \  
  --application-name my-codedeploy-application \  
  --revisions "[{\\"gitHubLocation\\": {\\"commitId\\":  
  \\"fa85936EXAMPLEa31736c051f10d77297EXAMPLE\\", \\"repository\\": \\"my-github-token/my-  
  repository\\"}, \\"revisionType\\": \\"GitHub\\"}]"
```

Output:

```
{  
  "revisions": [  
    {  
      "genericRevisionInfo": {  
        "description": "Application revision registered by Deployment ID: d-  
A1B2C3111",  
        "lastUsedTime": 1556912355.884,  
        "registerTime": 1556912355.884,  
        "firstUsedTime": 1556912355.884,  
        "deploymentGroups": []  
      },  
      "revisionLocation": {  
        "revisionType": "GitHub",  
        "gitHubLocation": {  
          "commitId": "fa85936EXAMPLEa31736c051f10d77297EXAMPLE",  
          "repository": "my-github-token/my-repository"  
        }  
      }  
    }  
  ],  
  "applicationName": "my-codedeploy-application",  
  "errorMessage": ""  
}
```

For more information, see [BatchGetApplicationRevisions](#) in the *AWS CodeDeploy API Reference*.

- For API details, see [BatchGetApplicationRevisions](#) in *AWS CLI Command Reference*.

batch-get-applications

The following code example shows how to use batch-get-applications.

AWS CLI

To get information about multiple applications

The following `batch-get-applications` example displays information about multiple applications that are associated with the user's AWS account.

```
aws deploy batch-get-applications --application-names WordPress_App MyOther_App
```

Output:

```
{
  "applicationsInfo": [
    {
      "applicationName": "WordPress_App",
      "applicationId": "d9dd6993-f171-44fa-a811-211e4EXAMPLE",
      "createTime": 1407878168.078,
      "linkedToGitHub": false
    },
    {
      "applicationName": "MyOther_App",
      "applicationId": "8ca57519-31da-42b2-9194-8bb16EXAMPLE",
      "createTime": 1407453571.63,
      "linkedToGitHub": false
    }
  ]
}
```

- For API details, see [BatchGetApplications](#) in *AWS CLI Command Reference*.

batch-get-deployment-groups

The following code example shows how to use `batch-get-deployment-groups`.

AWS CLI

To retrieve information about one or more deployment groups

The following `batch-get-deployment-groups` example retrieves information about two of the deployment groups that are associated with the specified CodeDeploy application.

```
aws deploy batch-get-deployment-groups \
```

```
--application-name my-codedeploy-application \  
--deployment-group-names ["my-deployment-group-1","my-deployment-group-2"]
```

Output:

```
{  
  "deploymentGroupsInfo": [  
    {  
      "deploymentStyle": {  
        "deploymentOption": "WITHOUT_TRAFFIC_CONTROL",  
        "deploymentType": "IN_PLACE"  
      },  
      "autoRollbackConfiguration": {  
        "enabled": false  
      },  
      "onPremisesTagSet": {  
        "onPremisesTagSetList": []  
      },  
      "serviceRoleArn": "arn:aws:iam::123456789012:role/  
CodeDeployServiceRole",  
      "lastAttemptedDeployment": {  
        "endTime": 1556912366.415,  
        "status": "Failed",  
        "createTime": 1556912355.884,  
        "deploymentId": "d-A1B2C3111"  
      },  
      "autoScalingGroups": [],  
      "deploymentGroupName": "my-deployment-group-1",  
      "ec2TagSet": {  
        "ec2TagSetList": [  
          [  
            {  
              "Type": "KEY_AND_VALUE",  
              "Value": "my-EC2-instance",  
              "Key": "Name"  
            }  
          ]  
        ]  
      },  
      "deploymentGroupId": "a1b2c3d4-5678-90ab-cdef-11111example",  
      "triggerConfigurations": [],  
      "applicationName": "my-codedeploy-application",  
      "computePlatform": "Server",  
    }  
  ]  
}
```

```

    "deploymentConfigName": "CodeDeployDefault.AllAtOnce"
  },
  {
    "deploymentStyle": {
      "deploymentOption": "WITHOUT_TRAFFIC_CONTROL",
      "deploymentType": "IN_PLACE"
    },
    "autoRollbackConfiguration": {
      "enabled": false
    },
    "onPremisesTagSet": {
      "onPremisesTagSetList": []
    },
    "serviceRoleArn": "arn:aws:iam::123456789012:role/
CodeDeployServiceRole",
    "autoScalingGroups": [],
    "deploymentGroupName": "my-deployment-group-2",
    "ec2TagSet": {
      "ec2TagSetList": [
        [
          {
            "Type": "KEY_AND_VALUE",
            "Value": "my-EC2-instance",
            "Key": "Name"
          }
        ]
      ]
    },
    "deploymentGroupId": "a1b2c3d4-5678-90ab-cdef-22222example",
    "triggerConfigurations": [],
    "applicationName": "my-codedeploy-application",
    "computePlatform": "Server",
    "deploymentConfigName": "CodeDeployDefault.AllAtOnce"
  }
],
"errorMessage": ""
}

```

For more information, see [BatchGetDeploymentGroups](#) in the *AWS CodeDeploy API Reference*.

- For API details, see [BatchGetDeploymentGroups](#) in *AWS CLI Command Reference*.

batch-get-deployment-targets

The following code example shows how to use `batch-get-deployment-targets`.

AWS CLI

To retrieve the targets associated with a deployment

The following `batch-get-deployment-targets` example returns information about one of the targets associated with the specified deployment.

```
aws deploy batch-get-deployment-targets \  
  --deployment-id "d-1A2B3C4D5" \  
  --target-ids "i-01a2b3c4d5e6f1111"
```

Output:

```
{  
  "deploymentTargets": [  
    {  
      "deploymentTargetType": "InstanceTarget",  
      "instanceTarget": {  
        "lifecycleEvents": [  
          {  
            "startTime": 1556918592.162,  
            "lifecycleEventName": "ApplicationStop",  
            "status": "Succeeded",  
            "endTime": 1556918592.247,  
            "diagnostics": {  
              "scriptName": "",  
              "errorCode": "Success",  
              "logTail": "",  
              "message": "Succeeded"  
            }  
          }  
        ],  
      },  
      {  
        "startTime": 1556918593.193,  
        "lifecycleEventName": "DownloadBundle",  
        "status": "Succeeded",  
        "endTime": 1556918593.981,  
        "diagnostics": {  
          "scriptName": "",  
          "errorCode": "Success",
```

```

        "logTail": "",
        "message": "Succeeded"
      }
    },
    {
      "startTime": 1556918594.805,
      "lifecycleEventName": "BeforeInstall",
      "status": "Succeeded",
      "endTime": 1556918681.807,
      "diagnostics": {
        "scriptName": "",
        "errorCode": "Success",
        "logTail": "",
        "message": "Succeeded"
      }
    }
  ],
  "targetArn": "arn:aws:ec2:us-west-2:123456789012:instance/i-01a2b3c4d5e6f1111",
  "deploymentId": "d-1A2B3C4D5",
  "lastUpdatedAt": 1556918687.504,
  "targetId": "i-01a2b3c4d5e6f1111",
  "status": "Succeeded"
}
]
}

```

For more information, see [BatchGetDeploymentTargets](#) in the *AWS CodeDeploy API Reference*.

- For API details, see [BatchGetDeploymentTargets](#) in *AWS CLI Command Reference*.

batch-get-deployments

The following code example shows how to use batch-get-deployments.

AWS CLI

To get information about multiple deployments

The following batch-get-deployments example displays information about multiple deployments that are associated with the user's AWS account.


```
aws deploy batch-get-deployments --deployment-ids d-A1B2C3111 d-A1B2C3222
```

Output:

```
{
  "deploymentsInfo": [
    {
      "applicationName": "WordPress_App",
      "status": "Failed",
      "deploymentOverview": {
        "Failed": 0,
        "InProgress": 0,
        "Skipped": 0,
        "Succeeded": 1,
        "Pending": 0
      },
      "deploymentConfigName": "CodeDeployDefault.OneAtATime",
      "creator": "user",
      "deploymentGroupName": "WordPress_DG",
      "revision": {
        "revisionType": "S3",
        "s3Location": {
          "bundleType": "zip",
          "version": "uTecLusEXAMPLEFXtfUcyfV8bEXAMPLE",
          "bucket": "CodeDeployDemoBucket",
          "key": "WordPressApp.zip"
        }
      },
      "deploymentId": "d-A1B2C3111",
      "createTime": 1408480721.9,
      "completeTime": 1408480741.822
    },
    {
      "applicationName": "MyOther_App",
      "status": "Failed",
      "deploymentOverview": {
        "Failed": 1,
        "InProgress": 0,
        "Skipped": 0,
        "Succeeded": 0,
        "Pending": 0
      },
      "deploymentConfigName": "CodeDeployDefault.OneAtATime",
```

```

    "creator": "user",
    "errorInformation": {
      "message": "Deployment failed: Constraint default violated: No hosts
succeeded.",
      "code": "HEALTH_CONSTRAINTS"
    },
    "deploymentGroupName": "MyOther_DG",
    "revision": {
      "revisionType": "S3",
      "s3Location": {
        "bundleType": "zip",
        "eTag": "\"dd56cfdEXAMPLE8e768f9d77fEXAMPLE\"",
        "bucket": "CodeDeployDemoBucket",
        "key": "MyOtherApp.zip"
      }
    },
    "deploymentId": "d-A1B2C3222",
    "createTime": 1409764576.589,
    "completeTime": 1409764596.101
  }
]
}

```

- For API details, see [BatchGetDeployments](#) in *AWS CLI Command Reference*.

batch-get-on-premises-instances

The following code example shows how to use `batch-get-on-premises-instances`.

AWS CLI

To get information about one or more on-premises instances

The following `batch-get-on-premises-instances` example gets information about two on-premises instances.

```
aws deploy batch-get-on-premises-instances --instance-names AssetTag12010298EX
AssetTag23121309EX
```

Output:

```
{
```

```

    "instanceInfos": [
      {
        "iamUserArn": "arn:aws:iam::123456789012:user/AWS/CodeDeploy/
AssetTag12010298EX",
        "tags": [
          {
            "Value": "CodeDeployDemo-OnPrem",
            "Key": "Name"
          }
        ],
        "instanceName": "AssetTag12010298EX",
        "registerTime": 1425579465.228,
        "instanceArn": "arn:aws:codedeploy:us-west-2:123456789012:instance/
AssetTag12010298EX_4IwLNI2Alh"
      },
      {
        "iamUserArn": "arn:aws:iam::123456789012:user/AWS/CodeDeploy/
AssetTag23121309EX",
        "tags": [
          {
            "Value": "CodeDeployDemo-OnPrem",
            "Key": "Name"
          }
        ],
        "instanceName": "AssetTag23121309EX",
        "registerTime": 1425595585.988,
        "instanceArn": "arn:aws:codedeploy:us-west-2:80398EXAMPLE:instance/
AssetTag23121309EX_PomUy64Was"
      }
    ]
  }

```

- For API details, see [BatchGetOnPremisesInstances](#) in *AWS CLI Command Reference*.

continue-deployment

The following code example shows how to use continue-deployment.

AWS CLI

To start rerouting traffic without waiting for a specified wait time to elapse.

The following `continue-deployment` example starts rerouting traffic from instances in the original environment that are ready to start shifting traffic to instances in the replacement environment.

```
aws deploy continue-deployment \  
  --deployment-id "d-A1B2C3111" \  
  --deployment-wait-type "READY_WAIT"
```

This command produces no output.

For more information, see [ContinueDeployment](#) in the *AWS CodeDeploy API Reference*.

- For API details, see [ContinueDeployment](#) in *AWS CLI Command Reference*.

create-application

The following code example shows how to use `create-application`.

AWS CLI

To create an application

The following `create-application` example creates an application and associates it with the user's AWS account.

```
aws deploy create-application --application-name MyOther_App
```

Output:

```
{  
  "applicationId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"  
}
```

- For API details, see [CreateApplication](#) in *AWS CLI Command Reference*.

create-deployment-config

The following code example shows how to use `create-deployment-config`.

AWS CLI

To create a custom deployment configuration

The following `create-deployment-config` example creates a custom deployment configuration and associates it with the user's AWS account.

```
aws deploy create-deployment-config \  
  --deployment-config-name ThreeQuartersHealthy \  
  --minimum-healthy-hosts type=FLEET_PERCENT,value=75
```

Output:

```
{  
  "deploymentConfigId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"  
}
```

- For API details, see [CreateDeploymentConfig](#) in *AWS CLI Command Reference*.

create-deployment-group

The following code example shows how to use `create-deployment-group`.

AWS CLI

To create a deployment group

The following `create-deployment-group` example creates a deployment group and associates it with the specified application and the user's AWS account.

```
aws deploy create-deployment-group \  
  --application-name WordPress_App \  
  --auto-scaling-groups CodeDeployDemo-ASG \  
  --deployment-config-name CodeDeployDefault.OneAtATime \  
  --deployment-group-name WordPress_DG \  
  --ec2-tag-filters Key=Name,Value=CodeDeployDemo,Type=KEY_AND_VALUE \  
  --service-role-arn arn:aws:iam::123456789012:role/CodeDeployDemoRole
```

Output:

```
{  
  "deploymentGroupId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"  
}
```

- For API details, see [CreateDeploymentGroup](#) in *AWS CLI Command Reference*.

create-deployment

The following code example shows how to use create-deployment.

AWS CLI

Example 1: To create a CodeDeploy deployment using the EC2/On-premises compute platform

The following create-deployment example creates a deployment and associates it with the user's AWS account.

```
aws deploy create-deployment \  
  --application-name WordPress_App \  
  --deployment-config-name CodeDeployDefault.OneAtATime \  
  --deployment-group-name WordPress_DG \  
  --description "My demo deployment" \  
  --s3-location  
bucket=CodeDeployDemoBucket,bundleType=zip,eTag=dd56cfdEXAMPLE8e768f9d77fEXAMPLE,key=WordPress
```

Output:

```
{  
  "deploymentId": "d-A1B2C3111"  
}
```

Example 2: To create a CodeDeploy deployment using the Amazon ECS compute platform

The following create-deployment example uses the following two files to deploy an Amazon ECS service.

Contents of create-deployment.json file:

```
{  
  "applicationName": "ecs-deployment",  
  "deploymentGroupName": "ecs-deployment-dg",  
  "revision": {  
    "revisionType": "S3",  
    "s3Location": {
```

```
        "bucket": "ecs-deployment-bucket",
        "key": "appspec.yaml",
        "bundleType": "YAML"
    }
}
```

That file, in turn, retrieves the following file `appspec.yaml` from an S3 bucket called `ecs-deployment-bucket`.

```
version: 0.0
Resources:
  - TargetService:
      Type: AWS::ECS::Service
      Properties:
        TaskDefinition: "arn:aws:ecs:region:123456789012:task-definition/ecs-task-def:2"
        LoadBalancerInfo:
          ContainerName: "sample-app"
          ContainerPort: 80
          PlatformVersion: "LATEST"
```

Command:

```
aws deploy create-deployment \
  --cli-input-json file://create-deployment.json \
  --region us-east-1
```

Output:

```
{
  "deploymentId": "d-1234ABCDE"
}
```

For more information, see [CreateDeployment](#) in the *AWS CodeDeploy API Reference*.

- For API details, see [CreateDeployment](#) in *AWS CLI Command Reference*.

delete-application

The following code example shows how to use `delete-application`.

AWS CLI

To delete an application

The following `delete-application` example deletes the specified application that is associated with the user's AWS account.

```
aws deploy delete-application --application-name WordPress_App
```

This command produces no output.

- For API details, see [DeleteApplication](#) in *AWS CLI Command Reference*.

`delete-deployment-config`

The following code example shows how to use `delete-deployment-config`.

AWS CLI

To delete a deployment configuration

The following `delete-deployment-config` example deletes a custom deployment configuration that is associated with the user's AWS account.

```
aws deploy delete-deployment-config --deployment-config-name ThreeQuartersHealthy
```

This command produces no output.

- For API details, see [DeleteDeploymentConfig](#) in *AWS CLI Command Reference*.

`delete-deployment-group`

The following code example shows how to use `delete-deployment-group`.

AWS CLI

To delete a deployment group

The following `delete-deployment-group` example deletes a deployment group that is associated with the specified application.

```
aws deploy delete-deployment-group \
```



```
--application-name WordPress_App \  
--deployment-group-name WordPress_DG
```

Output:

```
{  
  "hooksNotCleanedUp": []  
}
```

- For API details, see [DeleteDeploymentGroup](#) in *AWS CLI Command Reference*.

delete-git-hub-account-token

The following code example shows how to use `delete-git-hub-account-token`.

AWS CLI**To deletes a GitHub account connection**

The following `delete-git-hub-account-token` example deletes the connection of the specified GitHub account.

```
aws deploy delete-git-hub-account-token --token-name my-github-account
```

Output:

```
{  
  "tokenName": "my-github-account"  
}
```

For more information, see [DeleteGitHubAccountToken](#) in the *AWS CodeDeploy API Reference*.

- For API details, see [DeleteGitHubAccountToken](#) in *AWS CLI Command Reference*.

deregister-on-premises-instance

The following code example shows how to use `deregister-on-premises-instance`.

AWS CLI**To deregister an on-premises instance**

The following `deregister-on-premises-instance` example deregisters an on-premises instance with AWS CodeDeploy, but it does not delete the IAM user associated with the instance, nor does it disassociate in AWS CodeDeploy the on-premises instance tags from the instance. It also does not uninstall the AWS CodeDeploy Agent from the instance nor remove the on-premises configuration file from the instance.

```
aws deploy deregister-on-premises-instance --instance-name AssetTag12010298EX
```

This command produces no output.

- For API details, see [DeregisterOnPremisesInstance](#) in *AWS CLI Command Reference*.

deregister

The following code example shows how to use `deregister`.

AWS CLI

To deregister an on-premises instance

The following `deregister` example deregisters an on-premises instance with AWS CodeDeploy. It does not delete the IAM user that is associated with the instance. It disassociates in AWS CodeDeploy the on-premises tags from the instance. It does not uninstall the AWS CodeDeploy Agent from the instance nor remove the on-premises configuration file from the instance.

```
aws deploy deregister \  
  --instance-name AssetTag12010298EX \  
  --no-delete-iam-user \  
  --region us-west-2
```

Output:

```
Retrieving on-premises instance information... DONE  
IamUserArn: arn:aws:iam::80398EXAMPLE:user/AWS/CodeDeploy/AssetTag12010298EX  
Tags: Key=Name,Value=CodeDeployDemo-OnPrem  
Removing tags from the on-premises instance... DONE  
Deregistering the on-premises instance... DONE  
Run the following command on the on-premises instance to uninstall the codedeploy-agent:
```

```
aws deploy uninstall
```

- For API details, see [Deregister](#) in *AWS CLI Command Reference*.

get-application-revision

The following code example shows how to use `get-application-revision`.

AWS CLI

To get information about an application revision

The following `get-application-revision` example displays information about an application revision that is associated with the specified application.

```
aws deploy get-application-revision \  
  --application-name WordPress_App \  
  --s3-location  
bucket=CodeDeployDemoBucket,bundleType=zip,eTag=dd56cfdEXAMPLE8e768f9d77fEXAMPLE,key=WordPressApp.zip
```

Output:

```
{  
  "applicationName": "WordPress_App",  
  "revisionInfo": {  
    "description": "Application revision registered by Deployment ID: d-A1B2C3111",  
    "registerTime": 1411076520.009,  
    "deploymentGroups": "WordPress_DG",  
    "lastUsedTime": 1411076520.009,  
    "firstUsedTime": 1411076520.009  
  },  
  "revision": {  
    "revisionType": "S3",  
    "s3Location": {  
      "bundleType": "zip",  
      "eTag": "dd56cfdEXAMPLE8e768f9d77fEXAMPLE",  
      "bucket": "CodeDeployDemoBucket",  
      "key": "WordPressApp.zip"  
    }  
  }  
}
```

- For API details, see [GetApplicationRevision](#) in *AWS CLI Command Reference*.

get-application

The following code example shows how to use `get-application`.

AWS CLI

To get information about an application

The following `get-application` example displays information about an application that is associated with the user's AWS account.

```
aws deploy get-application --application-name WordPress_App
```

Output:

```
{
  "application": {
    "applicationName": "WordPress_App",
    "applicationId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "createTime": 1407878168.078,
    "linkedToGitHub": false
  }
}
```

- For API details, see [GetApplication](#) in *AWS CLI Command Reference*.

get-deployment-config

The following code example shows how to use `get-deployment-config`.

AWS CLI

To get information about a deployment configuration

The following `get-deployment-config` example displays information about a deployment configuration that is associated with the user's AWS account.

```
aws deploy get-deployment-config --deployment-config-name ThreeQuartersHealthy
```

Output:

```
{
  "deploymentConfigInfo": {
    "deploymentConfigId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "minimumHealthyHosts": {
      "type": "FLEET_PERCENT",
      "value": 75
    },
    "createTime": 1411081164.379,
    "deploymentConfigName": "ThreeQuartersHealthy"
  }
}
```

- For API details, see [GetDeploymentConfig](#) in *AWS CLI Command Reference*.

get-deployment-group

The following code example shows how to use `get-deployment-group`.

AWS CLI**To view information about a deployment group**

The following `get-deployment-group` example displays information about a deployment group that is associated with the specified application.

```
aws deploy get-deployment-group \
  --application-name WordPress_App \
  --deployment-group-name WordPress_DG
```

Output:

```
{
  "deploymentGroupInfo": {
    "applicationName": "WordPress_App",
    "autoScalingGroups": [
      "CodeDeployDemo-ASG"
    ],
    "deploymentConfigName": "CodeDeployDefault.OneAtATime",
    "ec2TagFilters": [
      {
```

```

        "Type": "KEY_AND_VALUE",
        "Value": "CodeDeployDemo",
        "Key": "Name"
    }
],
"deploymentGroupId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
"serviceRoleArn": "arn:aws:iam::123456789012:role/CodeDeployDemoRole",
"deploymentGroupName": "WordPress_DG"
}
}

```

- For API details, see [GetDeploymentGroup](#) in *AWS CLI Command Reference*.

get-deployment-instance

The following code example shows how to use `get-deployment-instance`.

AWS CLI

To get information about a deployment instance

The following `get-deployment-instance` example displays information about a deployment instance that is associated with the specified deployment.

```
aws deploy get-deployment-instance --deployment-id d-QA4G4F9EX --instance-id
i-902e9fEX
```

Output:

```

{
  "instanceSummary": {
    "instanceId": "arn:aws:ec2:us-east-1:80398EXAMPLE:instance/i-902e9fEX",
    "lifecycleEvents": [
      {
        "status": "Succeeded",
        "endTime": 1408480726.569,
        "startTime": 1408480726.437,
        "lifecycleEventName": "ApplicationStop"
      },
      {
        "status": "Succeeded",
        "endTime": 1408480728.016,

```

```
        "startTime": 1408480727.665,  
        "lifecycleEventName": "DownloadBundle"  
    },  
    {  
        "status": "Succeeded",  
        "endTime": 1408480729.744,  
        "startTime": 1408480729.125,  
        "lifecycleEventName": "BeforeInstall"  
    },  
    {  
        "status": "Succeeded",  
        "endTime": 1408480730.979,  
        "startTime": 1408480730.844,  
        "lifecycleEventName": "Install"  
    },  
    {  
        "status": "Failed",  
        "endTime": 1408480732.603,  
        "startTime": 1408480732.1,  
        "lifecycleEventName": "AfterInstall"  
    },  
    {  
        "status": "Skipped",  
        "endTime": 1408480732.606,  
        "lifecycleEventName": "ApplicationStart"  
    },  
    {  
        "status": "Skipped",  
        "endTime": 1408480732.606,  
        "lifecycleEventName": "ValidateService"  
    }  
],  
    "deploymentId": "d-QA4G4F9EX",  
    "lastUpdatedAt": 1408480733.152,  
    "status": "Failed"  
}  
}
```

- For API details, see [GetDeploymentInstance](#) in *AWS CLI Command Reference*.

get-deployment-target

The following code example shows how to use `get-deployment-target`.

AWS CLI

To return information about a deployment target

The following `get-deployment-target` example returns information about a deployment target that is associated with the specified deployment.

```
aws deploy get-deployment-target \  
  --deployment-id "d-A1B2C3111" \  
  --target-id "i-a1b2c3d4e5f611111"
```

Output:

```
{  
  "deploymentTarget": {  
    "deploymentTargetType": "InstanceTarget",  
    "instanceTarget": {  
      "lastUpdatedAt": 1556918687.504,  
      "targetId": "i-a1b2c3d4e5f611111",  
      "targetArn": "arn:aws:ec2:us-west-2:123456789012:instance/i-a1b2c3d4e5f611111",  
      "status": "Succeeded",  
      "lifecycleEvents": [  
        {  
          "status": "Succeeded",  
          "diagnostics": {  
            "errorCode": "Success",  
            "message": "Succeeded",  
            "logTail": "",  
            "scriptName": ""  
          },  
          "lifecycleEventName": "ApplicationStop",  
          "startTime": 1556918592.162,  
          "endTime": 1556918592.247  
        },  
        {  
          "status": "Succeeded",  
          "diagnostics": {  
            "errorCode": "Success",  
            "message": "Succeeded",  
            "logTail": "",  
            "scriptName": ""  
          }  
        }  
      ]  
    }  
  }  
}
```



```
    "lifecycleEventName": "DownloadBundle",
    "startTime": 1556918593.193,
    "endTime": 1556918593.981
  },
  {
    "status": "Succeeded",
    "diagnostics": {
      "errorCode": "Success",
      "message": "Succeeded",
      "logTail": "",
      "scriptName": ""
    }
  },
  {
    "lifecycleEventName": "BeforeInstall",
    "startTime": 1556918594.805,
    "endTime": 1556918681.807
  },
  {
    "status": "Succeeded",
    "diagnostics": {
      "errorCode": "Success",
      "message": "Succeeded",
      "logTail": "",
      "scriptName": ""
    }
  },
  {
    "lifecycleEventName": "Install",
    "startTime": 1556918682.696,
    "endTime": 1556918683.005
  },
  {
    "status": "Succeeded",
    "diagnostics": {
      "errorCode": "Success",
      "message": "Succeeded",
      "logTail": "",
      "scriptName": ""
    }
  },
  {
    "lifecycleEventName": "AfterInstall",
    "startTime": 1556918684.135,
    "endTime": 1556918684.216
  },
  {
    "status": "Succeeded",
    "diagnostics": {
      "errorCode": "Success",
```

```

        "message": "Succeeded",
        "logTail": "",
        "scriptName": ""
    },
    "lifecycleEventName": "ApplicationStart",
    "startTime": 1556918685.211,
    "endTime": 1556918685.295
},
{
    "status": "Succeeded",
    "diagnostics": {
        "errorCode": "Success",
        "message": "Succeeded",
        "logTail": "",
        "scriptName": ""
    },
    "lifecycleEventName": "ValidateService",
    "startTime": 1556918686.65,
    "endTime": 1556918686.747
}
],
"deploymentId": "d-A1B2C3111"
}
}
}

```

For more information, see [GetDeploymentTarget](#) in the *AWS CodeDeploy API Reference*.

- For API details, see [GetDeploymentTarget](#) in *AWS CLI Command Reference*.

get-deployment

The following code example shows how to use `get-deployment`.

AWS CLI

To get information about a deployment

The following `get-deployment` example displays information about a deployment that is associated with the user's AWS account.

```
aws deploy get-deployment --deployment-id d-A1B2C3123
```

Output:

```
{
  "deploymentInfo": {
    "applicationName": "WordPress_App",
    "status": "Succeeded",
    "deploymentOverview": {
      "Failed": 0,
      "InProgress": 0,
      "Skipped": 0,
      "Succeeded": 1,
      "Pending": 0
    },
    "deploymentConfigName": "CodeDeployDefault.OneAtATime",
    "creator": "user",
    "description": "My WordPress app deployment",
    "revision": {
      "revisionType": "S3",
      "s3Location": {
        "bundleType": "zip",
        "eTag": "\"dd56cfdEXAMPLE8e768f9d77fEXAMPLE\"",
        "bucket": "CodeDeployDemoBucket",
        "key": "WordPressApp.zip"
      }
    },
    "deploymentId": "d-A1B2C3123",
    "deploymentGroupName": "WordPress_DG",
    "createTime": 1409764576.589,
    "completeTime": 1409764596.101,
    "ignoreApplicationStopFailures": false
  }
}
```

- For API details, see [GetDeployment](#) in *AWS CLI Command Reference*.

get-on-premises-instance

The following code example shows how to use `get-on-premises-instance`.

AWS CLI**To get information about an on-premises instance**

The following `get-on-premises-instance` example retrieves information about the specified on-premises instance.

```
aws deploy get-on-premises-instance --instance-name AssetTag12010298EX
```

Output:

```
{
  "instanceInfo": {
    "iamUserArn": "arn:aws:iam::123456789012:user/AWS/CodeDeploy/
AssetTag12010298EX",
    "tags": [
      {
        "Value": "CodeDeployDemo-OnPrem",
        "Key": "Name"
      }
    ],
    "instanceName": "AssetTag12010298EX",
    "registerTime": 1425579465.228,
    "instanceArn": "arn:aws:codedeploy:us-east-1:123456789012:instance/
AssetTag12010298EX_4IwLNI2Alh"
  }
}
```

- For API details, see [GetOnPremisesInstance](#) in *AWS CLI Command Reference*.

install

The following code example shows how to use `install`.

AWS CLI

To install an on-premises instance

The following `install` example copies the on-premises configuration file from the specified location on the instance to the location on the instance that the AWS CodeDeploy Agent expects to find it. It also installs the AWS CodeDeploy Agent on the instance. It does not create any IAM user, nor register the on-premises instance with AWS CodeDeploy, nor associate any on-premises instance tags in AWS CodeDeploy for the instance.

```
aws deploy install \
```

```
--override-config \  
--config-file C:\temp\codedeploy.onpremises.yml \  
--region us-west-2 \  
--agent-installer s3://aws-codedeploy-us-west-2/latest/codedeploy-agent.msi
```

Output:

```
Creating the on-premises instance configuration file... DONE  
Installing the AWS CodeDeploy Agent... DONE
```

- For API details, see [Install](#) in *AWS CLI Command Reference*.

list-application-revisions

The following code example shows how to use `list-application-revisions`.

AWS CLI

To get information about application revisions

The following `list-application-revisions` example displays information about all application revisions that are associated with the specified application.

```
aws deploy list-application-revisions \  
  --application-name WordPress_App \  
  --s3-bucket CodeDeployDemoBucket \  
  --deployed exclude \  
  --s3-key-prefix WordPress_ \  
  --sort-by lastUsedTime \  
  --sort-order descending
```

Output:

```
{  
  "revisions": [  
    {  
      "revisionType": "S3",  
      "s3Location": {  
        "version": "uTecLusvCB_JqHFXtfUcyfV8bEXAMPLE",  
        "bucket": "CodeDeployDemoBucket",
```

```
        "key": "WordPress_App.zip",
        "bundleType": "zip"
    },
    {
        "revisionType": "S3",
        "s3Location": {
            "version": "tMk.UxgDpMEVb7V187ZM6wVAWEXAMPLE",
            "bucket": "CodeDeployDemoBucket",
            "key": "WordPress_App_2-0.zip",
            "bundleType": "zip"
        }
    }
]
```

- For API details, see [ListApplicationRevisions](#) in *AWS CLI Command Reference*.

list-applications

The following code example shows how to use `list-applications`.

AWS CLI

To get information about applications

The following `list-applications` example displays information about all applications that are associated with the user's AWS account.

```
aws deploy list-applications
```

Output:

```
{
  "applications": [
    "WordPress_App",
    "MyOther_App"
  ]
}
```

- For API details, see [ListApplications](#) in *AWS CLI Command Reference*.

list-deployment-configs

The following code example shows how to use `list-deployment-configs`.

AWS CLI

To get information about deployment configurations

The following `list-deployment-configs` example displays information about all deployment configurations that are associated with the user's AWS account.

```
aws deploy list-deployment-configs
```

Output:

```
{
  "deploymentConfigsList": [
    "ThreeQuartersHealthy",
    "CodeDeployDefault.AllAtOnce",
    "CodeDeployDefault.HalfAtATime",
    "CodeDeployDefault.OneAtATime"
  ]
}
```

- For API details, see [ListDeploymentConfigs](#) in *AWS CLI Command Reference*.

list-deployment-groups

The following code example shows how to use `list-deployment-groups`.

AWS CLI

To get information about deployment groups

The following `list-deployment-groups` example displays information about all deployment groups that are associated with the specified application.

```
aws deploy list-deployment-groups --application-name WordPress_App
```

Output:

```
{
  "applicationName": "WordPress_App",
  "deploymentGroups": [
    "WordPress_DG",
    "WordPress_Beta_DG"
  ]
}
```

- For API details, see [ListDeploymentGroups](#) in *AWS CLI Command Reference*.

list-deployment-instances

The following code example shows how to use `list-deployment-instances`.

AWS CLI

To get information about deployment instances

The following `list-deployment-instances` example displays information about all deployment instances that are associated with the specified deployment.

```
aws deploy list-deployment-instances \
  --deployment-id d-A1B2C3111 \
  --instance-status-filter Succeeded
```

Output:

```
{
  "instancesList": [
    "i-EXAMPLE11",
    "i-EXAMPLE22"
  ]
}
```

- For API details, see [ListDeploymentInstances](#) in *AWS CLI Command Reference*.

list-deployment-targets

The following code example shows how to use `list-deployment-targets`.

AWS CLI

To retrieve a list of target IDs that are associated with a deployment

The following `list-deployment-targets` example retrieves a list of target IDs associated with deployments that have a status of "Failed" or "InProgress."

```
aws deploy list-deployment-targets \  
  --deployment-id "d-A1B2C3111" \  
  --target-filters "{\"TargetStatus\": [\"Failed\", \"InProgress\"]}"
```

Output:

```
{  
  "targetIds": [  
    "i-0f1558aaf90e5f1f9"  
  ]  
}
```

For more information, see [ListDeploymentTargets](#) in the *AWS CodeDeploy API Reference*.

- For API details, see [ListDeploymentTargets](#) in *AWS CLI Command Reference*.

list-deployments

The following code example shows how to use `list-deployments`.

AWS CLI

To get information about deployments

The following `list-deployments` example displays information about all deployments that are associated with the specified application and deployment group.

```
aws deploy list-deployments \  
  --application-name WordPress_App \  
  --create-time-range start=2014-08-19T00:00:00,end=2014-08-20T00:00:00 \  
  --deployment-group-name WordPress_DG \  
  --include-only-statuses Failed
```

Output:

```
{
  "deployments": [
    "d-EXAMPLE11",
    "d-EXAMPLE22",
    "d-EXAMPLE33"
  ]
}
```

- For API details, see [ListDeployments](#) in *AWS CLI Command Reference*.

list-git-hub-account-token-names

The following code example shows how to use `list-git-hub-account-token-names`.

AWS CLI

To lists the names of stored connections to GitHub accounts

The following `list-git-hub-account-token-names` example lists the names of the stored connections to GitHub accounts for the current AWS user.

```
aws deploy list-git-hub-account-token-names
```

Output:

```
{
  "tokenNameList": [
    "my-first-token",
    "my-second-token",
    "my-third-token"
  ]
}
```

For more information, see [ListGitHubAccountTokenNames](#) in the *AWS CodeDeploy API Reference*.

- For API details, see [ListGitHubAccountTokenNames](#) in *AWS CLI Command Reference*.

list-on-premises-instances

The following code example shows how to use `list-on-premises-instances`.

AWS CLI

To get information about one or more on-premises instances

The following `list-on-premises-instances` example retrieves a list of available on-premises instance names for instances that are registered in AWS CodeDeploy and also have the specified on-premises instance tag associated in AWS CodeDeploy with the instance.

```
aws deploy list-on-premises-instances \
  --registration-status Registered \
  --tag-filters Key=Name,Value=CodeDeployDemo-OnPrem,Type=KEY_AND_VALUE
```

Output:

```
{
  "instanceNames": [
    "AssetTag12010298EX"
  ]
}
```

- For API details, see [ListOnPremisesInstances](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags for a resource (application)

The following `list-tags-for-resource` example lists the tags applied to an application named `testApp` in CodeDeploy.

```
aws deploy list-tags-for-resource \
  --resource-arn arn:aws:codedeploy:us-west-2:111122223333:application:testApp
```

Output:

```
{
  "Tags": [
    {
```

```

        "Key": "Type",
        "Value": "testType"
    },
    {
        "Key": "Name",
        "Value": "testName"
    }
]
}

```

For more information, see [Tagging instances for deployment groups in CodeDeploy](#) in the *AWS CodeDeploy User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

push

The following code example shows how to use push.

AWS CLI

To bundle and deploy an AWS CodeDeploy compatible application revision to Amazon S3

The following push example bundles and deploys an application revision to Amazon S3 and then associates the application revision with the specified application.

```

aws deploy push \
  --application-name WordPress_App \
  --description "This is my deployment" \
  --ignore-hidden-files \
  --s3-location s3://CodeDeployDemoBucket/WordPressApp.zip \
  --source /tmp/MyLocalDeploymentFolder/

```

The output describes how to use the create-deployment command to create a deployment that uses the uploaded application revision.

To deploy with this revision, run:

```

aws deploy create-deployment --application-name WordPress_App
  --deployment-config-name <deployment-config-name> --
deployment-group-name <deployment-group-name> --s3-location
  bucket=CodeDeployDemoBucket,key=WordPressApp.zip,bundleType=zip,eTag="cecc9b8EXAMPLE50a6e71

```

- For API details, see [Push](#) in *AWS CLI Command Reference*.

register-application-revision

The following code example shows how to use `register-application-revision`.

AWS CLI

To register information about an already-uploaded application revision

The following `register-application-revision` example registers information about an already-uploaded application revision stored in Amazon S3 with AWS CodeDeploy.

```
aws deploy register-application-revision \  
  --application-name WordPress_App \  
  --description "Revised WordPress application" \  
  --s3-location  
bucket=CodeDeployDemoBucket,key=RevisedWordPressApp.zip,bundleType=zip,eTag=cecc9b8a08eac65
```

This command produces no output.

- For API details, see [RegisterApplicationRevision](#) in *AWS CLI Command Reference*.

register-on-premises-instance

The following code example shows how to use `register-on-premises-instance`.

AWS CLI

To register an on-premises instance

The following `register-on-premises-instance` example registers an on-premises instance with AWS CodeDeploy. It does not create the specified IAM user, nor does it associate in AWS CodeDeploy any on-premises instances tags with the registered instance.

```
aws deploy register-on-premises-instance \  
  --instance-name AssetTag12010298EX \  
  --iam-user-arn arn:aws:iam::80398EXAMPLE:user/CodeDeployDemoUser-OnPrem
```

This command produces no output.

- For API details, see [RegisterOnPremisesInstance](#) in *AWS CLI Command Reference*.

register

The following code example shows how to use `register`.

AWS CLI

To register an on-premises instance

The following `register` example registers an on-premises instance with AWS CodeDeploy, associates in AWS CodeDeploy the specified on-premises instance tag with the registered instance, and creates an on-premises configuration file that can be copied to the instance. It does not create the IAM user, nor does it install the AWS CodeDeploy Agent on the instance.

```
aws deploy register \  
  --instance-name AssetTag12010298EX \  
  --iam-user-arn arn:aws:iam::80398EXAMPLE:user/CodeDeployUser-OnPrem \  
  --tags Key=Name,Value=CodeDeployDemo-OnPrem \  
  --region us-west-2
```

Output:

```
Registering the on-premises instance... DONE  
Adding tags to the on-premises instance... DONE  
Copy the on-premises configuration file named codedeploy.onpremises.yml to the on-  
premises instance, and run the following command on the on-premises instance to  
install and configure the AWS CodeDeploy Agent:  
aws deploy install --config-file codedeploy.onpremises.yml
```

- For API details, see [Register](#) in *AWS CLI Command Reference*.

remove-tags-from-on-premises-instances

The following code example shows how to use `remove-tags-from-on-premises-instances`.

AWS CLI

To remove tags from one or more on-premises instances

The following `remove-tags-from-on-premises-instances` example disassociates the specified on-premises tags in AWS CodeDeploy from on-premises instances. It does not deregister the on-premises instances in AWS CodeDeploy, nor uninstall the AWS CodeDeploy

Agent from the instance, nor remove the on-premises configuration file from the instances, nor delete the IAM users that are associated with the instances.

```
aws deploy remove-tags-from-on-premises-instances \  
  --instance-names AssetTag12010298EX AssetTag23121309EX \  
  --tags Key=Name,Value=CodeDeployDemo-OnPrem
```

This command produces no output.

- For API details, see [RemoveTagsFromOnPremisesInstances](#) in *AWS CLI Command Reference*.

stop-deployment

The following code example shows how to use stop-deployment.

AWS CLI

To attempt to stop a deployment

The following stop-deployment example attempts to stop an in-progress deployment that is associated with the user's AWS account.

```
aws deploy stop-deployment --deployment-id d-A1B2C3111
```

Output:

```
{  
  "status": "Succeeded",  
  "statusMessage": "No more commands will be scheduled for execution in the  
  deployment instances"  
}
```

- For API details, see [StopDeployment](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To tag a resource (application)

The following `tag-resource` example adds two tags with keys `Name` and `Type`, and values `testName` and `testType` to an application named `testApp` in CodeDeploy:

```
aws deploy tag-resource \  
  --resource-arn arn:aws:codedeploy:us-west-2:111122223333:application:testApp \  
  --tags Key=Name,Value=testName Key=Type,Value=testType
```

If successful, this command produces no output.

For more information, see [Tagging instances for deployment groups in CodeDeploy](#) in the *AWS CodeDeploy User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

uninstall

The following code example shows how to use `uninstall`.

AWS CLI

To uninstall an on-premises instance

The following `uninstall` example uninstalls the AWS CodeDeploy Agent from the on-premises instance and removes the on-premises configuration file from the instance. It doesn't deregister the instance in AWS CodeDeploy, nor disassociate any on-premises instance tags in AWS CodeDeploy from the instance, nor delete the IAM user that is associated with the instance.

```
aws deploy uninstall
```

This command produces no output.

- For API details, see [Uninstall](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from a resource (application)

The following `untag-resource` example removes two tags with keys `Name` and `Type` from an application named `testApp` in CodeDeploy.

```
aws deploy untag-resource \  
  --resource-arn arn:aws:codedeploy:us-west-2:111122223333:application:testApp \  
  --tag-keys Name Type
```

If successful, this command produces no output.

For more information, see [Tagging instances for deployment groups in CodeDeploy](#) in the *AWS CodeDeploy User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-application

The following code example shows how to use `update-application`.

AWS CLI

To change details of an application

The following `update-application` example changes the name of an application that is associated with the user's AWS account.

```
aws deploy update-application \  
  --application-name WordPress_App \  
  --new-application-name My_WordPress_App
```

This command produces no output.

- For API details, see [UpdateApplication](#) in *AWS CLI Command Reference*.

update-deployment-group

The following code example shows how to use `update-deployment-group`.

AWS CLI

To change information about a deployment group

The following `update-deployment-group` example changes the settings of a deployment group that is associated with the specified application.

```
aws deploy update-deployment-group \  
  --application-name WordPress_App \  
  --auto-scaling-groups My_CodeDeployDemo_ASG \  
  --current-deployment-group-name WordPress_DG \  
  --deployment-config-name CodeDeployDefault.AllAtOnce \  
  --ec2-tag-filters Key=Name,Type=KEY_AND_VALUE,Value=My_CodeDeployDemo \  
  --new-deployment-group-name My_WordPress_DepGroup \  
  --service-role-arn arn:aws:iam::80398EXAMPLE:role/CodeDeployDemo-2
```

This command produces no output.

- For API details, see [UpdateDeploymentGroup](#) in *AWS CLI Command Reference*.

CodeGuru Reviewer examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with CodeGuru Reviewer.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-repository

The following code example shows how to use `associate-repository`.

AWS CLI

Example 1: To create a Bitbucket repository association

The following `associate-repository` example creates a repository association using an existing Bitbucket repository.

```
aws codeguru-reviewer associate-repository \  
  --repository 'Bitbucket={Owner=sample-owner, Name=mySampleRepo,  
  ConnectionArn=arn:aws:codestar-connections:us-west-2:123456789012:connection/  
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 }'
```

Output:

```
{  
  "RepositoryAssociation": {  
    "ProviderType": "Bitbucket",  
    "Name": "mySampleRepo",  
    "LastUpdatedTimeStamp": 1596216896.979,  
    "AssociationId": "association:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",  
    "CreatedTimeStamp": 1596216896.979,  
    "ConnectionArn": "arn:aws:codestar-connections:us-  
west-2:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
    "State": "Associating",  
    "StateReason": "Pending Repository Association",  
    "AssociationArn": "arn:aws:codeguru-reviewer:us-  
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",  
    "Owner": "sample-owner"  
  }  
}
```

For more information, see [Create a Bitbucket repository association in Amazon CodeGuru Reviewer](#) in the *Amazon CodeGuru Reviewer User Guide*.

Example 2: To create a GitHub Enterprise repository association

The following `associate-repository` example creates a repository association using an existing GitHub Enterprise repository.

```
aws codeguru-reviewer associate-repository \  
  --repository 'GitHubEnterprise={Owner=sample-owner, Name=mySampleRepo,  
  ConnectionArn=arn:aws:codestar-connections:us-west-2:123456789012:connection/  
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 }'
```

```
--repository 'GitHubEnterpriseServer={Owner=sample-owner, Name=mySampleRepo,
ConnectionArn=arn:aws:codestar-connections:us-west-2:123456789012:connection/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 }'
```

Output:

```
{
  "RepositoryAssociation": {
    "ProviderType": "GitHubEnterpriseServer",
    "Name": "mySampleRepo",
    "LastUpdatedTimeStamp": 1596216896.979,
    "AssociationId": "association:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "CreatedTimeStamp": 1596216896.979,
    "ConnectionArn": "arn:aws:codestar-connections:us-
west-2:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "State": "Associating",
    "StateReason": "Pending Repository Association",
    "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "Owner": "sample-owner"
  }
}
```

For more information, see [Create a GitHub Enterprise Server repository association in Amazon CodeGuru Reviewer](#) in the *Amazon CodeGuru Reviewer User Guide*.

Example 3: To create an AWS CodeCommit repository association

The following `associate-repository` example creates a repository association using an existing AWS CodeCommit repository.

```
aws codeguru-reviewer associate-repository \
  --repository CodeCommit={Name=mySampleRepo}
```

Output:

```
{
  "RepositoryAssociation": {
    "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "Name": "My-ecs-beta-repo",
    "LastUpdatedTimeStamp": 1595634764.029,
    "ProviderType": "CodeCommit",
```

```

    "CreatedTimeStamp": 1595634764.029,
    "Owner": "544120495673",
    "State": "Associating",
    "StateReason": "Pending Repository Association",
    "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:544120495673:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
  }
}

```

For more information, see [Create an AWS CodeCommit repository association in Amazon CodeGuru Reviewer](#) in the *Amazon CodeGuru Reviewer User Guide*.

- For API details, see [AssociateRepository](#) in *AWS CLI Command Reference*.

create-code-review

The following code example shows how to use `create-code-review`.

AWS CLI

To create a code review.

The following `create-code-review` creates a review of code in the `mainline` branch of an AWS CodeCommit repository that is named `my-repository-name`.

```

aws codeguru-reviewer create-code-review \
  --name my-code-review \
  --repository-association-arn arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
  --type '{"RepositoryAnalysis": {"RepositoryHead": {"BranchName": "mainline"}}}'

```

Output:

```

{
  "CodeReview": {
    "Name": "my-code-review",
    "CodeReviewArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222:code-
review:RepositoryAnalysis-my-code-review",
    "RepositoryName": "my-repository-name",
    "Owner": "123456789012",
    "ProviderType": "CodeCommit",

```

```

    "State": "Pending",
    "StateReason": "CodeGuru Reviewer has received the request, and a code
review is scheduled.",
    "CreatedTimeStamp": 1618873489.195,
    "LastUpdatedTimeStamp": 1618873489.195,
    "Type": "RepositoryAnalysis",
    "SourceCodeType": {
      "RepositoryHead": {
        "BranchName": "mainline"
      }
    },
    "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
  }
}

```

For more information, see [Create code reviews in Amazon CodeGuru Reviewer](#) in the *Amazon CodeGuru Reviewer User Guide*.

- For API details, see [CreateCodeReview](#) in *AWS CLI Command Reference*.

describe-code-review

The following code example shows how to use `describe-code-review`.

AWS CLI

List details about a code review.

The following `describe-code-review` lists information about a review of code in the "mainline" branch of an AWS CodeCommit repository that is named "my-repo-name".

```

aws codeguru-reviewer put-recommendation-feedback \
  --code-review-arn arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111:code-
review:RepositoryAnalysis-my-repository-name-branch-abcdefg12345678 \
  --recommendation-id
3be1b2e5d7ef6e298a06499379ee290c9c596cf688fdcadb08285ddb0dd390eb \
  --reactions ThumbsUp

```

Output

```
{
```

```

    "CodeReview": {
      "Name": "My-ecs-beta-repo-master-xs6di4kfd4j269dz",
      "CodeReviewArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222:code-
review:RepositoryAnalysis-my-repo-name",
      "RepositoryName": "My-ecs-beta-repo",
      "Owner": "123456789012",
      "ProviderType": "CodeCommit",
      "State": "Pending",
      "StateReason": "CodeGuru Reviewer is reviewing the source code.",
      "CreatedTimeStamp": 1618874226.226,
      "LastUpdatedTimeStamp": 1618874233.689,
      "Type": "RepositoryAnalysis",
      "SourceCodeType": {
        "RepositoryHead": {
          "BranchName": "mainline"
        }
      },
      "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    }
  }
}

```

For more information, see [View code review details](#) in the *Amazon CodeGuru Reviewer User Guide*.

- For API details, see [DescribeCodeReview](#) in *AWS CLI Command Reference*.

describe-recommendation-feedback

The following code example shows how to use `describe-recommendation-feedback`.

AWS CLI

To view information about feedback on a recommendation

The following `describe-recommendation-feedback` displays information about feedback on a recommendation. This recommendation has one `ThumbsUp` reaction.

```

aws codeguru-reviewer describe-recommendation-feedback \
  --code-review-arn arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111:code-
review:RepositoryAnalysis-my-repository-name-branch-abcdefgh12345678 \

```

```
--recommendation-id
3be1b2e5d7ef6e298a06499379ee290c9c596cf688fdcadb08285ddb0dd390eb
```

Output:

```
{
  "RecommendationFeedback": {
    "CodeReviewArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111:code-
review:RepositoryAnalysis-my-repository-name-branch-abcdefgh12345678",
    "RecommendationId":
"3be1b2e5d7ef6e298a06499379ee290c9c596cf688fdcadb08285ddb0dd390eb",
    "Reactions": [
      "ThumbsUp"
    ],
    "UserId": "aws-user-id",
    "CreatedTimeStamp": 1618877070.313,
    "LastUpdatedTimeStamp": 1618877948.881
  }
}
```

For more information, see [View recommendations and provide feedback](#) and [Step 4: Provide feedback](#) in the *Amazon CodeGuru Reviewer User Guide*.

- For API details, see [DescribeRecommendationFeedback](#) in *AWS CLI Command Reference*.

describe-repository-association

The following code example shows how to use `describe-repository-association`.

AWS CLI

Example 1: To return information about a GitHub repository association

The following `describe-repository-association` example returns information about a repository association that uses a GitHub Enterprise repository and is in the Associated state.

```
aws codeguru-reviewer describe-repository-association \
  --association-arn arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:


```
{
  "RepositoryAssociation": {
    "AssociationId": "b822717e-0711-4e8a-bada-0e738289c75e",
    "Name": "mySampleRepo",
    "LastUpdatedTimeStamp": 1588102637.649,
    "ProviderType": "GitHub",
    "CreatedTimeStamp": 1588102615.636,
    "Owner": "sample-owner",
    "State": "Associated",
    "StateReason": "Pull Request Notification configuration successful",
    "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
  }
}
```

For more information, see [Create a GitHub Enterprise Server repository association in Amazon CodeGuru Reviewer](#) in the *Amazon CodeGuru Reviewer User Guide*.

Example 2: To return information about a failed repository association

The following describe-repository-association example returns information about a repository association that uses a GitHub Enterprise repository and is in the Failed state.

```
aws codeguru-reviewer describe-repository-association \
  --association-arn arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{
  "RepositoryAssociation": {
    "ProviderType": "GitHubEnterpriseServer",
    "Name": "mySampleRepo",
    "LastUpdatedTimeStamp": 1596217036.892,
    "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "CreatedTimeStamp": 1596216896.979,
    "ConnectionArn": "arn:aws:codestar-connections:us-
west-2:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "State": "Failed",
    "StateReason": "Failed, Please retry.",
    "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
```

```
    "Owner": "sample-owner"
  }
}
```

For more information, see [Create a GitHub Enterprise Server repository association in Amazon CodeGuru Reviewer](#) in the *Amazon CodeGuru Reviewer User Guide*.

Example 3: To return information about a disassociating repository association

The following `describe-repository-association` example returns information about a repository association that uses a GitHub Enterprise repository and is in the Disassociating state.

```
aws codeguru-reviewer describe-repository-association \
  --association-arn arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{
  "RepositoryAssociation": {
    "ProviderType": "GitHubEnterpriseServer",
    "Name": "mySampleRepo",
    "LastUpdatedTimeStamp": 1596217036.892,
    "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "CreatedTimeStamp": 1596216896.979,
    "ConnectionArn": "arn:aws:codestar-connections:us-
west-2:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "State": "Disassociating",
    "StateReason": "Source code access removal in progress",
    "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "Owner": "sample-owner"
  }
}
```

For more information, see [Create a GitHub Enterprise Server repository association in Amazon CodeGuru Reviewer](#) in the *Amazon CodeGuru Reviewer User Guide*.

- For API details, see [DescribeRepositoryAssociation](#) in *AWS CLI Command Reference*.

disassociate-repository

The following code example shows how to use `disassociate-repository`.

AWS CLI

To disassociate a repository association

The following `disassociate-repository` disassociates a repository association that is using an AWS CodeCommit repository.

```
aws codeguru-reviewer disassociate-repository \  
  --association-arn arn:aws:codeguru-reviewer:us-  
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{  
  "RepositoryAssociation": {  
    "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
    "AssociationArn": "arn:aws:codeguru-reviewer:us-  
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
    "Name": "my-repository",  
    "Owner": "123456789012",  
    "ProviderType": "CodeCommit",  
    "State": "Disassociating",  
    "LastUpdatedTimeStamp": 1618939174.759,  
    "CreatedTimeStamp": 1595636947.096  
  },  
  "Tags": {  
    "Status": "Secret",  
    "Team": "Saanvi"  
  }  
}
```

For more information, see [Disassociate a repository in CodeGuru Reviewer](#) in the *Amazon CodeGuru Reviewer User Guide*.

- For API details, see [DisassociateRepository](#) in *AWS CLI Command Reference*.

list-code-reviews

The following code example shows how to use `list-code-reviews`.

AWS CLI

To list code reviews created in your AWS account in the last 90 days.

The following `list-code-reviews` example lists the code reviews created in the last 90 days using pull requests.

```
aws codeguru-reviewer list-code-reviews \  
  --type PullRequest
```

Output:

```
{  
  "CodeReviewSummaries": [  
    {  
      "LastUpdatedTimeStamp": 1588897288.054,  
      "Name": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
      "ProviderType": "GitHub",  
      "PullRequestId": "5",  
      "MetricsSummary": {  
        "MeteredLinesOfCodeCount": 24,  
        "FindingsCount": 1  
      },  
      "CreatedTimeStamp": 1588897068.512,  
      "State": "Completed",  
      "CodeReviewArn": "arn:aws:codeguru-reviewer:us-west-2:123456789012:code-  
review:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
      "Owner": "sample-owner",  
      "RepositoryName": "sample-repository-name",  
      "Type": "PullRequest"  
    },  
    {  
      "LastUpdatedTimeStamp": 1588869793.263,  
      "Name": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",  
      "ProviderType": "GitHub",  
      "PullRequestId": "4",  
      "MetricsSummary": {  
        "MeteredLinesOfCodeCount": 29,  
        "FindingsCount": 0  
      },  
      "CreatedTimeStamp": 1588869575.949,  
      "State": "Completed",
```

```
    "CodeReviewArn": "arn:aws:codeguru-reviewer:us-west-2:123456789012:code-
review:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "Owner": "sample-owner",
    "RepositoryName": "sample-repository-name",
    "Type": "PullRequest"
  },
  {
    "LastUpdatedTimeStamp": 1588870511.211,
    "Name": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "ProviderType": "GitHub",
    "PullRequestId": "4",
    "MetricsSummary": {
      "MeteredLinesOfCodeCount": 2,
      "FindingsCount": 0
    },
    "CreatedTimeStamp": 1588870292.425,
    "State": "Completed",
    "CodeReviewArn": "arn:aws:codeguru-reviewer:us-west-2:123456789012:code-
review:a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "Owner": "sample-owner",
    "RepositoryName": "sample-repository-name",
    "Type": "PullRequest"
  },
  {
    "LastUpdatedTimeStamp": 1588118522.452,
    "Name": "a1b2c3d4-5678-90ab-cdef-EXAMPLE44444",
    "ProviderType": "GitHub",
    "PullRequestId": "3",
    "MetricsSummary": {
      "MeteredLinesOfCodeCount": 29,
      "FindingsCount": 0
    },
    "CreatedTimeStamp": 1588118301.131,
    "State": "Completed",
    "CodeReviewArn": "arn:aws:codeguru-reviewer:us-west-2:123456789012:code-
review:a1b2c3d4-5678-90ab-cdef-EXAMPLE44444",
    "Owner": "sample-owner",
    "RepositoryName": "sample-repository-name",
    "Type": "PullRequest"
  },
  {
    "LastUpdatedTimeStamp": 1588112205.207,
    "Name": "a1b2c3d4-5678-90ab-cdef-EXAMPLE55555",
    "ProviderType": "GitHub",
```

```

    "PullRequestId": "2",
    "MetricsSummary": {
      "MeteredLinesOfCodeCount": 25,
      "FindingsCount": 0
    },
    "CreatedTimeStamp": 1588111987.443,
    "State": "Completed",
    "CodeReviewArn": "arn:aws:codeguru-reviewer:us-west-2:123456789012:code-
review:a1b2c3d4-5678-90ab-cdef-EXAMPLE55555",
    "Owner": "sample-owner",
    "RepositoryName": "sample-repository-name",
    "Type": "PullRequest"
  },
  {
    "LastUpdatedTimeStamp": 1588104489.981,
    "Name": "a1b2c3d4-5678-90ab-cdef-EXAMPLE66666",
    "ProviderType": "GitHub",
    "PullRequestId": "1",
    "MetricsSummary": {
      "MeteredLinesOfCodeCount": 25,
      "FindingsCount": 0
    },
    "CreatedTimeStamp": 1588104270.223,
    "State": "Completed",
    "CodeReviewArn": "arn:aws:codeguru-reviewer:us-west-2:123456789012:code-
review:a1b2c3d4-5678-90ab-cdef-EXAMPLE66666",
    "Owner": "sample-owner",
    "RepositoryName": "sample-repository-name",
    "Type": "PullRequest"
  }
]
}

```

For more information, see [View all code reviews](#) in the *Amazon CodeGuru Reviewer User Guide*.

- For API details, see [ListCodeReviews](#) in *AWS CLI Command Reference*.

list-recommendation-feedback

The following code example shows how to use `list-recommendation-feedback`.

AWS CLI

To list customer recommendation feedback for a recommendation on an associated repository

The following `list-recommendation-feedback` Lists customer feedback on all recommendations on a code review. This code review has one piece of feedback, a "ThumbsUp", from a customer.

```
aws codeguru-reviewer list-recommendation-feedback \
  --code-review-arn arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111:code-
review:RepositoryAnalysis-my-repository-name-branch-abcdefgh12345678
```

Output:

```
{
  "RecommendationFeedbackSummaries": [
    {
      "RecommendationId":
"3be1b2e5d7ef6e298a06499379ee290c9c596cf688fdcadb08285ddb0dd390eb",
      "Reactions": [
        "ThumbsUp"
      ],
      "UserId": "aws-user-id"
    }
  ]
}
```

For more information, see [Step 4: Provide feedback](#) in the *Amazon CodeGuru Reviewer User Guide*.

- For API details, see [ListRecommendationFeedback](#) in *AWS CLI Command Reference*.

list-recommendations

The following code example shows how to use `list-recommendations`.

AWS CLI

To list the recommendations for a completed code review

The following `list-recommendations` example lists the recommendations for a completed code review. This code review has one recommendations.

```
aws codeguru-reviewer list-recommendations \
  --code-review-arn arn:aws:codeguru-reviewer:us-west-2:544120495673:code-
  review:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{
  "RecommendationSummaries": [
    {
      "Description": "\n\nProblem  \n You are using a `ConcurrentHashMap`,
      but your usage of `containsKey()` and `get()` may not be thread-safe at lines: **63
      and 64**. In between the check and the `get()` another thread can remove the key
      and the `get()` will return `null`. The remove that can remove the key is at line:
      **59**.\n\nFix  \n Consider calling `get()`, checking instead of your current
      check if the returned object is `null`, and then using that object only, without
      calling `get()` again.\n\nMore info  \n [View an example on GitHub](https://
      github.com/apache/hadoop/blob/f16cf877e565084c66bc63605659b157c4394dc8/hadoop-tools/
      hadoop-aws/src/main/java/org/apache/hadoop/fs/s3a/s3guard/S3Guard.java#L302-L304)
      (external link).",
      "RecommendationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
      "StartLine": 63,
      "EndLine": 64,
      "FilePath": "src/main/java/com/company/sample/application/
      CreateOrderThread.java"
    }
  ]
}
```

For more information, see [Step 4: Provide feedback](#) in the *Amazon CodeGuru Reviewer User Guide*.

- For API details, see [ListRecommendations](#) in *AWS CLI Command Reference*.

list-repository-associations

The following code example shows how to use `list-repository-associations`.

AWS CLI

To list the repository associations in your AWS account

The following `list-repository-associations` example returns a list of repository association summary objects in your account. You can filter the returned list by `ProviderType`, `Name`, `State`, and `Owner`.

```
aws codeguru-reviewer list-repository-associations
```

Output:

```
{
  "RepositoryAssociationSummaries": [
    {
      "LastUpdatedTimeStamp": 1595886609.616,
      "Name": "test",
      "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "Owner": "sample-owner",
      "State": "Associated",
      "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "ProviderType": "Bitbucket"
    },
    {
      "LastUpdatedTimeStamp": 1595636969.035,
      "Name": "CodeDeploy-CodePipeline-ECS-Tutorial",
      "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
      "Owner": "123456789012",
      "State": "Associated",
      "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
      "ProviderType": "CodeCommit"
    },
    {
      "LastUpdatedTimeStamp": 1595634785.983,
      "Name": "My-ecs-beta-repo",
      "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
      "Owner": "123456789012",
      "State": "Associated",
      "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    }
  ]
}
```

```

        "ProviderType": "CodeCommit"
    },
    {
        "LastUpdatedTimeStamp": 1590712811.77,
        "Name": "MyTestCodeCommit",
        "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE44444",
        "Owner": "123456789012",
        "State": "Associated",
        "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE44444",
        "ProviderType": "CodeCommit"
    },
    {
        "LastUpdatedTimeStamp": 1588102637.649,
        "Name": "aws-codeguru-profiler-sample-application",
        "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE55555",
        "Owner": "sample-owner",
        "State": "Associated",
        "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE55555",
        "ProviderType": "GitHub"
    },
    {
        "LastUpdatedTimeStamp": 1588028233.995,
        "Name": "codeguru-profiler-demo-app",
        "AssociationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE66666",
        "Owner": "sample-owner",
        "State": "Associated",
        "AssociationArn": "arn:aws:codeguru-reviewer:us-
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE66666",
        "ProviderType": "GitHub"
    }
]
}

```

For more information, see [View all repository associations in CodeGuru Reviewer](#) in the *Amazon CodeGuru Reviewer User Guide*.

- For API details, see [ListRepositoryAssociations](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags on an associated repository

The following `list-tags-for-resource` lists the tags on an associated repository. This associated repository has two tags.

```
aws codeguru-reviewer list-tags-for-resource \  
  --resource-arn arn:aws:codeguru-reviewer:us-  
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{  
  "Tags": {  
    "Status": "Secret",  
    "Team": "Saanvi"  
  }  
}
```

For more information, see [View tags for a CodeGuru Reviewer associated repository \(AWS CLI\)](#) in the *Amazon CodeGuru Reviewer User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

put-recommendation-feedback

The following code example shows how to use `put-recommendation-feedback`.

AWS CLI

To add a recommendation to a code review

The following `put-recommendation-feedback` puts a ThumbsUp recommendation on a code review.

```
aws codeguru-reviewer put-recommendation-feedback \  
  --code-review-arn \arn:aws:codeguru-reviewer:us-  
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111:code-  
review:RepositoryAnalysis-my-repository-name-branch-abcdefg12345678 \  
  --recommendation-id  
3be1b2e5d7ef6e298a06499379ee290c9c596cf688fdcadb08285ddb0dd390eb \  
 \
```

```
--reactions ThumbsUp
```

This command produces no output.

For more information, see [Step 4: Provide feedback](#) in the *Amazon CodeGuru Reviewer User Guide*.

- For API details, see [PutRecommendationFeedback](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add a tag to an associated repository

The following `tag-resource` adds two tags to an associated repository

```
aws codeguru-reviewer tag-resource \  
  --resource-arn arn:aws:codeguru-reviewer:us-  
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \  
  --tags Status=Secret,Team=Saanvi
```

This command produces no output.

For more information, see [Add a tag to a CodeGuru Reviewer associated repository \(AWS CLI\)](#) and [Add or update tags for a CodeGuru Reviewer associated repository \(AWS CLI\)](#) in the *Amazon CodeGuru Reviewer User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To untag an associated repository

The following `untag-resource` removes two tags with keys "Secret" and "Team" from an associated repository.

```
aws codeguru-reviewer untag-resource \  
  --resource-arn arn:aws:codeguru-reviewer:us-  
west-2:123456789012:association:a1b2c3d4-5678-90ab-cdef-EXAMPLE111111 \  
  --tag-keys Status Team
```

This command produces no output.

For more information, see [Remove tags from a CodeGuru Reviewer associated repository \(AWS CLI\)](#) in the *Amazon CodeGuru Reviewer User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

CodePipeline examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with CodePipeline.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

acknowledge-job

The following code example shows how to use `acknowledge-job`.

AWS CLI

To retrieve information about a specified job

This example returns information about a specified job, including the status of that job if it exists. This is only used for job workers and custom actions. To determine the value of nonce and the job ID, use `aws codepipeline poll-for-jobs`.

Command:

```
aws codepipeline acknowledge-job --job-id f4f4ff82-2d11-EXAMPLE --nonce 3
```

Output:

```
{
  "status": "InProgress"
}
```

- For API details, see [AcknowledgeJob](#) in *AWS CLI Command Reference*.

create-custom-action-type

The following code example shows how to use `create-custom-action-type`.

AWS CLI

To create a custom action

This example creates a custom action for AWS CodePipeline using an already-created JSON file (here named `MyCustomAction.json`) that contains the structure of the custom action. For more information about the requirements for creating a custom action, including the structure of the file, see the *AWS CodePipeline User Guide*.

```
aws codepipeline create-custom-action-type --cli-input-json file://
MyCustomAction.json
```

Contents of JSON file `MyCustomAction.json`:

```
{
  "category": "Build",
  "provider": "MyJenkinsProviderName",
  "version": "1",
  "settings": {
    "entityUrlTemplate": "https://192.0.2.4/job/{Config:ProjectName}/",
  }
}
```

```
    "executionUrlTemplate": "https://192.0.2.4/job/{Config:ProjectName}/  
lastSuccessfulBuild/{ExternalExecutionId}/"  
  },  
  "configurationProperties": [  
    {  
      "name": "MyJenkinsExampleBuildProject",  
      "required": true,  
      "key": true,  
      "secret": false,  
      "queryable": false,  
      "description": "The name of the build project must be provided when this  
action is added to the pipeline.",  
      "type": "String"  
    }  
  ],  
  "inputArtifactDetails": {  
    "maximumCount": 1,  
    "minimumCount": 0  
  },  
  "outputArtifactDetails": {  
    "maximumCount": 1,  
    "minimumCount": 0  
  }  
}
```

This command returns the structure of the custom action.

- For API details, see [CreateCustomActionType](#) in *AWS CLI Command Reference*.

create-pipeline

The following code example shows how to use `create-pipeline`.

AWS CLI

To create a pipeline

This example creates a pipeline in AWS CodePipeline using an already-created JSON file (here named `MySecondPipeline.json`) that contains the structure of the pipeline. For more information about the requirements for creating a pipeline, including the structure of the file, see the *AWS CodePipeline User Guide*.

Command:

```
aws codepipeline create-pipeline --cli-input-json file://MySecondPipeline.json
```

JSON file sample contents:

```
{
  "pipeline": {
    "roleArn": "arn:aws:iam::111111111111:role/AWS-CodePipeline-Service",
    "stages": [
      {
        "name": "Source",
        "actions": [
          {
            "inputArtifacts": [],
            "name": "Source",
            "actionTypeId": {
              "category": "Source",
              "owner": "AWS",
              "version": "1",
              "provider": "S3"
            },
            "outputArtifacts": [
              {
                "name": "MyApp"
              }
            ],
            "configuration": {
              "S3Bucket": "awscodepipeline-demo-bucket",
              "S3ObjectKey": "aws-codepipeline-s3-aws-codedeploy_linux.zip"
            },
            "runOrder": 1
          }
        ]
      },
      {
        "name": "Beta",
        "actions": [
          {
            "inputArtifacts": [
              {
                "name": "MyApp"
              }
            ],
            "name": "CodePipelineDemoFleet",
```



```
    "actionTypeId": {
      "category": "Deploy",
      "owner": "AWS",
      "version": "1",
      "provider": "CodeDeploy"
    },
    "outputArtifacts": [],
    "configuration": {
      "ApplicationName": "CodePipelineDemoApplication",
      "DeploymentGroupName": "CodePipelineDemoFleet"
    },
    "runOrder": 1
  }
]
}
],
"artifactStore": {
  "type": "S3",
  "location": "codepipeline-us-east-1-11EXAMPLE11"
},
"name": "MySecondPipeline",
"version": 1
}
}
```

Output:

This command returns the structure of the pipeline.

- For API details, see [CreatePipeline](#) in *AWS CLI Command Reference*.

delete-custom-action-type

The following code example shows how to use `delete-custom-action-type`.

AWS CLI

To delete a custom action

This example deletes a custom action in AWS CodePipeline by using an already-created JSON file (here named `DeleteMyCustomAction.json`) that contains the action type, provider name,

and version number of the action to be deleted. Use the `list-action-types` command to view the correct values for category, version, and provider.

Command:

```
aws codepipeline delete-custom-action-type --cli-input-json file://
DeleteMyCustomAction.json
```

JSON file sample contents:

```
{
  "category": "Build",
  "version": "1",
  "provider": "MyJenkinsProviderName"
}
```

Output:

```
None.
```

- For API details, see [DeleteCustomActionType](#) in *AWS CLI Command Reference*.

delete-pipeline

The following code example shows how to use `delete-pipeline`.

AWS CLI

To delete a pipeline

This example deletes a pipeline named `MySecondPipeline` from AWS CodePipeline. Use the `list-pipelines` command to view a list of pipelines associated with your AWS account.

Command:

```
aws codepipeline delete-pipeline --name MySecondPipeline
```

Output:

```
None.
```

- For API details, see [DeletePipeline](#) in *AWS CLI Command Reference*.

delete-webhook

The following code example shows how to use `delete-webhook`.

AWS CLI

To delete a webhook

The following `delete-webhook` example deletes a webhook for a GitHub version 1 source action. You must use the `deregister-webhook-with-third-party` command to deregister the webhook before you delete it.

```
aws codepipeline delete-webhook \  
  --name my-webhook
```

This command produces no output.

For more information, see [Delete the webhook for your GitHub source](#) in the *AWS CodePipeline User Guide*.

- For API details, see [DeleteWebhook](#) in *AWS CLI Command Reference*.

deregister-webhook-with-third-party

The following code example shows how to use `deregister-webhook-with-third-party`.

AWS CLI

To deregister a webhook

The following `deregister-webhook-with-third-party` example deletes a webhook for a GitHub version 1 source action. You must deregister the webhook before you delete it.

```
aws codepipeline deregister-webhook-with-third-party \  
  --webhook-name my-webhook
```

This command produces no output.

For more information, see [Delete the webhook for your GitHub source](#) in the *AWS CodePipeline User Guide*.

- For API details, see [DeregisterWebhookWithThirdParty](#) in *AWS CLI Command Reference*.

disable-stage-transition

The following code example shows how to use `disable-stage-transition`.

AWS CLI

To disable a transition to a stage in a pipeline

This example disables transitions into the Beta stage of the MyFirstPipeline pipeline in AWS CodePipeline.

Command:

```
aws codepipeline disable-stage-transition --pipeline-name MyFirstPipeline --stage-name Beta --transition-type Inbound
```

Output:

```
None.
```

- For API details, see [DisableStageTransition](#) in *AWS CLI Command Reference*.

enable-stage-transition

The following code example shows how to use `enable-stage-transition`.

AWS CLI

To enable a transition to a stage in a pipeline

This example enables transitions into the Beta stage of the MyFirstPipeline pipeline in AWS CodePipeline.

Command:

```
aws codepipeline enable-stage-transition --pipeline-name MyFirstPipeline --stage-name Beta --transition-type Inbound
```

Output:

```
None.
```

- For API details, see [EnableStageTransition](#) in *AWS CLI Command Reference*.

get-job-details

The following code example shows how to use `get-job-details`.

AWS CLI**To get details of a job**

This example returns details about a job whose ID is represented by `f4f4ff82-2d11-EXAMPLE`. This command is only used for custom actions. When this command is called, AWS CodePipeline returns temporary credentials for the Amazon S3 bucket used to store artifacts for the pipeline, if required for the custom action. This command will also return any secret values defined for the action, if any are defined.

Command:

```
aws codepipeline get-job-details --job-id f4f4ff82-2d11-EXAMPLE
```

Output:

```
{
  "jobDetails": {
    "accountId": "111111111111",
    "data": {
      "actionConfiguration": {
        "__type": "ActionConfiguration",
        "configuration": {
          "ProjectName": "MyJenkinsExampleTestProject"
        }
      },
      "actionTypeId": {
        "__type": "ActionTypeId",
        "category": "Test",
        "owner": "Custom",
```

```

    "provider": "MyJenkinsProviderName",
    "version": "1"
  },
  "artifactCredentials": {
    "__type": "AWSSessionCredentials",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "secretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY",
    "sessionToken":
    "fICCQD6m7oRw0uX0jANBqkqhkiG9w0BAQUFADCBiDELMaKGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwd
+a4GmWIWJ21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEI03IyNoH/
f0wYK8m9TrDHudUZg3qX4waLG5M43q7Wgc/
MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpEIbb30hjZnzcvcQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQ
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0FkbFFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs
  },
  "inputArtifacts": [
    {
      "__type": "Artifact",
      "location": {
        "s3Location": {
          "bucketName": "codepipeline-us-east-1-11EXAMPLE11",
          "objectKey": "MySecondPipeline/MyAppBuild/EXAMPLE"
        },
        "type": "S3"
      },
      "name": "MyAppBuild"
    }
  ],
  "outputArtifacts": [],
  "pipelineContext": {
    "__type": "PipelineContext",
    "action": {
      "name": "MyJenkinsTest-Action"
    },
    "pipelineName": "MySecondPipeline",
    "stage": {
      "name": "Testing"
    }
  }
},
"id": "f4f4ff82-2d11-EXAMPLE"
}
}

```

- For API details, see [GetJobDetails](#) in *AWS CLI Command Reference*.

get-pipeline-state

The following code example shows how to use `get-pipeline-state`.

AWS CLI

To get information about the state of a pipeline

This example returns the most recent state of a pipeline named `MyFirstPipeline`.

Command:

```
aws codepipeline get-pipeline-state --name MyFirstPipeline
```

Output:

```
{
  "created": 1446137312.204,
  "pipelineName": "MyFirstPipeline",
  "pipelineVersion": 1,
  "stageStates": [
    {
      "actionStates": [
        {
          "actionName": "Source",
          "entityUrl": "https://console.aws.amazon.com/s3/home?#",
          "latestExecution": {
            "lastStatusChange": 1446137358.328,
            "status": "Succeeded"
          }
        }
      ],
      "stageName": "Source"
    },
    {
      "actionStates": [
        {
          "actionName": "CodePipelineDemoFleet",
          "entityUrl": "https://console.aws.amazon.com/codedeploy/home?#/applications/CodePipelineDemoApplication/deployment-groups/CodePipelineDemoFleet",
          "latestExecution": {
            "externalExecutionId": "d-EXAMPLE",
            "externalExecutionUrl": "https://console.aws.amazon.com/codedeploy/home?#/deployments/d-EXAMPLE",

```

```

        "lastStatusChange": 1446137493.131,
        "status": "Succeeded",
        "summary": "Deployment Succeeded"
    }
}
],
"inboundTransitionState": {
    "enabled": true
},
"stageName": "Beta"
}
],
"updated": 1446137312.204
}

```

- For API details, see [GetPipelineState](#) in *AWS CLI Command Reference*.

get-pipeline

The following code example shows how to use get-pipeline.

AWS CLI

To view the structure of a pipeline

This example returns the structure of a pipeline named MyFirstPipeline.

Command:

```
aws codepipeline get-pipeline --name MyFirstPipeline
```

Output:

```

{
  "pipeline": {
    "roleArn": "arn:aws:iam::111111111111:role/AWS-CodePipeline-Service",
    "stages": [
      {
        "name": "Source",
        "actions": [
          {
            "inputArtifacts": [],
            "name": "Source",

```



```
        "actionTypeId": {
            "category": "Source",
            "owner": "AWS",
            "version": "1",
            "provider": "S3"
        },
        "outputArtifacts": [
            {
                "name": "MyApp"
            }
        ],
        "configuration": {
            "S3Bucket": "awscodepipeline-demo-bucket",
            "S3ObjectKey": "aws-codepipeline-s3-aws-
codedeploy_linux.zip"
        },
        "runOrder": 1
    }
]
},
{
    "name": "Beta",
    "actions": [
        {
            "inputArtifacts": [
                {
                    "name": "MyApp"
                }
            ],
            "name": "CodePipelineDemoFleet",
            "actionTypeId": {
                "category": "Deploy",
                "owner": "AWS",
                "version": "1",
                "provider": "CodeDeploy"
            },
            "outputArtifacts": [],
            "configuration": {
                "ApplicationName": "CodePipelineDemoApplication",
                "DeploymentGroupName": "CodePipelineDemoFleet"
            },
            "runOrder": 1
        }
    ]
}
```

```
    }
  ],
  "artifactStore": {
    "type": "S3",
    "location": "codepipeline-us-east-1-11EXAMPLE11"
  },
  "name": "MyFirstPipeline",
  "version": 1
}
}
```

- For API details, see [GetPipeline](#) in *AWS CLI Command Reference*.

list-action-executions

The following code example shows how to use `list-action-executions`.

AWS CLI

To list action executions

The following `list-action-executions` example views action execution details for a pipeline, such as action execution ID, input artifacts, output artifacts, execution result, and status.

```
aws codepipeline list-action-executions \
  --pipeline-name myPipeline
```

Output:

```
{
  "actionExecutionDetails": [
    {
      "pipelineExecutionId": "EXAMPLE0-adfc-488e-bf4c-1111111720d3",
      "actionExecutionId": "EXAMPLE4-2ee8-4853-bd6a-111111158148",
      "pipelineVersion": 12,
      "stageName": "Deploy",
      "actionName": "Deploy",
      "startTime": 1598572628.6,
      "lastUpdateTime": 1598572661.255,
      "status": "Succeeded",
      "input": {
```

```
    "actionTypeId": {
      "category": "Deploy",
      "owner": "AWS",
      "provider": "CodeDeploy",
      "version": "1"
    },
    "configuration": {
      "ApplicationName": "my-application",
      "DeploymentGroupName": "my-deployment-group"
    },
    "resolvedConfiguration": {
      "ApplicationName": "my-application",
      "DeploymentGroupName": "my-deployment-group"
    },
    "region": "us-east-1",
    "inputArtifacts": [
      {
        "name": "SourceArtifact",
        "s3location": {
          "bucket": "artifact-bucket",
          "key": "myPipeline/SourceArti/key"
        }
      }
    ],
    "namespace": "DeployVariables"
  },
  "output": {
    "outputArtifacts": [],
    "executionResult": {
      "externalExecutionId": "d-EXAMPLEE5",
      "externalExecutionSummary": "Deployment Succeeded",
      "externalExecutionUrl": "https://myaddress.com"
    },
    "outputVariables": {}
  }
},
{
  "pipelineExecutionId": "EXAMPLE0-adfc-488e-bf4c-1111111720d3",
  "actionExecutionId": "EXAMPLE5-abb4-4192-9031-11111113a7b0",
  "pipelineVersion": 12,
  "stageName": "Source",
  "actionName": "Source",
  "startTime": 1598572624.387,
  "lastUpdateTime": 1598572628.16,
```

```
"status": "Succeeded",
"input": {
  "actionTypeId": {
    "category": "Source",
    "owner": "AWS",
    "provider": "CodeCommit",
    "version": "1"
  },
  "configuration": {
    "BranchName": "production",
    "PollForSourceChanges": "false",
    "RepositoryName": "my-repo"
  },
  "resolvedConfiguration": {
    "BranchName": "production",
    "PollForSourceChanges": "false",
    "RepositoryName": "my-repo"
  },
  "region": "us-east-1",
  "inputArtifacts": [],
  "namespace": "SourceVariables"
},
"output": {
  "outputArtifacts": [
    {
      "name": "SourceArtifact",
      "s3location": {
        "bucket": "my-bucket",
        "key": "myPipeline/SourceArti/key"
      }
    }
  ],
  "executionResult": {
    "externalExecutionId":
"11111111ad99dcd35914c00b7fbea13995EXAMPLE",
    "externalExecutionSummary": "Edited template.yml",
    "externalExecutionUrl": "https://myaddress.com"
  },
  "outputVariables": {
    "AuthorDate": "2020-05-08T17:45:43Z",
    "BranchName": "production",
    "CommitId": "EXAMPLEad99dcd35914c00b7fbea139951111111",
    "CommitMessage": "Edited template.yml",
    "CommitterDate": "2020-05-08T17:45:43Z",
```

```

        "RepositoryName": "my-repo"
      }
    },
    . . . .

```

For more information, see [View action executions \(CLI\)](#) in the *AWS CodePipeline User Guide*.

- For API details, see [ListActionExecutions](#) in *AWS CLI Command Reference*.

list-action-types

The following code example shows how to use `list-action-types`.

AWS CLI

To view the action types available

Used by itself, the `list-action-types` command returns the structure of all actions available to your AWS account. This example uses the `--action-owner-filter` option to return only custom actions.

Command:

```
aws codepipeline list-action-types --action-owner-filter Custom
```

Output:

```

{
  "actionTypes": [
    {
      "inputArtifactDetails": {
        "maximumCount": 5,
        "minimumCount": 0
      },
      "actionConfigurationProperties": [
        {
          "secret": false,
          "required": true,
          "name": "MyJenkinsExampleBuildProject",
          "key": true,
          "queryable": true
        }
      ]
    }
  ]
}

```

```
    ],
    "outputArtifactDetails": {
      "maximumCount": 5,
      "minimumCount": 0
    },
    "id": {
      "category": "Build",
      "owner": "Custom",
      "version": "1",
      "provider": "MyJenkinsProviderName"
    },
    "settings": {
      "entityUrlTemplate": "http://192.0.2.4/job/{Config:ProjectName}",
      "executionUrlTemplate": "http://192.0.2.4/job/{Config:ProjectName}/
{ExternalExecutionId}"
    }
  },
  {
    "inputArtifactDetails": {
      "maximumCount": 5,
      "minimumCount": 0
    },
    "actionConfigurationProperties": [
      {
        "secret": false,
        "required": true,
        "name": "MyJenkinsExampleTestProject",
        "key": true,
        "queryable": true
      }
    ],
    "outputArtifactDetails": {
      "maximumCount": 5,
      "minimumCount": 0
    },
    "id": {
      "category": "Test",
      "owner": "Custom",
      "version": "1",
      "provider": "MyJenkinsProviderName"
    },
    "settings": {
      "entityUrlTemplate": "http://192.0.2.4/job/{Config:ProjectName}",
```

```

        "executionUrlTemplate": "http://192.0.2.4/job/{Config:ProjectName}/
{ExternalExecutionId}"
    }
}
]
}

```

- For API details, see [ListActionTypes](#) in *AWS CLI Command Reference*.

list-pipeline-executions

The following code example shows how to use `list-pipeline-executions`.

AWS CLI

To view pipeline execution history

The following `list-pipeline-executions` example shows the pipeline execution history for a pipeline in your AWS account.

```

aws codepipeline list-pipeline-executions \
  --pipeline-name MyPipeline

```

Output:

```

{
  "pipelineExecutionSummaries": [
    {
      "lastUpdateTime": 1496380678.648,
      "pipelineExecutionId": "7cf7f7cb-3137-539g-j458-d7eu3EXAMPLE",
      "startTime": 1496380258.243,
      "status": "Succeeded"
    },
    {
      "lastUpdateTime": 1496591045.634,
      "pipelineExecutionId": "3137f7cb-8d494hj4-039j-d84l-d7eu3EXAMPLE",
      "startTime": 1496590401.222,
      "status": "Succeeded"
    },
    {
      "lastUpdateTime": 1496946071.6456,
      "pipelineExecutionId": "4992f7jf-7cf7-913k-k334-d7eu3EXAMPLE",

```

```
        "startTime": 1496945471.5645,  
        "status": "Succeeded"  
      }  
    ]  
  }  
}
```

For more information, see [View execution history](#) in the *AWS CodePipeline User Guide*.

- For API details, see [ListPipelineExecutions](#) in *AWS CLI Command Reference*.

list-pipelines

The following code example shows how to use `list-pipelines`.

AWS CLI

To view a list of pipelines

This example lists all AWS CodePipeline pipelines associated with the user's AWS account.

Command:

```
aws codepipeline list-pipelines
```

Output:

```
{  
  "pipelines": [  
    {  
      "updated": 1439504274.641,  
      "version": 1,  
      "name": "MyFirstPipeline",  
      "created": 1439504274.641  
    },  
    {  
      "updated": 1436461837.992,  
      "version": 2,  
      "name": "MySecondPipeline",  
      "created": 1436460801.381  
    }  
  ]  
}
```


- For API details, see [ListPipelines](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags

The following `list-tags-for-resource` example retrieves a list of all tags attached to the specified pipeline resource.

```
aws codepipeline list-tags-for-resource \
  --resource-arn arn:aws:codepipeline:us-east-1:123456789012:MyPipeline
```

Output:

```
{
  "tags": {
    "Project": "ProjectA",
    "IscontainerBased": "true"
  }
}
```

For more information, see [View tags for a pipeline \(CLI\)](#) in the *AWS CodePipeline User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

list-webhooks

The following code example shows how to use `list-webhooks`.

AWS CLI

To list webhooks

The following `list-webhooks` example retrieves a list of all tags attached to the specified pipeline resource.

```
aws codepipeline list-webhooks \
  --endpoint-url "https://codepipeline.eu-central-1.amazonaws.com" \
```

```
--region "eu-central-1"
```

Output:

```
{
  "webhooks": [
    {
      "url": "https://webhooks.domain.com/
trigger1111111111EXAMPLE1111111111111111111": {
      "authenticationConfiguration": {
        "SecretToken": "Secret"
      },
      "name": "my-webhook",
      "authentication": "GITHUB_HMAC",
      "targetPipeline": "my-Pipeline",
      "targetAction": "Source",
      "filters": [
        {
          "jsonPath": "$.ref",
          "matchEquals": "refs/heads/{Branch}"
        }
      ]
    },
    "arn": "arn:aws:codepipeline:eu-central-1:123456789012:webhook:my-
webhook"
  ]
}
```

For more information, see [List webhooks in your account](#) in the *AWS CodePipeline User Guide*.

- For API details, see [ListWebhooks](#) in *AWS CLI Command Reference*.

poll-for-jobs

The following code example shows how to use `poll-for-jobs`.

AWS CLI

To view any available jobs

This example returns information about any jobs for a job worker to act upon. This example uses a pre-defined JSON file (`MyActionTypesInfo.json`) to supply information about the action

type for which the job worker processes jobs. This command is only used for custom actions. When this command is called, AWS CodePipeline returns temporary credentials for the Amazon S3 bucket used to store artifacts for the pipeline. This command will also return any secret values defined for the action, if any are defined.

Command:

```
aws codepipeline poll-for-jobs --cli-input-json file://MyActionTypeInfo.json
```

JSON file sample contents:

```
{
  "actionTypeId": {
    "category": "Test",
    "owner": "Custom",
    "provider": "MyJenkinsProviderName",
    "version": "1"
  },
  "maxBatchSize": 5,
  "queryParam": {
    "ProjectName": "MyJenkinsTestProject"
  }
}
```

Output:

```
{
  "jobs": [
    {
      "accountId": "111111111111",
      "data": {
        "actionConfiguration": {
          "__type": "ActionConfiguration",
          "configuration": {
            "ProjectName": "MyJenkinsExampleTestProject"
          }
        },
        "actionTypeId": {
          "__type": "ActionTypeId",
          "category": "Test",
          "owner": "Custom",
          "provider": "MyJenkinsProviderName",
```

```
    "version": "1"
  },
  "artifactCredentials": {
    "__type": "AWSSessionCredentials",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "secretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "sessionToken":
    "fICcQD6m7oRw0uX0jANBqkqhkiG9w0BAQUFADCBiDELMaKGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwd
+a4GmWIWJ21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEI03IyNoH/
f0wYK8m9TrDHudUZg3qX4waLG5M43q7Wgc/
MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpEIbb30hjZnzcVQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQ
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0FkbFFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs
  },
  "inputArtifacts": [
    {
      "__type": "Artifact",
      "location": {
        "s3Location": {
          "bucketName": "codepipeline-us-east-1-11EXAMPLE11",
          "objectKey": "MySecondPipeline/MyAppBuild/EXAMPLE"
        },
        "type": "S3"
      },
      "name": "MyAppBuild"
    }
  ],
  "outputArtifacts": [],
  "pipelineContext": {
    "__type": "PipelineContext",
    "action": {
      "name": "MyJenkinsTest-Action"
    },
    "pipelineName": "MySecondPipeline",
    "stage": {
      "name": "Testing"
    }
  },
  "id": "ef66c259-64f9-EXAMPLE",
  "nonce": "3"
}
]
```

- For API details, see [PollForJobs](#) in *AWS CLI Command Reference*.

put-webhook

The following code example shows how to use `put-webhook`.

AWS CLI

To create a webhook

The following `put-webhook` example creates a webhook for a GitHub version 1 source action. After you create the webhook, you must use the `register-webhook-with-third-party` command to register it.

```
aws codepipeline put-webhook \  
  --cli-input-json file://webhook_json.json \  
  --region "eu-central-1"
```

Contents of `webhook_json.json`:

```
{  
  "webhook": {  
    "name": "my-webhook",  
    "targetPipeline": "pipeline_name",  
    "targetAction": "source_action_name",  
    "filters": [  
      {  
        "jsonPath": "$.ref",  
        "matchEquals": "refs/heads/{Branch}"  
      }  
    ],  
    "authentication": "GITHUB_HMAC",  
    "authenticationConfiguration": {  
      "SecretToken": "secret"  
    }  
  }  
}
```

Output:

```
{  
  "webhook": {
```

```

    "url": "https://webhooks.domain.com/
trigger1111111111EXAMPLE111111111111111111",
    "definition": {
      "authenticationConfiguration": {
        "SecretToken": "secret"
      },
      "name": "my-webhook",
      "authentication": "GITHUB_HMAC",
      "targetPipeline": "pipeline_name",
      "targetAction": "Source",
      "filters": [
        {
          "jsonPath": "$.ref",
          "matchEquals": "refs/heads/{Branch}"
        }
      ]
    },
    "arn": "arn:aws:codepipeline:eu-central-1:123456789012:webhook:my-webhook"
  },
  "tags": [
    {
      "key": "Project",
      "value": "ProjectA"
    }
  ]
}

```

For more information, see [Create a webhook for a GitHub source](#) in the *AWS CodePipeline User Guide*.

- For API details, see [PutWebhook](#) in *AWS CLI Command Reference*.

retry-stage-execution

The following code example shows how to use `retry-stage-execution`.

AWS CLI

To retry a failed action

The following `retry-stage-execution` example retries a stage that has a failed action.

```
aws codepipeline retry-stage-execution \
```

```
--pipeline-name MyPipeline \  
--stage-name Deploy \  
--pipeline-execution-id b59babff-5f34-EXAMPLE \  
--retry-mode FAILED_ACTIONS
```

Output:

```
{  
  "pipelineExecutionId": "b59babff-5f34-EXAMPLE"  
}
```

For more information, see [Retry failed actions \(CLI\)](#) in the *AWS CodePipeline User Guide*.

- For API details, see [RetryStageExecution](#) in *AWS CLI Command Reference*.

start-pipeline-execution

The following code example shows how to use `start-pipeline-execution`.

AWS CLI

To run the latest revision through a pipeline

This example runs the latest revision present in the source stage of a pipeline through the pipeline named "MyFirstPipeline".

Command:

```
aws codepipeline start-pipeline-execution --name MyFirstPipeline
```

Output:

```
{  
  "pipelineExecutionId": "3137f7cb-7cf7-EXAMPLE"  
}
```

- For API details, see [StartPipelineExecution](#) in *AWS CLI Command Reference*.

stop-pipeline-execution

The following code example shows how to use `stop-pipeline-execution`.

AWS CLI

To stop a pipeline execution

The following `stop-pipeline-execution` example defaults to waiting until in-progress actions finish, and then stops the pipeline. You cannot choose to stop and wait if the execution is already in a Stopping state. You can choose to stop and abandon an execution that is already in a Stopping state.

```
aws codepipeline stop-pipeline-execution \  
  --pipeline-name MyFirstPipeline \  
  --pipeline-execution-id d-EXAMPLE \  
  --reason "Stopping pipeline after the build action is done"
```

This command returns no output.

For more information, see [Stop a pipeline execution \(CLI\)](#) in the *AWS CodePipeline User Guide*.

- For API details, see [StopPipelineExecution](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To tag a resource

The following `tag-resource` example associates a set of provided tags with a pipeline. Use this command to add or edit tags.

```
aws codepipeline tag-resource \  
  --resource-arn arn:aws:codepipeline:us-east-1:123456789012:MyPipeline \  
  --tags key=Project,value=ProjectA key=IscontainerBased,value=true
```

This command produces no output.

For more information, see [Add tags to a pipeline \(CLI\)](#) in the *AWS CodePipeline User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove AWS tags from a connections resource

The following `untag-resource` example removes a tag from the specified resource.

```
aws codepipeline untag-resource \  
  --resource-arn arn:aws:codepipeline:us-east-1:123456789012:MyPipeline \  
  --tag-keys Project IscontainerBased
```

This command produces no output.

For more information, see [Remove tags from a pipeline \(CLI\)](#) in the *AWS CodePipeline User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-pipeline

The following code example shows how to use `update-pipeline`.

AWS CLI

To update the structure of a pipeline

This example uses the `update-pipeline` command with the `--cli-input-json` argument. This example uses a pre-defined JSON file (`MyFirstPipeline.json`) to update the structure of a pipeline. AWS CodePipeline recognizes the pipeline name contained in the JSON file, and then applies any changes from modified fields in the pipeline structure to update the pipeline.

Use the following guidelines when creating the pre-defined JSON file:

If you are working with a pipeline structure retrieved using the `get-pipeline` command, you must remove the metadata section from the pipeline structure in the JSON file (the `"metadata": { }` lines and the `"created," "pipelineARN,"` and `"updated"` fields within). The pipeline name cannot be changed.

Command:

```
aws codepipeline update-pipeline --cli-input-json file://MyFirstPipeline.json
```

Sample JSON file contents:

```
{
  "pipeline": {
    "roleArn": "arn:aws:iam::111111111111:role/AWS-CodePipeline-Service",
    "stages": [
      {
        "name": "Source",
        "actions": [
          {
            "inputArtifacts": [],
            "name": "Source",
            "actionTypeId": {
              "category": "Source",
              "owner": "AWS",
              "version": "1",
              "provider": "S3"
            },
            "outputArtifacts": [
              {
                "name": "MyApp"
              }
            ],
            "configuration": {
              "S3Bucket": "awscodepipeline-demo-bucket2",
              "S3ObjectKey": "aws-codepipeline-s3-aws-codedeploy_linux.zip"
            },
            "runOrder": 1
          }
        ]
      },
      {
        "name": "Beta",
        "actions": [
          {
            "inputArtifacts": [
              {
                "name": "MyApp"
              }
            ],
            "name": "CodePipelineDemoFleet",
```

```

        "actionTypeId": {
            "category": "Deploy",
            "owner": "AWS",
            "version": "1",
            "provider": "CodeDeploy"
        },
        "outputArtifacts": [],
        "configuration": {
            "ApplicationName": "CodePipelineDemoApplication",
            "DeploymentGroupName": "CodePipelineDemoFleet"
        },
        "runOrder": 1
    }
]
}
],
"artifactStore": {
    "type": "S3",
    "location": "codepipeline-us-east-1-11EXAMPLE11"
},
"name": "MyFirstPipeline",
"version": 1
}
}

```

Output:

```

{
  "pipeline": {
    "artifactStore": {
      "location": "codepipeline-us-east-1-11EXAMPLE11",
      "type": "S3"
    },
    "name": "MyFirstPipeline",
    "roleArn": "arn:aws:iam::111111111111:role/AWS-CodePipeline-Service",
    "stages": [
      {
        "actions": [
          {
            "actionTypeId": {
              "__type": "ActionTypeId",
              "category": "Source",
              "owner": "AWS",

```

```
        "provider": "S3",
        "version": "1"
    },
    "configuration": {
        "S3Bucket": "awscodepipeline-demo-bucket2",
        "S3ObjectKey": "aws-codepipeline-s3-aws-codedeploy_linux.zip"
    },
    "inputArtifacts": [],
    "name": "Source",
    "outputArtifacts": [
        {
            "name": "MyApp"
        }
    ],
    "runOrder": 1
}
],
"name": "Source"
},
{
    "actions": [
        {
            "actionTypeId": {
                "__type": "ActionTypeId",
                "category": "Deploy",
                "owner": "AWS",
                "provider": "CodeDeploy",
                "version": "1"
            },
            "configuration": {
                "ApplicationName": "CodePipelineDemoApplication",
                "DeploymentGroupName": "CodePipelineDemoFleet"
            },
            "inputArtifacts": [
                {
                    "name": "MyApp"
                }
            ],
            "name": "CodePipelineDemoFleet",
            "outputArtifacts": [],
            "runOrder": 1
        }
    ],
    "name": "Beta"
}
```

```
    }  
  ],  
  "version": 3  
}  
}
```

- For API details, see [UpdatePipeline](#) in *AWS CLI Command Reference*.

AWS CodeStar examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS CodeStar.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-team-member

The following code example shows how to use `associate-team-member`.

AWS CLI

To add a team member to a project

The following `associate-team-member` example makes the `intern` user a viewer on the project with the specified ID.

```
aws codestar associate-team-member \
```

```
--project-id my-project \  
--user-arn arn:aws:iam::123456789012:user/intern \  
--project-role Viewer
```

This command produces no output.

- For API details, see [AssociateTeamMember](#) in *AWS CLI Command Reference*.

create-project

The following code example shows how to use create-project.

AWS CLI

To create a project

The following create-project example uses a JSON input file to create a CodeStar project.

```
aws codestar create-project \  
  --cli-input-json file://create-project.json
```

Contents of create-project.json:

```
{  
  "name": "Custom Project",  
  "id": "custom-project",  
  "sourceCode": [  
    {  
      "source": {  
        "s3": {  
          "bucketName": "codestar-artifacts",  
          "bucketKey": "nodejs-function.zip"  
        }  
      },  
      "destination": {  
        "codeCommit": {  
          "name": "codestar-custom-project"  
        }  
      }  
    }  
  ],  
  "toolchain": {
```

```
    "source": {
      "s3": {
        "bucketName": "codestar-artifacts",
        "bucketKey": "toolchain.yml"
      }
    },
    "roleArn": "arn:aws:iam::123456789012:role/service-role/aws-codestar-
service-role",
    "stackParameters": {
      "ProjectId": "custom-project"
    }
  }
}
```

Output:

```
{
  "id": "my-project",
  "arn": "arn:aws:codestar:us-east-2:123456789012:project/custom-project"
}
```

For a tutorial that includes sample code and templates for a custom project, see [Create a Project in AWS CodeStar with the AWS CLI](https://docs.aws.amazon.com/codestar/latest/userguide/cli-tutorial.html) in the *AWS CodeStar User Guide*.

- For API details, see [CreateProject](#) in *AWS CLI Command Reference*.

create-user-profile

The following code example shows how to use `create-user-profile`.

AWS CLI

To create a user profile

The following `create-user-profile` example creates a user profile for the IAM user with the specified ARN.

```
aws codestar create-user-profile \
  --user-arn arn:aws:iam::123456789012:user/intern \
  --display-name Intern \
```

```
--email-address intern@example.com
```

Output:

```
{
  "userArn": "arn:aws:iam::123456789012:user/intern",
  "displayName": "Intern",
  "emailAddress": "intern@example.com",
  "sshPublicKey": "",
  "createdTimestamp": 1572552308.607,
  "lastModifiedTimestamp": 1572552308.607
}
```

- For API details, see [CreateUserProfile](#) in *AWS CLI Command Reference*.

delete-project

The following code example shows how to use delete-project.

AWS CLI

To delete a project

The following delete-project example deletes the specified project.

```
aws codestar delete-project \
  --project-id my-project
```

Output:

```
{
  "projectArn": "arn:aws:codestar:us-east-2:123456789012:project/my-project"
}
```

- For API details, see [DeleteProject](#) in *AWS CLI Command Reference*.

delete-user-profile

The following code example shows how to use delete-user-profile.

AWS CLI

To delete a user profile

The following `delete-user-profile` example deletes the user profile for the user with the specified ARN.

```
aws codestar delete-user-profile \  
  --user-arn arn:aws:iam::123456789012:user/intern
```

Output:

```
{  
  "userArn": "arn:aws:iam::123456789012:user/intern"  
}
```

- For API details, see [DeleteUserProfile](#) in *AWS CLI Command Reference*.

describe-project

The following code example shows how to use `describe-project`.

AWS CLI

To view a project

The following `describe-project` example retrieves details about the specified project.

```
aws codestar describe-project \  
  --id my-project
```

Output:

```
{  
  "name": "my project",  
  "id": "my-project",  
  "arn": "arn:aws:codestar:us-west-2:123456789012:project/my-project",  
  "description": "My first CodeStar project.",  
  "createdTimeStamp": 1572547510.128,  
  "status": {
```

```
    "state": "CreateComplete"
  }
}
```

- For API details, see [DescribeProject](#) in *AWS CLI Command Reference*.

describe-user-profile

The following code example shows how to use `describe-user-profile`.

AWS CLI

To view a user profile

The following `describe-user-profile` example retrieves details about the user profile for the user with the specified ARN.

```
aws codestar describe-user-profile \
  --user-arn arn:aws:iam::123456789012:user/intern
```

Output:

```
{
  "userArn": "arn:aws:iam::123456789012:user/intern",
  "displayName": "Intern",
  "emailAddress": "intern@example.com",
  "sshPublicKey": "intern",
  "createdTimestamp": 1572552308.607,
  "lastModifiedTimestamp": 1572553495.47
}
```

- For API details, see [DescribeUserProfile](#) in *AWS CLI Command Reference*.

disassociate-team-member

The following code example shows how to use `disassociate-team-member`.

AWS CLI

To remove a team member

The following `disassociate-team-member` example removes the user with the specified ARN from the project `my-project`.

```
aws codestar disassociate-team-member \  
  --project-id my-project \  
  --user-arn arn:aws:iam::123456789012:user/intern
```

This command produces no output.

- For API details, see [DisassociateTeamMember](#) in *AWS CLI Command Reference*.

list-projects

The following code example shows how to use `list-projects`.

AWS CLI

To view projects

The following `list-projects` example retrieves a list of projects in the current Region.

```
aws codestar list-projects
```

Output:

```
{  
  "projects": [  
    {  
      "projectId": "intern-projects",  
      "projectArn": "arn:aws:codestar:us-west-2:123456789012:project/intern-  
projects"  
    },  
    {  
      "projectId": "my-project",  
      "projectArn": "arn:aws:codestar:us-west-2:123456789012:project/my-  
project"  
    }  
  ]  
}
```

- For API details, see [ListProjects](#) in *AWS CLI Command Reference*.

list-resources

The following code example shows how to use `list-resources`.

AWS CLI

To view resources

The following `list-resources` example retrieves a list of resources for the specified project.

```
aws codestar list-resources \  
  --id my-project
```

Output:

```
{  
  "resources": [  
    {  
      "id": "arn:aws:execute-api:us-east-2:123456789012:r3wxmplbv8"  
    },  
    {  
      "id": "arn:aws:codedeploy:us-east-2:123456789012:application:awscodestar-my-project-lambda-ServerlessDeploymentApplication-PF0LXMPL1KA0"  
    },  
    {  
      "id": "arn:aws:s3:::aws-codestar-us-east-2-123456789012-my-project-pipe"  
    },  
    {  
      "id": "arn:aws:lambda:us-east-2:123456789012:function:awscodestar-my-project-lambda-GetHelloWorld-16W3LVXMPLNNS"  
    },  
    {  
      "id": "arn:aws:cloudformation:us-east-2:123456789012:stack/awscodestar-my-project-lambda/b4904ea0-fc20-xmpl-bec6-029123b1cc42"  
    },  
    {  
      "id": "arn:aws:cloudformation:us-east-2:123456789012:stack/awscodestar-my-project/1b133f30-fc20-xmpl-a93a-0688c4290cb8"  
    },  
    {  
      "id": "arn:aws:iam::123456789012:role/CodeStarWorker-my-project-ToolChain"  
    }  
  ]  
}
```

```

    {
      "id": "arn:aws:iam::123456789012:policy/CodeStar_my-
project_PermissionsBoundary"
    },
    {
      "id": "arn:aws:s3:::aws-codestar-us-east-2-123456789012-my-project-app"
    },
    {
      "id": "arn:aws:codepipeline:us-east-2:123456789012:my-project-Pipeline"
    },
    {
      "id": "arn:aws:codedeploy:us-east-2:123456789012:deploymentgroup:my-
project/awscodestar-my-project-lambda-GetHelloWorldDeploymentGroup-P7YWXMPLT0QB"
    },
    {
      "id": "arn:aws:iam::123456789012:role/CodeStar-my-project-Execution"
    },
    {
      "id": "arn:aws:iam::123456789012:role/CodeStarWorker-my-project-
CodeDeploy"
    },
    {
      "id": "arn:aws:codebuild:us-east-2:123456789012:project/my-project"
    },
    {
      "id": "arn:aws:iam::123456789012:role/CodeStarWorker-my-project-
CloudFormation"
    },
    {
      "id": "arn:aws:codecommit:us-east-2:123456789012:Go-project"
    }
  ]
}

```

- For API details, see [ListResources](#) in *AWS CLI Command Reference*.

list-tags-for-project

The following code example shows how to use `list-tags-for-project`.

AWS CLI

To view tags for a project

The following `list-tags-for-project` example retrieves the tags attached to the specified project.

```
aws codestar list-tags-for-project \  
  --id my-project
```

Output:

```
{  
  "tags": {  
    "Department": "Marketing",  
    "Team": "Website"  
  }  
}
```

- For API details, see [ListTagsForProject](#) in *AWS CLI Command Reference*.

list-team-members

The following code example shows how to use `list-team-members`.

AWS CLI

To view a list of team members

The following `list-team-members` example retrieves a list of users associated with the specified project.

```
aws codestar list-team-members \  
  --project-id my-project
```

Output:

```
{  
  "teamMembers": [  
    {  
      "userArn": "arn:aws:iam::123456789012:user/admin",  
      "projectRole": "Owner",  
      "remoteAccessAllowed": false  
    },  
    {
```

```
    "userArn": "arn:aws:iam::123456789012:user/intern",
    "projectRole": "Contributor",
    "remoteAccessAllowed": false
  }
]
```

- For API details, see [ListTeamMembers](#) in *AWS CLI Command Reference*.

list-user-profiles

The following code example shows how to use `list-user-profiles`.

AWS CLI

To view a list of user profiles

The following `list-user-profiles` example retrieves a list of all user profiles in the current Region.

```
aws codestar list-user-profiles
```

Output:

```
{
  "userProfiles": [
    {
      "userArn": "arn:aws:iam::123456789012:user/admin",
      "displayName": "me",
      "emailAddress": "me@example.com",
      "sshPublicKey": ""
    },
    {
      "userArn": "arn:aws:iam::123456789012:user/intern",
      "displayName": "Intern",
      "emailAddress": "intern@example.com",
      "sshPublicKey": "intern"
    }
  ]
}
```

- For API details, see [ListUserProfiles](#) in *AWS CLI Command Reference*.

tag-project

The following code example shows how to use tag-project.

AWS CLI

To attach a tag to a project

The following tag-project example adds a tag named Department and a value of Marketing to the specified project.

```
aws codestar tag-project \  
  --id my-project \  
  --tags Department=Marketing
```

Output:

```
{  
  "tags": {  
    "Department": "Marketing"  
  }  
}
```

- For API details, see [TagProject](#) in *AWS CLI Command Reference*.

untag-project

The following code example shows how to use untag-project.

AWS CLI

To remove a tag from a project

The following untag-project example removes any tag with a key name of Team from the specific project.

```
aws codestar untag-project \  
  --id my-project \  
  --tags Team
```

This command produces no output.

- For API details, see [UntagProject](#) in *AWS CLI Command Reference*.

update-project

The following code example shows how to use `update-project`.

AWS CLI

To update a project

The following `update-project` example adds a description to the specified project.

```
aws codestar update-project \  
  --id my-project \  
  --description "My first CodeStar project"
```

This command produces no output.

- For API details, see [UpdateProject](#) in *AWS CLI Command Reference*.

update-team-member

The following code example shows how to use `update-team-member`.

AWS CLI

To modify a team member

The following `update-team-member` example makes the specified user a contributor on a project and grants them remote access to project resources.

```
aws codestar update-team-member \  
  --project-id my-project \  
  --user-arn arn:aws:iam::123456789012:user/intern \  
  --project-role Contributor -\  
  --remote-access-allowed
```

Output:

```
{  
  "userArn": "arn:aws:iam::123456789012:user/intern",  
  "projectRole": "Contributor",
```

```
"remoteAccessAllowed": true
}
```

- For API details, see [UpdateTeamMember](#) in *AWS CLI Command Reference*.

update-user-profile

The following code example shows how to use `update-user-profile`.

AWS CLI

To modify a user profile

The following `update-user-profile` example adds the specified SSH key to the specified user.

```
aws codestar update-user-profile \
  --ssh-public-key intern \
  --user-arn arn:aws:iam::123456789012:user/intern
```

Output:

```
{
  "userArn": "arn:aws:iam::123456789012:user/intern",
  "displayName": "Intern",
  "emailAddress": "intern@example.com",
  "sshPublicKey": "intern",
  "createdTimestamp": 1572552308.607,
  "lastModifiedTimestamp": 1572553495.47
}
```

- For API details, see [UpdateUserProfile](#) in *AWS CLI Command Reference*.

AWS CodeStar Notifications examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS CodeStar Notifications.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-notification-rule

The following code example shows how to use `create-notification-rule`.

AWS CLI

To create a notification rule

The following `create-notification-rule` example uses a JSON file named `rule.json` to create a notification rule named `MyNotificationRule` for a repository named `MyDemoRepo` in the specified AWS account. Notifications with the `FULL` detail type are sent to the specified target Amazon SNS topic when branches and tags are created.

```
aws codestar-notifications create-notification-rule \  
  --cli-input-json file://rule.json
```

Contents of `rule.json`:

```
{  
  "Name": "MyNotificationRule",  
  "EventTypeIds": [  
    "codecommit-repository-branches-and-tags-created"  
  ],  
  "Resource": "arn:aws:codecommit:us-east-1:123456789012:MyDemoRepo",  
  "Targets": [  
    {  
      "TargetType": "SNS",  
      "TargetAddress": "arn:aws:sns:us-  
east-1:123456789012:MyNotificationTopic"  
    }  
  ]  
}
```

```
    ],  
    "Status": "ENABLED",  
    "DetailType": "FULL"  
  }  
}
```

Output:

```
{  
  "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/  
dc82df7a-EXAMPLE"  
}
```

For more information, see [Create a Notification rule](#) in the *AWS Developer Tools Console User Guide*.

- For API details, see [CreateNotificationRule](#) in *AWS CLI Command Reference*.

delete-notification-rule

The following code example shows how to use `delete-notification-rule`.

AWS CLI

To delete a notification rule

The following `delete-notification-rule` example deletes the specified notification rule.

```
aws codestar-notifications delete-notification-rule \  
  --arn arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/  
dc82df7a-EXAMPLE
```

Output:

```
{  
  "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/  
dc82df7a-EXAMPLE"  
}
```

For more information, see [Delete a Notification Rule](#) in the *AWS Developer Tools Console User Guide*.

- For API details, see [DeleteNotificationRule](#) in *AWS CLI Command Reference*.

delete-target

The following code example shows how to use `delete-target`.

AWS CLI

To delete a notification rule target

The following `delete-target` example removes the specified target from all notification rules configured to use it as a target, and then deletes the target.

```
aws codestar-notifications delete-target \  
  --target-address arn:aws:sns:us-east-1:123456789012:MyNotificationTopic \  
  --force-unsubscribe-all
```

This command produces no output.

For more information, see [Delete a Notification Rule Target](#) in the *AWS Developer Tools Console User Guide*.

- For API details, see [DeleteTarget](#) in *AWS CLI Command Reference*.

describe-notification-rule

The following code example shows how to use `describe-notification-rule`.

AWS CLI

To retrieve details of a notification rule

The following `describe-notification-rule` example retrieves the details of the specified notification rule.

```
aws codestar-notifications describe-notification-rule \  
  --arn arn:aws:codestar-notifications:us-west-2:123456789012:notificationrule/  
  dc82df7a-EXAMPLE
```

Output:

```
{  
  "LastModifiedTimestamp": 1569199844.857,  
  "EventTypes": [  
    ...  
  ]  
}
```

```

    {
      "ServiceName": "CodeCommit",
      "EventTypeName": "Branches and tags: Created",
      "ResourceType": "Repository",
      "EventTypeId": "codecommit-repository-branches-and-tags-created"
    }
  ],
  "Status": "ENABLED",
  "DetailType": "FULL",
  "Resource": "arn:aws:codecommit:us-west-2:123456789012:MyDemoRepo",
  "Arn": "arn:aws:codestar-notifications:us-west-w:123456789012:notificationrule/
dc82df7a-EXAMPLE",
  "Targets": [
    {
      "TargetStatus": "ACTIVE",
      "TargetAddress": "arn:aws:sns:us-
west-2:123456789012:MyNotificationTopic",
      "TargetType": "SNS"
    }
  ],
  "Name": "MyNotificationRule",
  "CreatedTimestamp": 1569199844.857,
  "CreatedBy": "arn:aws:iam::123456789012:user/Mary_Major"
}

```

For more information, see [View Notification Rules](#) in the *AWS Developer Tools Console User Guide*.

- For API details, see [DescribeNotificationRule](#) in *AWS CLI Command Reference*.

list-event-types

The following code example shows how to use `list-event-types`.

AWS CLI

To get a list of event types for a notification rule

The following `list-event-types` example retrieves a filtered list of all available notification event types for CodeDeploy applications. If instead you use no filter, the command returns all notification event types for all resource types.

```
aws codestar-notifications list-event-types \
```

```
--filters Name=SERVICE_NAME,Value=CodeDeploy
```

Output:

```
{
  "EventTypes": [
    {
      "EventTypeId": "codedeploy-application-deployment-succeeded",
      "ServiceName": "CodeDeploy",
      "EventTypeName": "Deployment: Succeeded",
      "ResourceType": "Application"
    },
    {
      "EventTypeId": "codedeploy-application-deployment-failed",
      "ServiceName": "CodeDeploy",
      "EventTypeName": "Deployment: Failed",
      "ResourceType": "Application"
    },
    {
      "EventTypeId": "codedeploy-application-deployment-started",
      "ServiceName": "CodeDeploy",
      "EventTypeName": "Deployment: Started",
      "ResourceType": "Application"
    }
  ]
}
```

For more information, see [Create a Notification Rule](#) in the *AWS Developer Tools Console User Guide*.

- For API details, see [ListEventTypes](#) in *AWS CLI Command Reference*.

list-notification-rules

The following code example shows how to use `list-notification-rules`.

AWS CLI

To retrieve a list of notification rules

The following `list-notification-rules` example retrieves a list of all notification rules in the specified AWS Region.

```
aws codestar-notifications list-notification-rules --region us-east-1
```

Output:

```
{
  "NotificationRules": [
    {
      "Id": "dc82df7a-EXAMPLE",
      "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/dc82df7a-EXAMPLE"
    },
    {
      "Id": "8d1f0983-EXAMPLE",
      "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/8d1f0983-EXAMPLE"
    }
  ]
}
```

For more information, see [View Notification Rules](#) in the *AWS Developer Tools Console User Guide*.

- For API details, see [ListNotificationRules](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To get a list of tags attached to a notification rule

The following `list-tags-for-resource` example retrieves a list of all tags attached to the specified notification rule. In this example, the notification rule currently has no tags associated with it.

```
aws codestar-notifications list-tags-for-resource \
  --arn arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/
  fe1efd35-EXAMPLE
```

Output:


```
{
  "Tags": {}
}
```

For more information, see [Create a Notification Rule](#) in the *AWS Developer Tools Console User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

list-targets

The following code example shows how to use `list-targets`.

AWS CLI

To retrieve a list of notification rule targets

The following `list-targets` example retrieves a list of all notification rule targets in the specified AWS Region.

```
aws codestar-notifications list-targets \
  --region us-east-1
```

Output:

```
{
  "Targets": [
    {
      "TargetAddress": "arn:aws:sns:us-
east-1:123456789012:MySNSTopicForNotificationRules",
      "TargetType": "SNS",
      "TargetStatus": "ACTIVE"
    },
    {
      "TargetAddress": "arn:aws:sns:us-
east-1:123456789012:MySNSTopicForNotificationsAboutMyDemoRepo",
      "TargetType": "SNS",
      "TargetStatus": "ACTIVE"
    }
  ]
}
```

For more information, see [View Notification Rule Targets](#) in the *AWS Developer Tools Console User Guide*.

- For API details, see [ListTargets](#) in *AWS CLI Command Reference*.

subscribe

The following code example shows how to use `subscribe`.

AWS CLI

To add a target to a notification rule

The following `subscribe` example adds an Amazon SNS topic as a target for the specified notification rule.

```
aws codestar-notifications subscribe \  
  --arn arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/  
dc82df7a-EXAMPLE \  
  --target TargetType=SNS,TargetAddress=arn:aws:sns:us-  
east-1:123456789012:MyNotificationTopic
```

Output:

```
{  
  "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/  
dc82df7a-EXAMPLE"  
}
```

For more information, see [Add or Remove an Amazon SNS Topic as a Target for a Notification Rule](#) in the *AWS Developer Tools Console User Guide*.

- For API details, see [Subscribe](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add a tag to a notification rule

The following tag-resource example adds a tag with the key name of Team and the value of Li_Juan to the specified notification rule.

```
aws codestar-notifications tag-resource \  
  --arn arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/  
fe1efd35-EXAMPLE \  
  --tags Team=Li_Juan
```

Output:

```
{  
  "Tags": {  
    "Team": "Li_Juan"  
  }  
}
```

For more information, see [Create a Notification Rule](#) in the *AWS Developer Tools Console User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

unsubscribe

The following code example shows how to use unsubscribe.

AWS CLI

To remove a target from a notification rule

The following unsubscribe example removes an Amazon SNS topic as a target from the specified notification rule.

```
aws codestar-notifications unsubscribe \  
  --arn arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/  
dc82df7a-EXAMPLE \  
  --target TargetType=SNS,TargetAddress=arn:aws:sns:us-  
east-1:123456789012:MyNotificationTopic
```

Output:

```
{
```

```
"Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/
dc82df7a-EXAMPLE"
  "TargetAddress": "arn:aws:sns:us-east-1:123456789012:MyNotificationTopic"
}
```

For more information, see [Add or Remove an Amazon SNS Topic as a Target for a Notification Rule](#) in the *AWS Developer Tools Console User Guide*.

- For API details, see [Unsubscribe](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag from a notification rule

The following `untag-resource` example removes the tag with the key name `Team` from the specified notification rule.

```
aws codestar-notifications untag-resource \
  --arn arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/
fe1efd35-EXAMPLE \
  --tag-keys Team
```

This command produces no output.

For more information, see [Edit a Notification Rule](#) in the *AWS Developer Tools Console User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-notification-rule

The following code example shows how to use `update-notification-rule`.

AWS CLI

To update a notification rule

The following `update-notification-rule` example updates a notification rule named `MyNotificationRule` in the AWS account `123456789012` using a JSON file named `update.json`.

```
aws codestar-notifications update-notification-rule \  
  --cli-input-json file://update.json
```

Contents of `update.json`:

```
{  
  "Name": "MyUpdatedNotificationRule",  
  "EventTypeIds": [  
    "codecommit-repository-branches-and-tags-created"  
  ],  
  "Resource": "arn:aws:codecommit:us-east-1:123456789012:MyDemoRepo",  
  "Targets": [  
    {  
      "TargetType": "SNS",  
      "TargetAddress": "arn:aws:sns:us-  
east-1:123456789012:MyNotificationTopic"  
    }  
  ],  
  "Status": "ENABLED",  
  "DetailType": "FULL"  
}
```

Output:

```
{  
  "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/  
dc82df7a-EXAMPLE"  
}
```

For more information, see [Edit a notification rule](#) in the *AWS Developer Tools Console User Guide*.

- For API details, see [UpdateNotificationRule](#) in *AWS CLI Command Reference*.

CodeConnections examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with CodeConnections.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-connection

The following code example shows how to use `create-connection`.

AWS CLI

To create a connection

The following `create-connection` example shows how to create a connection to a third-party repository. This example creates a connection where the third-party provider is Bitbucket.

A connection created through the AWS CLI or AWS CloudFormation is in Pending status by default. After you create a connection with the CLI or AWS CloudFormation, use the console to edit the connection to make its status Available.

```
aws codestar-connections create-connection \  
  --provider-type Bitbucket \  
  --connection-name MyConnection
```

Output:

```
{  
  "ConnectionArn": "arn:aws:codestar-connections:us-  
east-1:123456789012:connection/aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f"  
}
```

For more information, see [Create a connection](#) in the *Developer Tools console User Guide*.

- For API details, see [CreateConnection](#) in *AWS CLI Command Reference*.

create-host

The following code example shows how to use `create-host`.

AWS CLI

To create a host

The following `create-host` example shows how to create a host to represent the endpoint for the infrastructure where your third-party provider is installed. This example creates a host where the third-party installed provider is GitHub Enterprise Server.

A host created through the AWS CLI is in Pending status by default. After you create a host with the CLI, use the console or the CLI to set up the host to make its status Available.

```
aws codestar-connections create-host \  
  --name MyHost \  
  --provider-type GitHubEnterpriseServer \  
  --provider-endpoint "https://my-instance.dev"
```

Output:

```
{  
  "HostArn": "arn:aws:codestar-connections:us-east-1:123456789012:host/My-  
Host-28aef605"  
}
```

For more information, see [Create a host \(CLI\)](#) in the *Developer Tools console User Guide*.

- For API details, see [CreateHost](#) in *AWS CLI Command Reference*.

delete-connection

The following code example shows how to use `delete-connection`.

AWS CLI

To delete a connection

The following `delete-connection` example shows how to delete a connection.

```
aws codestar-connections delete-connection \  
  --connection-arn arn:aws:codestar-connections:us-west-2:123456789012:connection/  
aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f
```

This command produces no output.

For more information, see [Delete a connection \(CLI\)](#) in the *Developer Tools console User Guide*.

- For API details, see [DeleteConnection](#) in *AWS CLI Command Reference*.

delete-host

The following code example shows how to use `delete-host`.

AWS CLI

To delete a host

The following `delete-host` example shows how to delete a host. Before you can delete a host, you must delete all connections associated with the host.

```
aws codestar-connections delete-host \  
  --host-arn "arn:aws:codestar-connections:us-east-1 :123456789012:host/My-  
Host-28aef605"
```

This command produces no output.

For more information, see [Delete a host \(CLI\)](#) in the *Developer Tools console User Guide*.

- For API details, see [DeleteHost](#) in *AWS CLI Command Reference*.

get-connection

The following code example shows how to use `get-connection`.

AWS CLI

To get information about a connection

The following `get-connection` example shows details about a connection.


```
aws codestar-connections get-connection \  
  --connection-arn arn:aws:codestar-connections:us-east-1:123456789012:connection/  
aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f
```

Output:

```
{  
  "Connection": {  
    "ConnectionName": "MyConnection",  
    "ConnectionArn": "arn:aws:codestar-connections:us-  
east-1:123456789012:connection/aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f",  
    "ProviderType": "Bitbucket",  
    "OwnerAccountId": "123456789012",  
    "ConnectionStatus": "AVAILABLE"  
  }  
}
```

For more information, see [View connection details](#) in the *Developer Tools console User Guide*.

- For API details, see [GetConnection](#) in *AWS CLI Command Reference*.

get-host

The following code example shows how to use `get-host`.

AWS CLI

To get information about a host

The following `get-host` example shows details about a host:

```
aws codestar-connections get-host \  
  --host-arn arn:aws:codestar-connections:us-east-1:123456789012:host/  
MyHost-28aef605
```

Output:

```
{  
  "Name": "MyHost",  
  "Status": "AVAILABLE",  
  "ProviderType": "GitHubEnterpriseServer",  
  "ProviderEndpoint": "https://test-instance-1.dev/"
```

```
}
```

For more information, see [View host details \(CLI\)](#) in the *Developer Tools console User Guide*.

- For API details, see [GetHost](#) in *AWS CLI Command Reference*.

list-connections

The following code example shows how to use `list-connections`.

AWS CLI

To list connections

The following `list-connections` example retrieves a list of all connections in your account for the Bitbucket provider type.:

```
aws codestar-connections list-connections \  
--provider-type Bitbucket \  
--max-results 5 \  
--next-token: next-token
```

Output:

```
{  
  "Connections": [  
    {  
      "ConnectionName": "my-connection",  
      "ProviderType": "Bitbucket",  
      "Status": "PENDING",  
      "ARN": "arn:aws:codestar-connections:us-east-1:123456789012:connection/  
aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f",  
      "OwnerId": "123456789012"  
    },  
    {  
      "ConnectionName": "my-other-connection",  
      "ProviderType": "Bitbucket",  
      "Status": "AVAILABLE",  
      "ARN": "arn:aws:codestar-connections:us-east-1:123456789012:connection/  
aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f",  
      "OwnerId": "123456789012"  
    },  
  ]  
}
```

```
  ],  
  "NextToken": "next-token"  
}
```

For more information, see [List connections \(CLI\)](#) in the *Developer Tools console User Guide*.

- For API details, see [ListConnections](#) in *AWS CLI Command Reference*.

list-hosts

The following code example shows how to use `list-hosts`.

AWS CLI

To list hosts

The following `list-hosts` example retrieves a list of all hosts in your account.

```
aws codestar-connections list-hosts
```

Output:

```
{  
  "Hosts": [  
    {  
      "Name": "My-Host",  
      "HostArn": "arn:aws:codestar-connections:us-east-1:123456789012:host/My-Host-28aef605",  
      "ProviderType": "GitHubEnterpriseServer",  
      "ProviderEndpoint": "https://my-instance.test.dev",  
      "Status": "AVAILABLE"  
    }  
  ]  
}
```

For more information, see [List hosts \(CLI\)](#) in the *Developer Tools console User Guide*.

- For API details, see [ListHosts](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags

The following `list-tags-for-resource` example retrieves a list of all tags attached to the specified connections resource.

```
aws codestar-connections list-tags-for-resource \  
  --resource-arn arn:aws:codestar-connections:us-east-1:123456789012:connection/  
aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f
```

Output:

```
{  
  "Tags": [  
    {  
      "Key": "Project",  
      "Value": "ProjectA"  
    },  
    {  
      "Key": "ReadOnly",  
      "Value": "true"  
    }  
  ]  
}
```

For more information, see [View tags for a connections resource](#) in the *Developer Tools console User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To tag a resource

The following `tag-resource` example associates a set of provided tags with a connection. Use this command to add or edit tags.

```
aws codestar-connections tag-resource \  
  --resource-arn arn:aws:codestar-connections:us-east-1:123456789012:connection/  
aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f
```

```
--resource-arn arn:aws:codestar-connections:us-east-1:123456789012:connection/
aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f \
--tags Key=Project,Value=ProjectA Key=IscontainerBased,Value=true
```

This command produces no output.

For more information, see [Add tags to a connections resource](#) in the *Developer Tools console User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove AWS tags from a connections resource

The following `untag-resource` removes a tag from the specified resource.

```
aws codestar-connections untag-resource \
--resource-arn arn:aws:codestar-connections:us-east-1:123456789012:connection/
aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f \
--tag-keys Project ReadOnly
```

Output:

```
{
  "Tags": []
}
```

For more information, see [Remove tags from a connections resource](#) in the *Developer Tools console User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

Amazon Cognito Identity examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon Cognito Identity.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-identity-pool

The following code example shows how to use `create-identity-pool`.

AWS CLI

To create an identity pool with Cognito identity pool provider

This example creates an identity pool named `MyIdentityPool`. It has a Cognito identity pool provider. Unauthenticated identities are not allowed.

Command:

```
aws cognito-identity create-identity-pool --identity-pool-name
  MyIdentityPool --no-allow-unauthenticated-identities --cognito-
  identity-providers ProviderName="cognito-idp.us-west-2.amazonaws.com/us-
  west-2_aaaaaaaaa",ClientId="3n4b5urk1ft4fl3mg5e62d9ado",ServerSideTokenCheck=false
```

Output:

```
{
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
  "IdentityPoolName": "MyIdentityPool",
  "AllowUnauthenticatedIdentities": false,
  "CognitoIdentityProviders": [
```

```
{
  "ProviderName": "cognito-idp.us-west-2.amazonaws.com/us-west-2_111111111",
  "ClientId": "3n4b5urk1ft4f13mg5e62d9ado",
  "ServerSideTokenCheck": false
}
]
```

- For API details, see [CreateIdentityPool](#) in *AWS CLI Command Reference*.

delete-identities

The following code example shows how to use `delete-identities`.

AWS CLI

To delete identity pool

This example deletes an identity pool.

Command:

```
aws cognito-identity delete-identity-pool --identity-ids-to-delete "us-west-2:111111111-1111-1111-1111-111111111111"
```

Output:

```
{
  "UnprocessedIdentityIds": []
}
```

- For API details, see [DeleteIdentities](#) in *AWS CLI Command Reference*.

delete-identity-pool

The following code example shows how to use `delete-identity-pool`.

AWS CLI

To delete identity pool

The following `delete-identity-pool` example deletes the specified identity pool.

Command:

```
aws cognito-identity delete-identity-pool \  
  --identity-pool-id "us-west-2:11111111-1111-1111-1111-111111111111"
```

This command produces no output.

- For API details, see [DeleteIdentityPool](#) in *AWS CLI Command Reference*.

describe-identity-pool

The following code example shows how to use `describe-identity-pool`.

AWS CLI

To describe an identity pool

This example describes an identity pool.

Command:

```
aws cognito-identity describe-identity-pool --identity-pool-id "us-  
west-2:11111111-1111-1111-1111-111111111111"
```

Output:

```
{  
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",  
  "IdentityPoolName": "MyIdentityPool",  
  "AllowUnauthenticatedIdentities": false,  
  "CognitoIdentityProviders": [  
    {  
      "ProviderName": "cognito-idp.us-west-2.amazonaws.com/us-west-2_11111111",  
      "ClientId": "3n4b5urk1ft4fl3mg5e62d9ado",  
      "ServerSideTokenCheck": false  
    }  
  ]  
}
```

- For API details, see [DescribeIdentityPool](#) in *AWS CLI Command Reference*.

get-identity-pool-roles

The following code example shows how to use `get-identity-pool-roles`.

AWS CLI

To get identity pool roles

This example gets identity pool roles.

Command:

```
aws cognito-identity get-identity-pool-roles --identity-pool-id "us-west-2:11111111-1111-1111-1111-111111111111"
```

Output:

```
{
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
  "Roles": {
    "authenticated": "arn:aws:iam::111111111111:role/Cognito_MyIdentityPoolAuth_Role",
    "unauthenticated": "arn:aws:iam::111111111111:role/Cognito_MyIdentityPoolUnauth_Role"
  }
}
```

- For API details, see [GetIdentityPoolRoles](#) in *AWS CLI Command Reference*.

list-identity-pools

The following code example shows how to use `list-identity-pools`.

AWS CLI

To list identity pools

This example lists identity pools. There s a maximum of 20 identities listed.

Command:

```
aws cognito-identity list-identity-pools --max-results 20
```

Output:

```
{
  "IdentityPools": [
    {
      "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
      "IdentityPoolName": "MyIdentityPool"
    },
    {
      "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
      "IdentityPoolName": "AnotherIdentityPool"
    },
    {
      "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
      "IdentityPoolName": "IdentityPoolRegionA"
    }
  ]
}
```

- For API details, see [ListIdentityPools](#) in *AWS CLI Command Reference*.

set-identity-pool-roles

The following code example shows how to use `set-identity-pool-roles`.

AWS CLI**To set identity pool roles**

The following `set-identity-pool-roles` example sets an identity pool role.

```
aws cognito-identity set-identity-pool-roles \
  --identity-pool-id "us-west-2:11111111-1111-1111-1111-111111111111" \
  --roles authenticated="arn:aws:iam::111111111111:role/
Cognito_MyIdentityPoolAuth_Role"
```

- For API details, see [SetIdentityPoolRoles](#) in *AWS CLI Command Reference*.

update-identity-pool

The following code example shows how to use `update-identity-pool`.

AWS CLI

To update an identity pool

This example updates an identity pool. It sets the name to MyIdentityPool. It adds Cognito as an identity provider. It disallows unauthenticated identities.

Command:

```
aws cognito-identity update-identity-pool --identity-pool-id "us-west-2:11111111-1111-1111-1111-111111111111" --identity-pool-name "MyIdentityPool" --no-allow-unauthenticated-identities --cognito-identity-providers ProviderName="cognito-idp.us-west-2.amazonaws.com/us-west-2_1111111111",ClientId="3n4b5urk1ft4f13mg5e62d9ado",ServerSideTokenCheck=false
```

Output:

```
{
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
  "IdentityPoolName": "MyIdentityPool",
  "AllowUnauthenticatedIdentities": false,
  "CognitoIdentityProviders": [
    {
      "ProviderName": "cognito-idp.us-west-2.amazonaws.com/us-west-2_1111111111",
      "ClientId": "3n4b5urk1ft4f13mg5e62d9ado",
      "ServerSideTokenCheck": false
    }
  ]
}
```

- For API details, see [UpdateIdentityPool](#) in *AWS CLI Command Reference*.

Amazon Cognito Identity Provider examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon Cognito Identity Provider.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-custom-attributes

The following code example shows how to use `add-custom-attributes`.

AWS CLI

To add a custom attribute

This example adds a custom attribute `CustomAttr1` to a user pool. It is a String type, and requires a minimum of 1 character and a maximum of 15. It is not required.

Command:

```
aws cognito-idp add-custom-attributes --user-pool-id us-west-2_aaaaaaaaa --custom-attributes Name="CustomAttr1",AttributeDataType="String",DeveloperOnlyAttribute=false,Required=false,S
```

- For API details, see [AddCustomAttributes](#) in *AWS CLI Command Reference*.

admim-disable-user

The following code example shows how to use `admim-disable-user`.

AWS CLI

To disable a user

This example disables user `jane@example.com`.

Command:

```
aws cognito-idp admin-disable-user --user-pool-id us-west-2_aaaaaaaa --username jane@example.com
```

- For API details, see [AdminDisableUser](#) in *AWS CLI Command Reference*.

admin-enable-user

The following code example shows how to use `admin-enable-user`.

AWS CLI**To enable a user**

This example enables username `jane@example.com`.

Command:

```
aws cognito-idp admin-enable-user --user-pool-id us-west-2_aaaaaaaa --username jane@example.com
```

- For API details, see [AdminEnableUser](#) in *AWS CLI Command Reference*.

admin-add-user-to-group

The following code example shows how to use `admin-add-user-to-group`.

AWS CLI**To add a user to a group**

This example adds user Jane to group MyGroup.

Command:

```
aws cognito-idp admin-add-user-to-group --user-pool-id us-west-2_aaaaaaaa --username Jane --group-name MyGroup
```

- For API details, see [AdminAddUserToGroup](#) in *AWS CLI Command Reference*.

admin-confirm-sign-up

The following code example shows how to use `admin-confirm-sign-up`.

AWS CLI

To confirm user registration

This example confirms user `jane@example.com`.

Command:

```
aws cognito-idp admin-confirm-sign-up --user-pool-id us-west-2_aaaaaaaaa --username jane@example.com
```

- For API details, see [AdminConfirmSignUp](#) in *AWS CLI Command Reference*.

admin-create-user

The following code example shows how to use `admin-create-user`.

AWS CLI

To create a user

The following `admin-create-user` example creates a user with the specified settings email address and phone number.

```
aws cognito-idp admin-create-user \
  --user-pool-id us-west-2_aaaaaaaaa \
  --username diego \
  --user-attributes Name=email,Value=diego@example.com
Name=phone_number,Value="+15555551212" \
  --message-action SUPPRESS
```

Output:

```
{
  "User": {
    "Username": "diego",
    "Attributes": [
```

```
    {
      "Name": "sub",
      "Value": "7325c1de-b05b-4f84-b321-9adc6e61f4a2"
    },
    {
      "Name": "phone_number",
      "Value": "+15555551212"
    },
    {
      "Name": "email",
      "Value": "diego@example.com"
    }
  ],
  "UserCreateDate": 1548099495.428,
  "UserLastModifiedDate": 1548099495.428,
  "Enabled": true,
  "UserStatus": "FORCE_CHANGE_PASSWORD"
}
```

- For API details, see [AdminCreateUser](#) in *AWS CLI Command Reference*.

admin-delete-user-attributes

The following code example shows how to use `admin-delete-user-attributes`.

AWS CLI

To delete a user attribute

This example deletes a custom attribute `CustomAttr1` for user `diego@example.com`.

Command:

```
aws cognito-idp admin-delete-user-attributes --user-pool-id us-west-2_aaaaaaaaa --
username diego@example.com --user-attribute-names "custom:CustomAttr1"
```

- For API details, see [AdminDeleteUserAttributes](#) in *AWS CLI Command Reference*.

admin-delete-user

The following code example shows how to use `admin-delete-user`.

AWS CLI

To delete a user

This example deletes a user.

Command:

```
aws cognito-idp admin-delete-user --user-pool-id us-west-2_aaaaaaaaa --username
diego@example.com
```

- For API details, see [AdminDeleteUser](#) in *AWS CLI Command Reference*.

admin-forget-device

The following code example shows how to use admin-forget-device.

AWS CLI

To forget a device

This example forgets device for username jane@example.com

Command:

```
aws cognito-idp admin-forget-device --user-pool-id us-west-2_aaaaaaaaa --username
jane@example.com --device-key us-west-2_abcd_1234-5678
```

- For API details, see [AdminForgetDevice](#) in *AWS CLI Command Reference*.

admin-get-device

The following code example shows how to use admin-get-device.

AWS CLI

To get a device

This example gets a device for username jane@example.com

Command:


```
aws cognito-idp admin-get-device --user-pool-id us-west-2_aaaaaaaaa --username
jane@example.com --device-key us-west-2_abcd_1234-5678
```

- For API details, see [AdminGetDevice](#) in *AWS CLI Command Reference*.

admin-get-user

The following code example shows how to use `admin-get-user`.

AWS CLI

To get a user

This example gets information about username `jane@example.com`.

Command:

```
aws cognito-idp admin-get-user --user-pool-id us-west-2_aaaaaaaaa --username
jane@example.com
```

Output:

```
{
  "Username": "4320de44-2322-4620-999b-5e2e1c8df013",
  "Enabled": true,
  "UserStatus": "FORCE_CHANGE_PASSWORD",
  "UserCreateDate": 1548108509.537,
  "UserAttributes": [
    {
      "Name": "sub",
      "Value": "4320de44-2322-4620-999b-5e2e1c8df013"
    },
    {
      "Name": "email_verified",
      "Value": "true"
    },
    {
      "Name": "phone_number_verified",
      "Value": "true"
    },
    {
```

```
        "Name": "phone_number",
        "Value": "+01115551212"
    },
    {
        "Name": "email",
        "Value": "jane@example.com"
    }
],
"UserLastModifiedDate": 1548108509.537
}
```

- For API details, see [AdminGetUser](#) in *AWS CLI Command Reference*.

admin-initiate-auth

The following code example shows how to use `admin-initiate-auth`.

AWS CLI

To initiate authorization

This example initiates authorization using the `ADMIN_NO_SRP_AUTH` flow for username `jane@example.com`

The client must have sign-in API for server-based authentication (`ADMIN_NO_SRP_AUTH`) enabled.

Use the session information in the return value to call `admin-respond-to-auth-challenge`.

Command:

```
aws cognito-idp admin-initiate-auth --user-pool-id us-west-2_aaaaaaaaa --client-id 3n4b5urk1ft4fl3mg5e62d9ado --auth-flow ADMIN_NO_SRP_AUTH --auth-parameters USERNAME=jane@example.com,PASSWORD=password
```

Output:

```
{
  "ChallengeName": "NEW_PASSWORD_REQUIRED",
  "Session": "SESSION",
  "ChallengeParameters": {
```

```
"USER_ID_FOR_SRP": "84514837-dcbc-4af1-abff-f3c109334894",
"requiredAttributes": "[]",
"userAttributes": "{ \"email_verified\": \"true\", \"phone_number_verified\":
\"true\", \"phone_number\": \"+01xxx5550100\", \"email\": \"jane@example.com\" }"
}
```

- For API details, see [AdminInitiateAuth](#) in *AWS CLI Command Reference*.

admin-list-devices

The following code example shows how to use `admin-list-devices`.

AWS CLI

To list devices for a user

This example lists devices for username `jane@example.com`.

Command:

```
aws cognito-idp admin-list-devices --user-pool-id us-west-2_aaaaaaaaa --username
jane@example.com
```

- For API details, see [AdminListDevices](#) in *AWS CLI Command Reference*.

admin-list-groups-for-user

The following code example shows how to use `admin-list-groups-for-user`.

AWS CLI

To list groups for a user

This example lists groups for username `jane@example.com`.

Command:

```
aws cognito-idp admin-list-groups-for-user --user-pool-id us-west-2_aaaaaaaaa --
username diego@example.com
```

Output:

```
{
  "Groups": [
    {
      "Description": "Sample group",
      "Precedence": 1,
      "LastModifiedDate": 1548097827.125,
      "RoleArn": "arn:aws:iam::111111111111:role/SampleRole",
      "GroupName": "SampleGroup",
      "UserPoolId": "us-west-2_aaaaaaaaa",
      "CreationDate": 1548097827.125
    }
  ]
}
```

- For API details, see [AdminListGroupForUser](#) in *AWS CLI Command Reference*.

admin-list-user-auth-events

The following code example shows how to use `admin-list-user-auth-events`.

AWS CLI**To list authorization events for a user**

This example lists authorization events for username `diego@example.com`.

Command:

```
aws cognito-idp admin-list-user-auth-events --user-pool-id us-west-2_aaaaaaaaa --
username diego@example.com
```

- For API details, see [AdminListUserAuthEvents](#) in *AWS CLI Command Reference*.

admin-remove-user-from-group

The following code example shows how to use `admin-remove-user-from-group`.

AWS CLI**To remove a user from a group**

This example removes jane@example.com from SampleGroup.

Command:

```
aws cognito-idp admin-remove-user-from-group --user-pool-id us-west-2_aaaaaaaaa --username jane@example.com --group-name SampleGroup
```

- For API details, see [AdminRemoveUserFromGroup](#) in *AWS CLI Command Reference*.

admin-reset-user-password

The following code example shows how to use admin-reset-user-password.

AWS CLI

To reset a user password

This example resets the password for diego@example.com.

Command:

```
aws cognito-idp admin-reset-user-password --user-pool-id us-west-2_aaaaaaaaa --username diego@example.com
```

- For API details, see [AdminResetUserPassword](#) in *AWS CLI Command Reference*.

admin-set-user-mfa-preference

The following code example shows how to use admin-set-user-mfa-preference.

AWS CLI

To set the user MFA preference

This example sets the SMS MFA preference for username diego@example.com.

Command:

```
aws cognito-idp admin-set-user-mfa-preference --user-pool-id us-west-2_aaaaaaaaa --username diego@example.com --sms-mfa-settings Enabled=false,PreferredMfa=false
```

- For API details, see [AdminSetUserMfaPreference](#) in *AWS CLI Command Reference*.

admin-set-user-settings

The following code example shows how to use `admin-set-user-settings`.

AWS CLI

To set user settings

This example sets the MFA delivery preference for username `diego@example.com` to EMAIL.

Command:

```
aws cognito-idp admin-set-user-settings --user-pool-id us-west-2_aaaaaaaa --username diego@example.com --mfa-options DeliveryMedium=EMAIL
```

- For API details, see [AdminSetUserSettings](#) in *AWS CLI Command Reference*.

admin-update-auth-event-feedback

The following code example shows how to use `admin-update-auth-event-feedback`.

AWS CLI

To provide feedback for an authorization event

This example sets the feedback value for an authorization event identified by `event-id` to Valid.

Command:

```
aws cognito-idp admin-update-auth-event-feedback --user-pool-id us-west-2_aaaaaaaa --username diego@example.com --event-id c2c2cf89-c0d3-482d-aba6-99d78a5b0bfe --feedback-value Valid
```

- For API details, see [AdminUpdateAuthEventFeedback](#) in *AWS CLI Command Reference*.

admin-update-device-status

The following code example shows how to use `admin-update-device-status`.

AWS CLI

To update device status

This example sets the device remembered status for the device identified by device-key to not_remembered.

Command:

```
aws cognito-idp admin-update-device-status --user-pool-id us-west-2_aaaaaaaa
--username diego@example.com --device-key xxxx --device-remembered-status
not_remembered
```

- For API details, see [AdminUpdateDeviceStatus](#) in *AWS CLI Command Reference*.

admin-update-user-attributes

The following code example shows how to use admin-update-user-attributes.

AWS CLI

To update user attributes

This example updates a custom user attribute CustomAttr1 for user diego@example.com.

Command:

```
aws cognito-idp admin-update-user-attributes --user-pool-id us-
west-2_aaaaaaaa --username diego@example.com --user-attributes
Name="custom:CustomAttr1",Value="Purple"
```

- For API details, see [AdminUpdateUserAttributes](#) in *AWS CLI Command Reference*.

change-password

The following code example shows how to use change-password.

AWS CLI

To change a password

This example changes a password.

Command:

```
aws cognito-idp change-password --previous-password OldPassword --proposed-password  
NewPassword --access-token ACCESS_TOKEN
```

- For API details, see [ChangePassword](#) in *AWS CLI Command Reference*.

confirm-forgot-password

The following code example shows how to use `confirm-forgot-password`.

AWS CLI**To confirm a forgotten password**

This example confirms a forgotten password for username `diego@example.com`.

Command:

```
aws cognito-idp confirm-forgot-password --client-id 3n4b5urk1ft4f13mg5e62d9ado --  
username=diego@example.com --password PASSWORD --confirmation-code CONF_CODE
```

- For API details, see [ConfirmForgotPassword](#) in *AWS CLI Command Reference*.

confirm-sign-up

The following code example shows how to use `confirm-sign-up`.

AWS CLI**To confirm sign-up**

This example confirms sign-up for username `diego@example.com`.

Command:

```
aws cognito-idp confirm-sign-up --client-id 3n4b5urk1ft4f13mg5e62d9ado --  
username=diego@example.com --confirmation-code CONF_CODE
```

- For API details, see [ConfirmSignUp](#) in *AWS CLI Command Reference*.

create-group

The following code example shows how to use create-group.

AWS CLI

To create a group

This example creates a group with a description.

Command:

```
aws cognito-idp create-group --user-pool-id us-west-2_aaaaaaaa --group-name
MyNewGroup --description "New group."
```

Output:

```
{
  "Group": {
    "GroupName": "MyNewGroup",
    "UserPoolId": "us-west-2_aaaaaaaa",
    "Description": "New group.",
    "LastModifiedDate": 1548270073.795,
    "CreationDate": 1548270073.795
  }
}
```

To create a group with a role and precedence

This example creates a group with a description. It also includes a role and precedence.

Command:

```
aws cognito-idp create-group --user-pool-id us-west-2_aaaaaaaa --group-
name MyNewGroupWithRole --description "New group with a role." --role-arn
arn:aws:iam::111111111111:role/MyNewGroupRole --precedence 2
```

Output:

```
{
  "Group": {
    "GroupName": "MyNewGroupWithRole",
    "UserPoolId": "us-west-2_aaaaaaaa",
```

```
"Description": "New group with a role.",
"RoleArn": "arn:aws:iam::111111111111:role/MyNewGroupRole",
"Precedence": 2,
"LastModifiedDate": 1548270211.761,
"CreationDate": 1548270211.761
}
}
```

- For API details, see [CreateGroup](#) in *AWS CLI Command Reference*.

create-user-import-job

The following code example shows how to use `create-user-import-job`.

AWS CLI

To create a user import job

This example creates a user import job named `MyImportJob`.

For more information about importing users, see [Importing Users into User Pools From a CSV File](#).

Command:

```
aws cognito-idp create-user-import-job --user-pool-id us-west-2_aaaaaaaaa --
job-name MyImportJob --cloud-watch-logs-role-arn arn:aws:iam::111111111111:role/
CognitoCloudWatchLogsRole
```

Output:

```
{
  "UserImportJob": {
    "JobName": "MyImportJob",
    "JobId": "import-qQ0DCt2fRh",
    "UserPoolId": "us-west-2_aaaaaaaaa",
    "PreSignedUrl": "PRE_SIGNED_URL",
    "CreationDate": 1548271795.471,
    "Status": "Created",
    "CloudWatchLogsRoleArn": "arn:aws:iam::111111111111:role/
CognitoCloudWatchLogsRole",
    "ImportedUsers": 0,
  }
}
```

```
    "SkippedUsers": 0,  
    "FailedUsers": 0  
  }  
}
```

Upload the .csv file with curl using the pre-signed URL:

Command:

```
curl -v -T "PATH_TO_CSV_FILE" -H "x-amz-server-side-encryption:aws:kms"  
"PRE_SIGNED_URL"
```

- For API details, see [CreateUserImportJob](#) in *AWS CLI Command Reference*.

create-user-pool-client

The following code example shows how to use create-user-pool-client.

AWS CLI

To create a user pool client

This example creates a new user pool client with two explicit authorization flows: USER_PASSWORD_AUTH and ADMIN_NO_SRP_AUTH.

Command:

```
aws cognito-idp create-user-pool-client --user-pool-id us-west-2_aaaaaaaaa  
--client-name MyNewClient --no-generate-secret --explicit-auth-flows  
"USER_PASSWORD_AUTH" "ADMIN_NO_SRP_AUTH"
```

Output:

```
{  
  "UserPoolClient": {  
    "UserPoolId": "us-west-2_aaaaaaaaa",  
    "ClientName": "MyNewClient",  
    "ClientId": "6p3bs000no6a4ue1idruvd05ad",  
    "LastModifiedDate": 1548697449.497,  
    "CreationDate": 1548697449.497,  
    "RefreshTokenValidity": 30,  
    "ExplicitAuthFlows": [  
      "USER_PASSWORD_AUTH",  
      "ADMIN_NO_SRP_AUTH"  
    ]  
  }  
}
```

```
        "USER_PASSWORD_AUTH",
        "ADMIN_NO_SRP_AUTH"
    ],
    "AllowedOAuthFlowsUserPoolClient": false
}
}
```

- For API details, see [CreateUserPoolClient](#) in *AWS CLI Command Reference*.

create-user-pool-domain

The following code example shows how to use `create-user-pool-domain`.

AWS CLI

To create a user pool domain

This example creates a new user pool domain. with two explicit authorization flows: `USER_PASSWORD_AUTH` and `ADMIN_NO_SRP_AUTH`.

Command:

```
aws cognito-idp create-user-pool-domain --user-pool-id us-west-2_aaaaaaaaa --domain
my-new-domain
```

- For API details, see [CreateUserPoolDomain](#) in *AWS CLI Command Reference*.

create-user-pool

The following code example shows how to use `create-user-pool`.

AWS CLI

To create a minimally configured user pool

This example creates a user pool named `MyUserPool` using default values. There are no required attributes and no application clients. MFA and advanced security is disabled.

Command:

```
aws cognito-idp create-user-pool --pool-name MyUserPool
```

Output:

```
{
  "UserPool": {
    "SchemaAttributes": [
      {
        "Name": "sub",
        "StringAttributeConstraints": {
          "MinLength": "1",
          "MaxLength": "2048"
        },
        "DeveloperOnlyAttribute": false,
        "Required": true,
        "AttributeDataType": "String",
        "Mutable": false
      },
      {
        "Name": "name",
        "StringAttributeConstraints": {
          "MinLength": "0",
          "MaxLength": "2048"
        },
        "DeveloperOnlyAttribute": false,
        "Required": false,
        "AttributeDataType": "String",
        "Mutable": true
      },
      {
        "Name": "given_name",
        "StringAttributeConstraints": {
          "MinLength": "0",
          "MaxLength": "2048"
        },
        "DeveloperOnlyAttribute": false,
        "Required": false,
        "AttributeDataType": "String",
        "Mutable": true
      },
      {
        "Name": "family_name",
        "StringAttributeConstraints": {
          "MinLength": "0",
          "MaxLength": "2048"
        }
      }
    ]
  }
}
```

```
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "middle_name",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "nickname",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "preferred_username",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "profile",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },

```

```
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "picture",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "website",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "email",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "AttributeDataType": "Boolean",
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "Name": "email_verified",
    "Mutable": true
  }
}
```

```
  },
  {
    "Name": "gender",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "birthdate",
    "StringAttributeConstraints": {
      "MinLength": "10",
      "MaxLength": "10"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "zoneinfo",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "locale",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  }
}
```



```
    },
    {
      "Name": "phone_number",
      "StringAttributeConstraints": {
        "MinLength": "0",
        "MaxLength": "2048"
      },
      "DeveloperOnlyAttribute": false,
      "Required": false,
      "AttributeDataType": "String",
      "Mutable": true
    },
    {
      "AttributeDataType": "Boolean",
      "DeveloperOnlyAttribute": false,
      "Required": false,
      "Name": "phone_number_verified",
      "Mutable": true
    },
    {
      "Name": "address",
      "StringAttributeConstraints": {
        "MinLength": "0",
        "MaxLength": "2048"
      },
      "DeveloperOnlyAttribute": false,
      "Required": false,
      "AttributeDataType": "String",
      "Mutable": true
    },
    {
      "Name": "updated_at",
      "NumberAttributeConstraints": {
        "MinValue": "0"
      },
      "DeveloperOnlyAttribute": false,
      "Required": false,
      "AttributeDataType": "Number",
      "Mutable": true
    }
  ],
  "MfaConfiguration": "OFF",
  "Name": "MyUserPool",
  "LastModifiedDate": 1547833345.777,
```

```

    "AdminCreateUserConfig": {
      "UnusedAccountValidityDays": 7,
      "AllowAdminCreateUserOnly": false
    },
    "EmailConfiguration": {},
    "Policies": {
      "PasswordPolicy": {
        "RequireLowercase": true,
        "RequireSymbols": true,
        "RequireNumbers": true,
        "MinimumLength": 8,
        "RequireUppercase": true
      }
    },
    "CreationDate": 1547833345.777,
    "EstimatedNumberOfUsers": 0,
    "Id": "us-west-2_aaaaaaaaaa",
    "LambdaConfig": {}
  }
}

```

To create a user pool with two required attributes

This example creates a user pool MyUserPool. The pool is configured to accept email as a username attribute. It also sets the email source address to a validated address using Amazon Simple Email Service.

Command:

```

aws cognito-idp create-user-pool --pool-name MyUserPool --username-attributes "email" --email-configuration=SourceArn="arn:aws:ses:us-east-1:111111111111:identity/jane@example.com",ReplyToEmailAddress="jane@example.com"

```

Output:

```

{
  "UserPool": {
    "SchemaAttributes": [
      {
        "Name": "sub",
        "StringAttributeConstraints": {
          "MinLength": "1",

```

```
        "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": true,
    "AttributeDataType": "String",
    "Mutable": false
},
{
    "Name": "name",
    "StringAttributeConstraints": {
        "MinLength": "0",
        "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
},
{
    "Name": "given_name",
    "StringAttributeConstraints": {
        "MinLength": "0",
        "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
},
{
    "Name": "family_name",
    "StringAttributeConstraints": {
        "MinLength": "0",
        "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
},
{
    "Name": "middle_name",
    "StringAttributeConstraints": {
        "MinLength": "0",
```

```
        "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
},
{
    "Name": "nickname",
    "StringAttributeConstraints": {
        "MinLength": "0",
        "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
},
{
    "Name": "preferred_username",
    "StringAttributeConstraints": {
        "MinLength": "0",
        "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
},
{
    "Name": "profile",
    "StringAttributeConstraints": {
        "MinLength": "0",
        "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
},
{
    "Name": "picture",
    "StringAttributeConstraints": {
        "MinLength": "0",
```

```
        "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
},
{
    "Name": "website",
    "StringAttributeConstraints": {
        "MinLength": "0",
        "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
},
{
    "Name": "email",
    "StringAttributeConstraints": {
        "MinLength": "0",
        "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
},
{
    "AttributeDataType": "Boolean",
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "Name": "email_verified",
    "Mutable": true
},
{
    "Name": "gender",
    "StringAttributeConstraints": {
        "MinLength": "0",
        "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
```

```
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "birthdate",
    "StringAttributeConstraints": {
      "MinLength": "10",
      "MaxLength": "10"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "zoneinfo",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "locale",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "phone_number",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
```

```
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "AttributeDataType": "Boolean",
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "Name": "phone_number_verified",
    "Mutable": true
  },
  {
    "Name": "address",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "updated_at",
    "NumberAttributeConstraints": {
      "MinValue": "0"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "Number",
    "Mutable": true
  }
],
"MfaConfiguration": "OFF",
"Name": "MyUserPool",
"LastModifiedDate": 1547837788.189,
"AdminCreateUserConfig": {
  "UnusedAccountValidityDays": 7,
  "AllowAdminCreateUserOnly": false
},
"EmailConfiguration": {
  "ReplyToEmailAddress": "jane@example.com",
  "SourceArn": "arn:aws:ses:us-east-1:111111111111:identity/
jane@example.com"
},
}
```

```
"Policies": {
  "PasswordPolicy": {
    "RequireLowercase": true,
    "RequireSymbols": true,
    "RequireNumbers": true,
    "MinimumLength": 8,
    "RequireUppercase": true
  }
},
"UsernameAttributes": [
  "email"
],
"CreationDate": 1547837788.189,
"EstimatedNumberOfUsers": 0,
"Id": "us-west-2_aaaaaaaaaa",
"LambdaConfig": {}
}
}
```

- For API details, see [CreateUserPool](#) in *AWS CLI Command Reference*.

delete-group

The following code example shows how to use delete-group.

AWS CLI

To delete a group

This example deletes a group.

Command:

```
aws cognito-idp delete-group --user-pool-id us-west-2_aaaaaaaaaa --group-name
MyGroupName
```

- For API details, see [DeleteGroup](#) in *AWS CLI Command Reference*.

delete-identity-provider

The following code example shows how to use delete-identity-provider.

AWS CLI

To delete an identity provider

This example deletes an identity provider.

Command:

```
aws cognito-idp delete-identity-provider --user-pool-id us-west-2_aaaaaaaa --
provider-name Facebook
```

- For API details, see [DeleteIdentityProvider](#) in *AWS CLI Command Reference*.

delete-resource-server

The following code example shows how to use delete-resource-server.

AWS CLI

To delete a resource server

This example deletes a resource server named weather.example.com.

Command:

```
aws cognito-idp delete-resource-server --user-pool-id us-west-2_aaaaaaaa --
identifier weather.example.com
```

- For API details, see [DeleteResourceServer](#) in *AWS CLI Command Reference*.

delete-user-attributes

The following code example shows how to use delete-user-attributes.

AWS CLI

To delete user attributes

This example deletes the user attribute "FAVORITE_ANIMAL".

Command:

```
aws cognito-idp delete-user-attributes --access-token ACCESS_TOKEN --user-attribute-names "FAVORITE_ANIMAL"
```

- For API details, see [DeleteUserAttributes](#) in *AWS CLI Command Reference*.

delete-user-pool-client

The following code example shows how to use `delete-user-pool-client`.

AWS CLI

To delete a user pool client

This example deletes a user pool client.

Command:

```
aws cognito-idp delete-user-pool-client --user-pool-id us-west-2_aaaaaaaaa --client-id 38fjsnc484p94kpbsnet7mpld0
```

- For API details, see [DeleteUserPoolClient](#) in *AWS CLI Command Reference*.

delete-user-pool-domain

The following code example shows how to use `delete-user-pool-domain`.

AWS CLI

To delete a user pool domain

The following `delete-user-pool-domain` example deletes a user pool domain named `my-domain`

```
aws cognito-idp delete-user-pool-domain \
  --user-pool-id us-west-2_aaaaaaaaa \
  --domain my-domain
```

- For API details, see [DeleteUserPoolDomain](#) in *AWS CLI Command Reference*.

delete-user-pool

The following code example shows how to use `delete-user-pool`.

AWS CLI

To delete a user pool

This example deletes a user pool using the user pool id, `us-west-2_aaaaaaaaa`.

Command:

```
aws cognito-idp delete-user-pool --user-pool-id us-west-2_aaaaaaaaa
```

- For API details, see [DeleteUserPool](#) in *AWS CLI Command Reference*.

delete-user

The following code example shows how to use `delete-user`.

AWS CLI

To delete a user

This example deletes a user.

Command:

```
aws cognito-idp delete-user --access-token ACCESS_TOKEN
```

- For API details, see [DeleteUser](#) in *AWS CLI Command Reference*.

describe-identity-provider

The following code example shows how to use `describe-identity-provider`.

AWS CLI

To describe an identity provider

This example describes an identity provider named Facebook.

Command:

```
aws cognito-idp describe-identity-provider --user-pool-id us-west-2_aaaaaaaaa --
provider-name Facebook
```

Output:

```
{
  "IdentityProvider": {
    "UserPoolId": "us-west-2_aaaaaaaaa",
    "ProviderName": "Facebook",
    "ProviderType": "Facebook",
    "ProviderDetails": {
      "attributes_url": "https://graph.facebook.com/me?fields=",
      "attributes_url_add_attributes": "true",
      "authorize_scopes": "myscope",
      "authorize_url": "https://www.facebook.com/v2.9/dialog/oauth",
      "client_id": "11111",
      "client_secret": "11111",
      "token_request_method": "GET",
      "token_url": "https://graph.facebook.com/v2.9/oauth/access_token"
    },
    "AttributeMapping": {
      "username": "id"
    },
    "IdpIdentifiers": [],
    "LastModifiedDate": 1548105901.736,
    "CreationDate": 1548105901.736
  }
}
```

- For API details, see [DescribeIdentityProvider](#) in *AWS CLI Command Reference*.

describe-resource-server

The following code example shows how to use `describe-resource-server`.

AWS CLI

To describe a resource server

This example describes the resource server `weather.example.com`.

Command:

```
aws cognito-idp describe-resource-server --user-pool-id us-west-2_aaaaaaaaa --
identifier weather.example.com
```

Output:

```
{
  "ResourceServer": {
    "UserPoolId": "us-west-2_aaaaaaaaa",
    "Identifier": "weather.example.com",
    "Name": "Weather",
    "Scopes": [
      {
        "ScopeName": "weather.update",
        "ScopeDescription": "Update weather forecast"
      },
      {
        "ScopeName": "weather.read",
        "ScopeDescription": "Read weather forecasts"
      },
      {
        "ScopeName": "weather.delete",
        "ScopeDescription": "Delete a weather forecast"
      }
    ]
  }
}
```

- For API details, see [DescribeResourceServer](#) in *AWS CLI Command Reference*.

describe-risk-configuration

The following code example shows how to use `describe-risk-configuration`.

AWS CLI

To describe a risk configuration

This example describes the risk configuration associated with pool `us-west-2_aaaaaaaaa`.

Command:

```
aws cognito-idp describe-risk-configuration --user-pool-id us-west-2_aaaaaaaaa
```

Output:

```
{
  "RiskConfiguration": {
    "UserPoolId": "us-west-2_aaaaaaaaa",
    "CompromisedCredentialsRiskConfiguration": {
      "EventFilter": [
        "SIGN_IN",
        "SIGN_UP",
        "PASSWORD_CHANGE"
      ],
      "Actions": {
        "EventAction": "BLOCK"
      }
    },
    "AccountTakeoverRiskConfiguration": {
      "NotifyConfiguration": {
        "From": "diego@example.com",
        "ReplyTo": "diego@example.com",
        "SourceArn": "arn:aws:ses:us-east-1:111111111111:identity/diego@example.com",
        "BlockEmail": {
          "Subject": "Blocked sign-in attempt",
          "HtmlBody": "<!DOCTYPE html>\n<html>\n<head>\n\t<title>HTML email context</title>\n\t<meta charset=\"utf-8\">\n</head>\n<body>\n<pre>We blocked an unrecognized sign-in to your account with this information:\n<ul>\n<li>Time: {login-time}</li>\n<li>Device: {device-name}</li>\n<li>Location: {city}, {country}</li>\n</ul>\nIf this sign-in was not by you, you should change your password and notify us by clicking on <a href={one-click-link-invalid}>this link</a>\nIf this sign-in was by you, you can follow <a href={one-click-link-valid}>this link</a> to let us know</pre>\n</body>\n</html>",
          "TextBody": "We blocked an unrecognized sign-in to your account with this information:\nTime: {login-time}\nDevice: {device-name}\nLocation: {city}, {country}\nIf this sign-in was not by you, you should change your password and notify us by clicking on {one-click-link-invalid}\nIf this sign-in was by you, you can follow {one-click-link-valid} to let us know"
        },
        "NoActionEmail": {
          "Subject": "New sign-in attempt",
          "HtmlBody": "<!DOCTYPE html>\n<html>\n<head>\n\t<title>HTML email context</title>\n\t<meta charset=\"utf-8\">\n</head>\n<body>\n<pre>We observed an unrecognized sign-in to your account with this information:\n<ul>\n<li>Time: {login-time}</li>\n<li>Device: {device-name}</li>\n<li>Location: {city}, {country}</li>\n</ul>\nIf this sign-in was not by you, you should change your"
        }
      }
    }
  }
}
```


- For API details, see [DescribeRiskConfiguration](#) in *AWS CLI Command Reference*.

describe-user-import-job

The following code example shows how to use `describe-user-import-job`.

AWS CLI

To describe a user import job

This example describes a user input job.

For more information about importing users, see [Importing Users into User Pools From a CSV File](#).

Command:

```
aws cognito-idp describe-user-import-job --user-pool-id us-west-2_aaaaaaaaa --job-id
import-TZqNQvDRnW
```

Output:

```
{
  "UserImportJob": {
    "JobName": "import-Test1",
    "JobId": "import-TZqNQvDRnW",
    "UserPoolId": "us-west-2_aaaaaaaaa",
    "PreSignedUrl": "PRE_SIGNED URL",
    "CreationDate": 1548271708.512,
    "Status": "Created",
    "CloudWatchLogsRoleArn": "arn:aws:iam::111111111111:role/
CognitoCloudWatchLogsRole",
    "ImportedUsers": 0,
    "SkippedUsers": 0,
    "FailedUsers": 0
  }
}
```

- For API details, see [DescribeUserImportJob](#) in *AWS CLI Command Reference*.

describe-user-pool-client

The following code example shows how to use `describe-user-pool-client`.

AWS CLI

To describe a user pool client

This example describes a user pool client.

Command:

```
aws cognito-idp describe-user-pool-client --user-pool-id us-west-2_aaaaaaaaa --client-id 38fjsnc484p94kpbsnet7mpld0
```

Output:

```
{
  "UserPoolClient": {
    "UserPoolId": "us-west-2_aaaaaaaaa",
    "ClientName": "MyApp",
    "ClientId": "38fjsnc484p94kpbsnet7mpld0",
    "ClientSecret": "CLIENT_SECRET",
    "LastModifiedDate": 1548108676.163,
    "CreationDate": 1548108676.163,
    "RefreshTokenValidity": 30,
    "ReadAttributes": [
      "address",
      "birthdate",
      "custom:CustomAttr1",
      "custom:CustomAttr2",
      "email",
      "email_verified",
      "family_name",
      "gender",
      "given_name",
      "locale",
      "middle_name",
      "name",
      "nickname",
      "phone_number",
      "phone_number_verified",
      "picture",
      "preferred_username",
```

```
        "profile",
        "updated_at",
        "website",
        "zoneinfo"
    ],
    "WriteAttributes": [
        "address",
        "birthdate",
        "custom:CustomAttr1",
        "custom:CustomAttr2",
        "email",
        "family_name",
        "gender",
        "given_name",
        "locale",
        "middle_name",
        "name",
        "nickname",
        "phone_number",
        "picture",
        "preferred_username",
        "profile",
        "updated_at",
        "website",
        "zoneinfo"
    ],
    "ExplicitAuthFlows": [
        "ADMIN_NO_SRP_AUTH",
        "USER_PASSWORD_AUTH"
    ],
    "AllowedOAuthFlowsUserPoolClient": false
}
}
```

- For API details, see [DescribeUserPoolClient](#) in *AWS CLI Command Reference*.

describe-user-pool-domain

The following code example shows how to use `describe-user-pool-domain`.

AWS CLI

To describe a user pool client

This example describes a user pool domain named my-domain.

Command:

```
aws cognito-idp describe-user-pool-domain --domain my-domain
```

Output:

```
{
  "DomainDescription": {
    "UserPoolId": "us-west-2_aaaaaaaaa",
    "AWSAccountId": "111111111111",
    "Domain": "my-domain",
    "S3Bucket": "aws-cognito-prod-pdx-assets",
    "CloudFrontDistribution": "aaaaaaaaaaaaa.cloudfront.net",
    "Version": "20190128175402",
    "Status": "ACTIVE",
    "CustomDomainConfig": {}
  }
}
```

- For API details, see [DescribeUserPoolDomain](#) in *AWS CLI Command Reference*.

describe-user-pool

The following code example shows how to use describe-user-pool.

AWS CLI

To describe a user pool

This example describes a user pool with the user pool id us-west-2_aaaaaaaaa.

Command:

```
aws cognito-idp describe-user-pool --user-pool-id us-west-2_aaaaaaaaa
```

Output:

```
{
  "UserPool": {
    "SmsVerificationMessage": "Your verification code is {####}. ",
  }
}
```

```
"SchemaAttributes": [  
  {  
    "Name": "sub",  
    "StringAttributeConstraints": {  
      "MinLength": "1",  
      "MaxLength": "2048"  
    },  
    "DeveloperOnlyAttribute": false,  
    "Required": true,  
    "AttributeDataType": "String",  
    "Mutable": false  
  },  
  {  
    "Name": "name",  
    "StringAttributeConstraints": {  
      "MinLength": "0",  
      "MaxLength": "2048"  
    },  
    "DeveloperOnlyAttribute": false,  
    "Required": false,  
    "AttributeDataType": "String",  
    "Mutable": true  
  },  
  {  
    "Name": "given_name",  
    "StringAttributeConstraints": {  
      "MinLength": "0",  
      "MaxLength": "2048"  
    },  
    "DeveloperOnlyAttribute": false,  
    "Required": false,  
    "AttributeDataType": "String",  
    "Mutable": true  
  },  
  {  
    "Name": "family_name",  
    "StringAttributeConstraints": {  
      "MinLength": "0",  
      "MaxLength": "2048"  
    },  
    "DeveloperOnlyAttribute": false,  
    "Required": false,  
    "AttributeDataType": "String",  
    "Mutable": true  
  }  
]
```

```
  },
  {
    "Name": "middle_name",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "nickname",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "preferred_username",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "profile",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  }
}
```

```
  },
  {
    "Name": "picture",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "website",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "email",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": true,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "AttributeDataType": "Boolean",
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "Name": "email_verified",
    "Mutable": true
  },
  {
    "Name": "gender",
    "StringAttributeConstraints": {
```

```
        "MinLength": "0",
        "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
},
{
    "Name": "birthdate",
    "StringAttributeConstraints": {
        "MinLength": "10",
        "MaxLength": "10"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
},
{
    "Name": "zoneinfo",
    "StringAttributeConstraints": {
        "MinLength": "0",
        "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
},
{
    "Name": "locale",
    "StringAttributeConstraints": {
        "MinLength": "0",
        "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
},
{
    "Name": "phone_number",
    "StringAttributeConstraints": {
```

```
        "MinLength": "0",
        "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
},
{
    "AttributeDataType": "Boolean",
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "Name": "phone_number_verified",
    "Mutable": true
},
{
    "Name": "address",
    "StringAttributeConstraints": {
        "MinLength": "0",
        "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
},
{
    "Name": "updated_at",
    "NumberAttributeConstraints": {
        "MinValue": "0"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "Number",
    "Mutable": true
}
],
"EmailVerificationSubject": "Your verification code",
"MfaConfiguration": "OFF",
"Name": "MyUserPool",
"EmailVerificationMessage": "Your verification code is {#####}. ",
"SmsAuthenticationMessage": "Your authentication code is {#####}. ",
"LastModifiedDate": 1547763720.822,
"AdminCreateUserConfig": {
```



```

    "InviteMessageTemplate": {
      "EmailMessage": "Your username is {username} and temporary password is
{#####}. ",
      "EmailSubject": "Your temporary password",
      "SMSMessage": "Your username is {username} and temporary password is
{#####}. "
    },
    "UnusedAccountValidityDays": 7,
    "AllowAdminCreateUserOnly": false
  },
  "EmailConfiguration": {
    "ReplyToEmailAddress": "myemail@mydomain.com"
    "SourceArn": "arn:aws:ses:us-east-1:000000000000:identity/
myemail@mydomain.com"
  },
  "AutoVerifiedAttributes": [
    "email"
  ],
  "Policies": {
    "PasswordPolicy": {
      "RequireLowercase": true,
      "RequireSymbols": true,
      "RequireNumbers": true,
      "MinimumLength": 8,
      "RequireUppercase": true
    }
  },
  "UserPoolTags": {},
  "UsernameAttributes": [
    "email"
  ],
  "CreationDate": 1547763720.822,
  "EstimatedNumberOfUsers": 1,
  "Id": "us-west-2_aaaaaaaaa",
  "LambdaConfig": {}
}
}

```

- For API details, see [DescribeUserPool](#) in *AWS CLI Command Reference*.

forget-device

The following code example shows how to use forget-device.

AWS CLI

To forget a device

This example forgets device a device.

Command:

```
aws cognito-idp forget-device --device-key us-west-2_abcd_1234-5678
```

- For API details, see [ForgetDevice](#) in *AWS CLI Command Reference*.

forgot-password

The following code example shows how to use forgot-password.

AWS CLI

To force a password change

The following forgot-password example sends a message to jane@example.com to change their password.

```
aws cognito-idp forgot-password --client-id 38fjsnc484p94kpqsnet7mpld0 --username jane@example.com
```

Output:

```
{
  "CodeDeliveryDetails": {
    "Destination": "j***@e***.com",
    "DeliveryMedium": "EMAIL",
    "AttributeName": "email"
  }
}
```

- For API details, see [ForgotPassword](#) in *AWS CLI Command Reference*.

get-csv-header

The following code example shows how to use get-csv-header.

AWS CLI

To create a csv header

This example creates a csv header.

For more information about importing users, see [Importing Users into User Pools From a CSV File](#).

Command:

```
aws cognito-idp get-csv-header --user-pool-id us-west-2_aaaaaaaaa
```

Output:

```
{
  "UserPoolId": "us-west-2_aaaaaaaaa",
  "CSVHeader": [
    "name",
    "given_name",
    "family_name",
    "middle_name",
    "nickname",
    "preferred_username",
    "profile",
    "picture",
    "website",
    "email",
    "email_verified",
    "gender",
    "birthdate",
    "zoneinfo",
    "locale",
    "phone_number",
    "phone_number_verified",
    "address",
    "updated_at",
    "cognito:mfa_enabled",
    "cognito:username"
  ]
}
```

... Importing Users into User Pools From a CSV File: <https://docs.aws.amazon.com/cognito/latest/developerguide/cognito-user-pools-using-import-tool.html>

- For API details, see [GetCsvHeader](#) in *AWS CLI Command Reference*.

get-group

The following code example shows how to use `get-group`.

AWS CLI

To get information about a group

This example gets information about a group named `MyGroup`.

Command:

```
aws cognito-idp get-group --user-pool-id us-west-2_aaaaaaaaa --group-name MyGroup
```

Output:

```
{
  "Group": {
    "GroupName": "MyGroup",
    "UserPoolId": "us-west-2_aaaaaaaaa",
    "Description": "A sample group.",
    "LastModifiedDate": 1548270073.795,
    "CreationDate": 1548270073.795
  }
}
```

- For API details, see [GetGroup](#) in *AWS CLI Command Reference*.

get-signing-certificate

The following code example shows how to use `get-signing-certificate`.

AWS CLI

To get a signing certificate

This example gets a signing certificate for a user pool.

Command:

```
aws cognito-idp get-signing-certificate --user-pool-id us-west-2_aaaaaaaaa
```

Output:

```
{
  "Certificate": "CERTIFICATE_DATA"
}
```

- For API details, see [GetSigningCertificate](#) in *AWS CLI Command Reference*.

get-ui-customization

The following code example shows how to use `get-ui-customization`.

AWS CLI

To get UI customization information

This example gets UI customization information for a user pool.

Command:

```
aws cognito-idp get-ui-customization --user-pool-id us-west-2_aaaaaaaaa
```

Output:

```
{
  "UICustomization": {
    "UserPoolId": "us-west-2_aaaaaaaaa",
    "ClientId": "ALL",
    "ImageUrl": "https://aaaaaaaaaaaaa.cloudfront.net/us-west-2_aaaaaaaaa/ALL/20190128231240/assets/images/image.jpg",
    "CSS": ".logo-customizable {\n\tmax-width: 60%;\n\tmax-height: 30%;\n}\n\n.banner-customizable {\n\tpadding: 25px 0px 25px 10px;\n\tbackground-color: lightgray;\n}\n\n.label-customizable {\n\tfont-weight: 300;\n}\n\n.textDescription-customizable {\n\tpadding-top: 10px;\n\tpadding-bottom: 10px;\n\tdisplay: block;
```

```

\n\tfont-size: 16px;\n}\n.idpDescription-customizable {\n\tpadding-top: 10px;\n\tpadding-bottom: 10px;\n\tdisplay: block;\n\tfont-size: 16px;\n}\n.legalText-customizable {\n\tcolor: #747474;\n\tfont-size: 11px;\n}\n.submitButton-customizable {\n\tfont-size: 14px;\n\tfont-weight: bold;\n\tmargin: 20px 0px 10px 0px;\n\theight: 40px;\n\twidth: 100%;\n\tcolor: #fff;\n\tbackground-color: #337ab7;\n}\n.submitButton-customizable:hover {\n\tcolor: #fff;\n\tbackground-color: #286090;\n}\n.errorMessage-customizable {\n\tpadding: 5px;\n\tfont-size: 14px;\n\twidth: 100%;\n\tbackground: #F5F5F5;\n\tborder: 2px solid #D64958;\n\tcolor: #D64958;\n}\n.inputField-customizable {\n\twidth: 100%;\n\theight: 34px;\n\tcolor: #555;\n\tbackground-color: #fff;\n\tborder: 1px solid #ccc;\n}\n.inputField-customizable:focus {\n\tborder-color: #66afe9;\n\toutline: 0;\n}\n.idpButton-customizable {\n\theight: 40px;\n\twidth: 100%;\n\ttext-align: center;\n\tmargin-bottom: 15px;\n\tcolor: #fff;\n\tbackground-color: #5bc0de;\n\tborder-color: #46b8da;\n}\n.idpButton-customizable:hover {\n\tcolor: #fff;\n\tbackground-color: #31b0d5;\n}\n.socialButton-customizable {\n\theight: 40px;\n\ttext-align: left;\n\twidth: 100%;\n\tmargin-bottom: 15px;\n}\n.redirect-customizable {\n\ttext-align: center;\n}\n.passwordCheck-notValid-customizable {\n\tcolor: #DF3312;\n}\n.passwordCheck-valid-customizable {\n\tcolor: #19BF00;\n}\n.background-customizable {\n\tbackground-color: #faf;\n}\n",
    "CSSVersion": "20190128231240"
  }
}

```

- For API details, see [GetUiCustomization](#) in *AWS CLI Command Reference*.

list-user-import-jobs

The following code example shows how to use `list-user-import-jobs`.

AWS CLI

To list user import jobs

This example lists user import jobs.

For more information about importing users, see [Importing Users into User Pools From a CSV File](#).

Command:

```
aws cognito-idp list-user-import-jobs --user-pool-id us-west-2_aaaaaaaaa --max-results 20
```

Output:

```
{
  "UserImportJobs": [
    {
      "JobName": "Test2",
      "JobId": "import-d00nwGA3mV",
      "UserPoolId": "us-west-2_aaaaaaaaaa",
      "PreSignedUrl": "PRE_SIGNED_URL",
      "CreationDate": 1548272793.069,
      "Status": "Created",
      "CloudWatchLogsRoleArn": "arn:aws:iam::111111111111:role/
CognitoCloudWatchLogsRole",
      "ImportedUsers": 0,
      "SkippedUsers": 0,
      "FailedUsers": 0
    },
    {
      "JobName": "Test1",
      "JobId": "import-qQ0DCt2fRh",
      "UserPoolId": "us-west-2_aaaaaaaaaa",
      "PreSignedUrl": "PRE_SIGNED_URL",
      "CreationDate": 1548271795.471,
      "Status": "Created",
      "CloudWatchLogsRoleArn": "arn:aws:iam::111111111111:role/
CognitoCloudWatchLogsRole",
      "ImportedUsers": 0,
      "SkippedUsers": 0,
      "FailedUsers": 0
    },
    {
      "JobName": "import-Test1",
      "JobId": "import-TZqNQvDRnW",
      "UserPoolId": "us-west-2_aaaaaaaaaa",
      "PreSignedUrl": "PRE_SIGNED_URL",
      "CreationDate": 1548271708.512,
      "StartDate": 1548277247.962,
      "CompletionDate": 1548277248.912,
      "Status": "Failed",
      "CloudWatchLogsRoleArn": "arn:aws:iam::111111111111:role/
CognitoCloudWatchLogsRole",
      "ImportedUsers": 0,
      "SkippedUsers": 0,
      "FailedUsers": 1,
    }
  ]
}
```

```
        "CompletionMessage": "Too many users have failed or been skipped during
the import."
    }
]
}
```

- For API details, see [ListUserImportJobs](#) in *AWS CLI Command Reference*.

list-user-pools

The following code example shows how to use `list-user-pools`.

AWS CLI

To list user pools

This example lists up to 20 user pools.

Command:

```
aws cognito-idp list-user-pools --max-results 20
```

Output:

```
{
  "UserPools": [
    {
      "CreationDate": 1547763720.822,
      "LastModifiedDate": 1547763720.822,
      "LambdaConfig": {},
      "Id": "us-west-2_aaaaaaaaaa",
      "Name": "MyUserPool"
    }
  ]
}
```

- For API details, see [ListUserPools](#) in *AWS CLI Command Reference*.

list-users-in-group

The following code example shows how to use `list-users-in-group`.

AWS CLI

To list users in a group

This example lists users in group MyGroup.

Command:

```
aws cognito-idp list-users-in-group --user-pool-id us-west-2_aaaaaaaaa --group-name
MyGroup
```

Output:

```
{
  "Users": [
    {
      "Username": "acf10624-80bb-401a-ac61-607bee2110ec",
      "Attributes": [
        {
          "Name": "sub",
          "Value": "acf10624-80bb-401a-ac61-607bee2110ec"
        },
        {
          "Name": "custom:CustomAttr1",
          "Value": "New Value!"
        },
        {
          "Name": "email",
          "Value": "jane@example.com"
        }
      ],
      "UserCreateDate": 1548102770.284,
      "UserLastModifiedDate": 1548103204.893,
      "Enabled": true,
      "UserStatus": "CONFIRMED"
    },
    {
      "Username": "22704aa3-fc10-479a-97eb-2af5806bd327",
      "Attributes": [
        {
          "Name": "sub",
          "Value": "22704aa3-fc10-479a-97eb-2af5806bd327"
        }
      ],
    }
  ]
}
```

```
    {
      "Name": "email_verified",
      "Value": "true"
    },
    {
      "Name": "email",
      "Value": "diego@example.com"
    }
  ],
  "UserCreateDate": 1548089817.683,
  "UserLastModifiedDate": 1548089817.683,
  "Enabled": true,
  "UserStatus": "FORCE_CHANGE_PASSWORD"
}
]
```

- For API details, see [ListUsersInGroup](#) in *AWS CLI Command Reference*.

list-users

The following code example shows how to use `list-users`.

AWS CLI

To list users

This example lists up to 20 users.

Command:

```
aws cognito-idp list-users --user-pool-id us-west-2_aaaaaaaaa --limit 20
```

Output:

```
{
  "Users": [
    {
      "Username": "22704aa3-fc10-479a-97eb-2af5806bd327",
      "Enabled": true,
      "UserStatus": "FORCE_CHANGE_PASSWORD",
      "UserCreateDate": 1548089817.683,
```

```
"UserLastModifiedDate": 1548089817.683,
"Attributes": [
  {
    "Name": "sub",
    "Value": "22704aa3-fc10-479a-97eb-2af5806bd327"
  },
  {
    "Name": "email_verified",
    "Value": "true"
  },
  {
    "Name": "email",
    "Value": "mary@example.com"
  }
]
}
```

- For API details, see [ListUsers](#) in *AWS CLI Command Reference*.

resend-confirmation-code

The following code example shows how to use `resend-confirmation-code`.

AWS CLI

To resend a confirmation code

The following `resend-confirmation-code` example sends a confirmation code to the user `jane`.

```
aws cognito-idp resend-confirmation-code \
  --client-id 12a3b456c7de890f11g123hijk \
  --username jane
```

Output:

```
{
  "CodeDeliveryDetails": {
    "Destination": "j***@e***.com",
    "DeliveryMedium": "EMAIL",
```

```
    "AttributeName": "email"
  }
}
```

For more information, see [Signing up and confirming user accounts](#) in the *Amazon Cognito Developer Guide*.

- For API details, see [ResendConfirmationCode](#) in *AWS CLI Command Reference*.

respond-to-auth-challenge

The following code example shows how to use `respond-to-auth-challenge`.

AWS CLI

To respond to an authorization challenge

This example responds to an authorization challenge initiated with `initiate-auth`. It is a response to the `NEW_PASSWORD_REQUIRED` challenge. It sets a password for user `jane@example.com`.

Command:

```
aws cognito-idp respond-to-auth-challenge --client-id 3n4b5urk1ft4f13mg5e62d9ado
--challenge-name NEW_PASSWORD_REQUIRED --challenge-responses
USERNAME=jane@example.com,NEW_PASSWORD="password" --session "SESSION_TOKEN"
```

Output:

```
{
  "ChallengeParameters": {},
  "AuthenticationResult": {
    "AccessToken": "ACCESS_TOKEN",
    "ExpiresIn": 3600,
    "TokenType": "Bearer",
    "RefreshToken": "REFRESH_TOKEN",
    "IdToken": "ID_TOKEN",
    "NewDeviceMetadata": {
      "DeviceKey": "us-west-2_fec070d2-fa88-424a-8ec8-b26d7198eb23",
      "DeviceGroupKey": "-wt2ha1Zd"
    }
  }
}
```

- For API details, see [RespondToAuthChallenge](#) in *AWS CLI Command Reference*.

set-risk-configuration

The following code example shows how to use `set-risk-configuration`.

AWS CLI

To set risk configuration

This example sets the risk configuration for a user pool. It sets the sign-up event action to `NO_ACTION`.

Command:

```
aws cognito-idp set-risk-configuration --user-pool-id us-west-2_aaaaaaaaaa --compromised-credentials-risk-configuration EventFilter=SIGN_UP,Actions={EventAction=NO_ACTION}
```

Output:

```
{
  "RiskConfiguration": {
    "UserPoolId": "us-west-2_aaaaaaaaaa",
    "CompromisedCredentialsRiskConfiguration": {
      "EventFilter": [
        "SIGN_UP"
      ],
      "Actions": {
        "EventAction": "NO_ACTION"
      }
    }
  }
}
```

- For API details, see [SetRiskConfiguration](#) in *AWS CLI Command Reference*.

set-ui-customization

The following code example shows how to use `set-ui-customization`.

AWS CLI

To set UI customization

This example customizes the CSS setting for a user pool.

Command:

```
aws cognito-idp set-ui-customization --user-pool-id us-west-2_aaaaaaaaa --css
".logo-customizable {\n\tmax-width: 60%;\n\tmax-height: 30%;\n}\n.banner-
customizable {\n\tpadding: 25px 0px 25px 10px;\n\tbackground-color: lightgray;
\n}\n.label-customizable {\n\tfont-weight: 300;\n}\n.textDescription-customizable
{\n\tpadding-top: 10px;\n\tpadding-bottom: 10px;\n\tdisplay: block;\n\tfont-
size: 16px;\n}\n.idpDescription-customizable {\n\tpadding-top: 10px;\n\tpadding-
bottom: 10px;\n\tdisplay: block;\n\tfont-size: 16px;\n}\n.legalText-customizable
{\n\tcolor: #747474;\n\tfont-size: 11px;\n}\n.submitButton-customizable
{\n\tfont-size: 14px;\n\tfont-weight: bold;\n\tmargin: 20px 0px 10px 0px;\n
\theight: 40px;\n\twidth: 100%;\n\tcolor: #fff;\n\tbackground-color: #337ab7;
\n}\n.submitButton-customizable:hover {\n\tcolor: #fff;\n\tbackground-color:
#286090;\n}\n.errorMessage-customizable {\n\tpadding: 5px;\n\tfont-size: 14px;
\n\twidth: 100%;\n\tbackground: #F5F5F5;\n\tborder: 2px solid #D64958;\n\tcolor:
#D64958;\n}\n.inputField-customizable {\n\twidth: 100%;\n\theight: 34px;\n\tcolor:
#555;\n\tbackground-color: #fff;\n\tborder: 1px solid #ccc;\n}\n.inputField-
customizable:focus {\n\tborder-color: #66afe9;\n\toutline: 0;\n}\n.idpButton-
customizable {\n\theight: 40px;\n\twidth: 100%;\n\ttext-align: center;\n\tmargin-
bottom: 15px;\n\tcolor: #fff;\n\tbackground-color: #5bc0de;\n\tborder-color:
#46b8da;\n}\n.idpButton-customizable:hover {\n\tcolor: #fff;\n\tbackground-color:
#31b0d5;\n}\n.socialButton-customizable {\n\theight: 40px;\n\ttext-align: left;
\n\twidth: 100%;\n\tmargin-bottom: 15px;\n}\n.redirect-customizable {\n\ttext-
align: center;\n}\n.passwordCheck-notValid-customizable {\n\tcolor: #DF3312;
\n}\n.passwordCheck-valid-customizable {\n\tcolor: #19BF00;\n}\n.background-
customizable {\n\tbackground-color: #faf;\n}\n"
```

Output:

```
{
  "UICustomization": {
    "UserPoolId": "us-west-2_aaaaaaaaa",
    "ClientId": "ALL",
    "CSS": ".logo-customizable {\n\tmax-width: 60%;\n\tmax-height: 30%;
\n}\n.banner-customizable {\n\tpadding: 25px 0px 25px 10px;\n\tbackground-color:
lightgray;\n}\n.label-customizable {\n\tfont-weight: 300;\n}\n.textDescription-
customizable {\n\tpadding-top: 10px;\n\tpadding-bottom: 10px;\n\tdisplay: block;
```

```

\n\tfont-size: 16px;\n}\n.idpDescription-customizable {\n\tpadding-top: 10px;\n\tpadding-bottom: 10px;\n\tdisplay: block;\n\tfont-size: 16px;\n}\n.legalText-customizable {\n\tcolor: #747474;\n\tfont-size: 11px;\n}\n.submitButton-customizable {\n\tfont-size: 14px;\n\tfont-weight: bold;\n\tmargin: 20px 0px 10px 0px;\n\theight: 40px;\n\twidth: 100%;\n\tcolor: #fff;\n\tbackground-color: #337ab7;\n}\n.submitButton-customizable:hover {\n\tcolor: #fff;\n\tbackground-color: #286090;\n}\n.errorMessage-customizable {\n\tpadding: 5px;\n\tfont-size: 14px;\n\twidth: 100%;\n\tbackground: #F5F5F5;\n\tborder: 2px solid #D64958;\n\tcolor: #D64958;\n}\n.inputField-customizable {\n\twidth: 100%;\n\theight: 34px;\n\tcolor: #555;\n\tbackground-color: #fff;\n\tborder: 1px solid #ccc;\n}\n.inputField-customizable:focus {\n\tborder-color: #66afe9;\n\toutline: 0;\n}\n.idpButton-customizable {\n\theight: 40px;\n\twidth: 100%;\n\ttext-align: center;\n\tmargin-bottom: 15px;\n\tcolor: #fff;\n\tbackground-color: #5bc0de;\n\tborder-color: #46b8da;\n}\n.idpButton-customizable:hover {\n\tcolor: #fff;\n\tbackground-color: #31b0d5;\n}\n.socialButton-customizable {\n\theight: 40px;\n\ttext-align: left;\n\twidth: 100%;\n\tmargin-bottom: 15px;\n}\n.redirect-customizable {\n\ttext-align: center;\n}\n.passwordCheck-notValid-customizable {\n\tcolor: #DF3312;\n}\n.passwordCheck-valid-customizable {\n\tcolor: #19BF00;\n}\n.background-customizable {\n\tbackground-color: #faf;\n}\n",
    "CSSVersion": "20190129172214"
  }
}

```

- For API details, see [SetUiCustomization](#) in *AWS CLI Command Reference*.

set-user-mfa-preference

The following code example shows how to use `set-user-mfa-preference`.

AWS CLI

To set user MFA settings

The following `set-user-mfa-preference` example modifies the MFA delivery options. It changes the MFA delivery medium to SMS.

```

aws cognito-idp set-user-mfa-preference \
  --access-token "eyJra12345EXAMPLE" \
  --software-token-mfa-settings Enabled=true,PreferredMfa=true \
  --sms-mfa-settings Enabled=false,PreferredMfa=false

```

This command produces no output.

For more information, see [Adding MFA to a user pool](#) in the *Amazon Cognito Developer Guide*.

- For API details, see [SetUserMfaPreference](#) in *AWS CLI Command Reference*.

set-user-settings

The following code example shows how to use `set-user-settings`.

AWS CLI

To set user settings

This example sets the MFA delivery preference to EMAIL.

Command:

```
aws cognito-idp set-user-settings --access-token ACCESS_TOKEN --mfa-options
DeliveryMedium=EMAIL
```

- For API details, see [SetUserSettings](#) in *AWS CLI Command Reference*.

sign-up

The following code example shows how to use `sign-up`.

AWS CLI

To sign up a user

This example signs up `jane@example.com`.

Command:

```
aws cognito-idp sign-up --client-id 3n4b5urk1ft4fl3mg5e62d9ado --
username jane@example.com --password PASSWORD --user-attributes
Name="email",Value="jane@example.com" Name="name",Value="Jane"
```

Output:

```
{
  "UserConfirmed": false,
```



```
"UserSub": "e04d60a6-45dc-441c-a40b-e25a787d4862"
}
```

- For API details, see [SignUp](#) in *AWS CLI Command Reference*.

start-user-import-job

The following code example shows how to use `start-user-import-job`.

AWS CLI

To start a user import job

This example starts a user input job.

For more information about importing users, see [Importing Users into User Pools From a CSV File](#).

Command:

```
aws cognito-idp start-user-import-job --user-pool-id us-west-2_aaaaaaaaa --job-id
import-TZqNQvDRnW
```

Output:

```
{
  "UserImportJob": {
    "JobName": "import-Test10",
    "JobId": "import-lmpxS0uIzH",
    "UserPoolId": "us-west-2_aaaaaaaaa",
    "PreSignedUrl": "PRE_SIGNED_URL",
    "CreationDate": 1548278378.928,
    "StartDate": 1548278397.334,
    "Status": "Pending",
    "CloudWatchLogsRoleArn": "arn:aws:iam::111111111111:role/
CognitoCloudWatchLogsRole",
    "ImportedUsers": 0,
    "SkippedUsers": 0,
    "FailedUsers": 0
  }
}
```

- For API details, see [StartUserImportJob](#) in *AWS CLI Command Reference*.

stop-user-import-job

The following code example shows how to use `stop-user-import-job`.

AWS CLI

To stop a user import job

This example stops a user input job.

For more information about importing users, see [Importing Users into User Pools From a CSV File](#).

Command:

```
aws cognito-idp stop-user-import-job --user-pool-id us-west-2_aaaaaaaaa --job-id
import-TZqNQvDRnW
```

Output:

```
{
  "UserImportJob": {
    "JobName": "import-Test5",
    "JobId": "import-Fx0kARISFL",
    "UserPoolId": "us-west-2_aaaaaaaaa",
    "PreSignedUrl": "PRE_SIGNED_URL",
    "CreationDate": 1548278576.259,
    "StartDate": 1548278623.366,
    "CompletionDate": 1548278626.741,
    "Status": "Stopped",
    "CloudWatchLogsRoleArn": "arn:aws:iam::111111111111:role/
CognitoCloudWatchLogsRole",
    "ImportedUsers": 0,
    "SkippedUsers": 0,
    "FailedUsers": 0,
    "CompletionMessage": "The Import Job was stopped by the developer."
  }
}
```

- For API details, see [StopUserImportJob](#) in *AWS CLI Command Reference*.

update-auth-event-feedback

The following code example shows how to use `update-auth-event-feedback`.

AWS CLI

To update auth event feedback

This example updates authorization event feedback. It marks the event "Valid".

Command:

```
aws cognito-idp update-auth-event-feedback --user-pool-id us-west-2_aaaaaaaaa --username diego@example.com --event-id EVENT_ID --feedback-token FEEDBACK_TOKEN --feedback-value "Valid"
```

- For API details, see [UpdateAuthEventFeedback](#) in *AWS CLI Command Reference*.

update-device-status

The following code example shows how to use `update-device-status`.

AWS CLI

To update device status

This example updates the status for a device to "not_remembered".

Command:

```
aws cognito-idp update-device-status --access-token ACCESS_TOKEN --device-key DEVICE_KEY --device-remembered-status "not_remembered"
```

- For API details, see [UpdateDeviceStatus](#) in *AWS CLI Command Reference*.

update-group

The following code example shows how to use `update-group`.

AWS CLI

To update a group

This example updates the description and precedence for MyGroup.

Command:

```
aws cognito-idp update-group --user-pool-id us-west-2_aaaaaaaaa --group-name MyGroup
--description "New description" --precedence 2
```

Output:

```
{
  "Group": {
    "GroupName": "MyGroup",
    "UserPoolId": "us-west-2_aaaaaaaaa",
    "Description": "New description",
    "RoleArn": "arn:aws:iam::111111111111:role/MyRole",
    "Precedence": 2,
    "LastModifiedDate": 1548800862.812,
    "CreationDate": 1548097827.125
  }
}
```

- For API details, see [UpdateGroup](#) in *AWS CLI Command Reference*.

update-resource-server

The following code example shows how to use `update-resource-server`.

AWS CLI

To update a resource server

This example updates the the resource server Weather. It adds a new scope.

Command:

```
aws cognito-idp update-resource-server --user-pool-id us-west-2_aaaaaaaaa
--identifier weather.example.com --name Weather --scopes
ScopeName=NewScope,ScopeDescription="New scope description"
```

Output:

```
{
```

```
"ResourceServer": {
  "UserPoolId": "us-west-2_aaaaaaaaa",
  "Identifier": "weather.example.com",
  "Name": "Happy",
  "Scopes": [
    {
      "ScopeName": "NewScope",
      "ScopeDescription": "New scope description"
    }
  ]
}
```

- For API details, see [UpdateResourceServer](#) in *AWS CLI Command Reference*.

update-user-attributes

The following code example shows how to use `update-user-attributes`.

AWS CLI

To update user attributes

This example updates the user attribute "nickname".

Command:

```
aws cognito-idp update-user-attributes --access-token ACCESS_TOKEN --user-attributes
Name="nickname",Value="Dan"
```

- For API details, see [UpdateUserAttributes](#) in *AWS CLI Command Reference*.

update-user-pool-client

The following code example shows how to use `update-user-pool-client`.

AWS CLI

To update a user pool client

This example updates the name of a user pool client. It also adds a writeable attribute "nickname".

Command:

```
aws cognito-idp update-user-pool-client --user-pool-id us-west-2_aaaaaaaa --client-id 3n4b5urk1ft4fl3mg5e62d9ado --client-name "NewClientName" --write-attributes "nickname"
```

Output:

```
{
  "UserPoolClient": {
    "UserPoolId": "us-west-2_aaaaaaaa",
    "ClientName": "NewClientName",
    "ClientId": "3n4b5urk1ft4fl3mg5e62d9ado",
    "LastModifiedDate": 1548802761.334,
    "CreationDate": 1548178931.258,
    "RefreshTokenValidity": 30,
    "WriteAttributes": [
      "nickname"
    ],
    "AllowedOAuthFlowsUserPoolClient": false
  }
}
```

- For API details, see [UpdateUserPoolClient](#) in *AWS CLI Command Reference*.

update-user-pool

The following code example shows how to use `update-user-pool`.

AWS CLI**To update a user pool**

This example adds tags to a user pool.

Command:

```
aws cognito-idp update-user-pool --user-pool-id us-west-2_aaaaaaaa --user-pool-tags Team=Blue,Area=West
```

- For API details, see [UpdateUserPool](#) in *AWS CLI Command Reference*.

Amazon Comprehend examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon Comprehend.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

batch-detect-dominant-language

The following code example shows how to use batch-detect-dominant-language.

AWS CLI

To detect the dominant language of multiple input texts

The following batch-detect-dominant-language example analyzes multiple input texts and returns the dominant language of each. The pre-trained models confidence score is also output for each prediction.

```
aws comprehend batch-detect-dominant-language \  
  --text-list "Physics is the natural science that involves the study of matter  
and its motion and behavior through space and time, along with related concepts  
such as energy and force."
```

Output:

```
{  
  "ResultList": [  
    {
```

```

        "Index": 0,
        "Languages": [
            {
                "LanguageCode": "en",
                "Score": 0.9986501932144165
            }
        ]
    },
    "ErrorList": []
}

```

For more information, see [Dominant Language](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [BatchDetectDominantLanguage](#) in *AWS CLI Command Reference*.

batch-detect-entities

The following code example shows how to use batch-detect-entities.

AWS CLI

To detect entities from multiple input texts

The following batch-detect-entities example analyzes multiple input texts and returns the named entities of each. The pre-trained model's confidence score is also output for each prediction.

```

aws comprehend batch-detect-entities \
  --language-code en \
  --text-list "Dear Jane, Your AnyCompany Financial Services LLC credit card
account 1111-XXXX-1111-XXXX has a minimum payment of $24.53 that is due by July
31st." "Please send customer feedback to Sunshine Spa, 123 Main St, Anywhere or to
Alice at AnySpa@example.com."

```

Output:

```

{
  "ResultList": [
    {
      "Index": 0,
      "Entities": [
        {

```



```
        "Score": 0.9985517859458923,
        "Type": "PERSON",
        "Text": "Jane",
        "BeginOffset": 5,
        "EndOffset": 9
    },
    {
        "Score": 0.9767839312553406,
        "Type": "ORGANIZATION",
        "Text": "AnyCompany Financial Services, LLC",
        "BeginOffset": 16,
        "EndOffset": 50
    },
    {
        "Score": 0.9856694936752319,
        "Type": "OTHER",
        "Text": "1111-XXXX-1111-XXXX",
        "BeginOffset": 71,
        "EndOffset": 90
    },
    {
        "Score": 0.9652159810066223,
        "Type": "QUANTITY",
        "Text": ".53",
        "BeginOffset": 116,
        "EndOffset": 119
    },
    {
        "Score": 0.9986667037010193,
        "Type": "DATE",
        "Text": "July 31st",
        "BeginOffset": 135,
        "EndOffset": 144
    }
]
},
{
    "Index": 1,
    "Entities": [
        {
            "Score": 0.720084547996521,
            "Type": "ORGANIZATION",
            "Text": "Sunshine Spa",
            "BeginOffset": 33,
```

```
        "EndOffset": 45
      },
      {
        "Score": 0.9865870475769043,
        "Type": "LOCATION",
        "Text": "123 Main St",
        "BeginOffset": 47,
        "EndOffset": 58
      },
      {
        "Score": 0.5895616412162781,
        "Type": "LOCATION",
        "Text": "Anywhere",
        "BeginOffset": 60,
        "EndOffset": 68
      },
      {
        "Score": 0.6809214353561401,
        "Type": "PERSON",
        "Text": "Alice",
        "BeginOffset": 75,
        "EndOffset": 80
      },
      {
        "Score": 0.9979087114334106,
        "Type": "OTHER",
        "Text": "AnySpa@example.com",
        "BeginOffset": 84,
        "EndOffset": 99
      }
    ]
  },
  "ErrorList": []
}
```

For more information, see [Entities](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [BatchDetectEntities](#) in *AWS CLI Command Reference*.

batch-detect-key-phrases

The following code example shows how to use batch-detect-key-phrases.

AWS CLI

To detect key phrases of multiple text inputs

The following `batch-detect-key-phrases` example analyzes multiple input texts and returns the key noun phrases of each. The pre-trained model's confidence score for each prediction is also output.

```
aws comprehend batch-detect-key-phrases \  
  --language-code en \  
  --text-list "Hello Zhang Wei, I am John, writing to you about the trip for next  
Saturday." "Dear Jane, Your AnyCompany Financial Services LLC credit card account  
1111-XXXX-1111-XXXX has a minimum payment of $24.53 that is due by July 31st."  
"Please send customer feedback to Sunshine Spa, 123 Main St, Anywhere or to Alice  
at AnySpa@example.com."
```

Output:

```
{  
  "ResultList": [  
    {  
      "Index": 0,  
      "KeyPhrases": [  
        {  
          "Score": 0.99700927734375,  
          "Text": "Zhang Wei",  
          "BeginOffset": 6,  
          "EndOffset": 15  
        },  
        {  
          "Score": 0.9929308891296387,  
          "Text": "John",  
          "BeginOffset": 22,  
          "EndOffset": 26  
        },  
        {  
          "Score": 0.9997230172157288,  
          "Text": "the trip",  
          "BeginOffset": 49,  
          "EndOffset": 57  
        },  
        {  
          "Score": 0.9999470114707947,  
          "Text": "Sunshine Spa",  
          "BeginOffset": 123,  
          "EndOffset": 141  
        }  
      ]  
    }  
  ]  
}
```

```
        "Text": "next Saturday",
        "BeginOffset": 62,
        "EndOffset": 75
      }
    ]
  },
  {
    "Index": 1,
    "KeyPhrases": [
      {
        "Score": 0.8358274102210999,
        "Text": "Dear Jane",
        "BeginOffset": 0,
        "EndOffset": 9
      },
      {
        "Score": 0.989359974861145,
        "Text": "Your AnyCompany Financial Services",
        "BeginOffset": 11,
        "EndOffset": 45
      },
      {
        "Score": 0.8812323808670044,
        "Text": "LLC credit card account 1111-XXXX-1111-XXXX",
        "BeginOffset": 47,
        "EndOffset": 90
      },
      {
        "Score": 0.9999381899833679,
        "Text": "a minimum payment",
        "BeginOffset": 95,
        "EndOffset": 112
      },
      {
        "Score": 0.9997439980506897,
        "Text": ".53",
        "BeginOffset": 116,
        "EndOffset": 119
      },
      {
        "Score": 0.996875524520874,
        "Text": "July 31st",
        "BeginOffset": 135,
        "EndOffset": 144
      }
    ]
  }
}
```

```
    }
  ]
},
{
  "Index": 2,
  "KeyPhrases": [
    {
      "Score": 0.9990295767784119,
      "Text": "customer feedback",
      "BeginOffset": 12,
      "EndOffset": 29
    },
    {
      "Score": 0.9994127750396729,
      "Text": "Sunshine Spa",
      "BeginOffset": 33,
      "EndOffset": 45
    },
    {
      "Score": 0.9892991185188293,
      "Text": "123 Main St",
      "BeginOffset": 47,
      "EndOffset": 58
    },
    {
      "Score": 0.9969810843467712,
      "Text": "Alice",
      "BeginOffset": 75,
      "EndOffset": 80
    },
    {
      "Score": 0.9703696370124817,
      "Text": "AnySpa@example.com",
      "BeginOffset": 84,
      "EndOffset": 99
    }
  ]
}
],
"ErrorList": []
}
```

For more information, see [Key Phrases](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [BatchDetectKeyPhrases](#) in *AWS CLI Command Reference*.

batch-detect-sentiment

The following code example shows how to use batch-detect-sentiment.

AWS CLI

To detect the prevailing sentiment of multiple input texts

The following batch-detect-sentiment example analyzes multiple input texts and returns the prevailing sentiment (POSITIVE, NEUTRAL, MIXED, or NEGATIVE, of each one).

```
aws comprehend batch-detect-sentiment \  
  --text-list "That movie was very boring, I can't believe it was over four hours long." "It is a beautiful day for hiking today." "My meal was okay, I'm excited to try other restaurants." \  
  --language-code en
```

Output:

```
{  
  "ResultList": [  
    {  
      "Index": 0,  
      "Sentiment": "NEGATIVE",  
      "SentimentScore": {  
        "Positive": 0.00011316669406369328,  
        "Negative": 0.9995445609092712,  
        "Neutral": 0.00014722718333359808,  
        "Mixed": 0.00019498742767609656  
      }  
    },  
    {  
      "Index": 1,  
      "Sentiment": "POSITIVE",  
      "SentimentScore": {  
        "Positive": 0.9981263279914856,  
        "Negative": 0.00015240783977787942,  
        "Neutral": 0.0013876151060685515,  
        "Mixed": 0.00033366199932061136  
      }  
    }  
  ],  
}
```

```

    {
      "Index": 2,
      "Sentiment": "MIXED",
      "SentimentScore": {
        "Positive": 0.15930435061454773,
        "Negative": 0.11471917480230331,
        "Neutral": 0.26897063851356506,
        "Mixed": 0.45700588822364807
      }
    }
  ],
  "ErrorList": []
}

```

For more information, see [Sentiment](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [BatchDetectSentiment](#) in *AWS CLI Command Reference*.

batch-detect-syntax

The following code example shows how to use `batch-detect-syntax`.

AWS CLI

To inspect the syntax and parts of speech of words in multiple input texts

The following `batch-detect-syntax` example analyzes the syntax of multiple input texts and returns the different parts of speech. The pre-trained model's confidence score is also output for each prediction.

```

aws comprehend batch-detect-syntax \
  --text-list "It is a beautiful day." "Can you please pass the salt?" "Please pay
the bill before the 31st." \
  --language-code en

```

Output:

```

{
  "ResultList": [
    {
      "Index": 0,
      "SyntaxTokens": [
        {

```

```
    "TokenId": 1,
    "Text": "It",
    "BeginOffset": 0,
    "EndOffset": 2,
    "PartOfSpeech": {
      "Tag": "PRON",
      "Score": 0.9999740719795227
    }
  },
  {
    "TokenId": 2,
    "Text": "is",
    "BeginOffset": 3,
    "EndOffset": 5,
    "PartOfSpeech": {
      "Tag": "VERB",
      "Score": 0.999937117099762
    }
  },
  {
    "TokenId": 3,
    "Text": "a",
    "BeginOffset": 6,
    "EndOffset": 7,
    "PartOfSpeech": {
      "Tag": "DET",
      "Score": 0.9999926686286926
    }
  },
  {
    "TokenId": 4,
    "Text": "beautiful",
    "BeginOffset": 8,
    "EndOffset": 17,
    "PartOfSpeech": {
      "Tag": "ADJ",
      "Score": 0.9987891912460327
    }
  },
  {
    "TokenId": 5,
    "Text": "day",
    "BeginOffset": 18,
    "EndOffset": 21,
```



```
        "PartOfSpeech": {
            "Tag": "NOUN",
            "Score": 0.9999778866767883
        }
    },
    {
        "TokenId": 6,
        "Text": ".",
        "BeginOffset": 21,
        "EndOffset": 22,
        "PartOfSpeech": {
            "Tag": "PUNCT",
            "Score": 0.9999974966049194
        }
    }
]
},
{
    "Index": 1,
    "SyntaxTokens": [
        {
            "TokenId": 1,
            "Text": "Can",
            "BeginOffset": 0,
            "EndOffset": 3,
            "PartOfSpeech": {
                "Tag": "AUX",
                "Score": 0.9999770522117615
            }
        },
        {
            "TokenId": 2,
            "Text": "you",
            "BeginOffset": 4,
            "EndOffset": 7,
            "PartOfSpeech": {
                "Tag": "PRON",
                "Score": 0.9999986886978149
            }
        },
        {
            "TokenId": 3,
            "Text": "please",
            "BeginOffset": 8,
```

```
        "EndOffset": 14,  
        "PartOfSpeech": {  
            "Tag": "INTJ",  
            "Score": 0.9681622385978699  
        }  
    },  
    {  
        "TokenId": 4,  
        "Text": "pass",  
        "BeginOffset": 15,  
        "EndOffset": 19,  
        "PartOfSpeech": {  
            "Tag": "VERB",  
            "Score": 0.9999874830245972  
        }  
    },  
    {  
        "TokenId": 5,  
        "Text": "the",  
        "BeginOffset": 20,  
        "EndOffset": 23,  
        "PartOfSpeech": {  
            "Tag": "DET",  
            "Score": 0.9999827146530151  
        }  
    },  
    {  
        "TokenId": 6,  
        "Text": "salt",  
        "BeginOffset": 24,  
        "EndOffset": 28,  
        "PartOfSpeech": {  
            "Tag": "NOUN",  
            "Score": 0.9995040893554688  
        }  
    },  
    {  
        "TokenId": 7,  
        "Text": "?",  
        "BeginOffset": 28,  
        "EndOffset": 29,  
        "PartOfSpeech": {  
            "Tag": "PUNCT",  
            "Score": 0.999998152256012  
        }  
    }  
}
```

```
    }
  }
]
},
{
  "Index": 2,
  "SyntaxTokens": [
    {
      "TokenId": 1,
      "Text": "Please",
      "BeginOffset": 0,
      "EndOffset": 6,
      "PartOfSpeech": {
        "Tag": "INTJ",
        "Score": 0.9997857809066772
      }
    },
    {
      "TokenId": 2,
      "Text": "pay",
      "BeginOffset": 7,
      "EndOffset": 10,
      "PartOfSpeech": {
        "Tag": "VERB",
        "Score": 0.9999252557754517
      }
    },
    {
      "TokenId": 3,
      "Text": "the",
      "BeginOffset": 11,
      "EndOffset": 14,
      "PartOfSpeech": {
        "Tag": "DET",
        "Score": 0.9999842643737793
      }
    },
    {
      "TokenId": 4,
      "Text": "bill",
      "BeginOffset": 15,
      "EndOffset": 19,
      "PartOfSpeech": {
        "Tag": "NOUN",
```

```
        "Score": 0.9999588131904602
      }
    },
    {
      "TokenId": 5,
      "Text": "before",
      "BeginOffset": 20,
      "EndOffset": 26,
      "PartOfSpeech": {
        "Tag": "ADP",
        "Score": 0.9958304762840271
      }
    },
    {
      "TokenId": 6,
      "Text": "the",
      "BeginOffset": 27,
      "EndOffset": 30,
      "PartOfSpeech": {
        "Tag": "DET",
        "Score": 0.9999947547912598
      }
    },
    {
      "TokenId": 7,
      "Text": "31st",
      "BeginOffset": 31,
      "EndOffset": 35,
      "PartOfSpeech": {
        "Tag": "NOUN",
        "Score": 0.9924124479293823
      }
    },
    {
      "TokenId": 8,
      "Text": ".",
      "BeginOffset": 35,
      "EndOffset": 36,
      "PartOfSpeech": {
        "Tag": "PUNCT",
        "Score": 0.9999955892562866
      }
    }
  ]
```

```

    }
  ],
  "ErrorList": []
}

```

For more information, see [Syntax Analysis](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [BatchDetectSyntax](#) in *AWS CLI Command Reference*.

batch-detect-targeted-sentiment

The following code example shows how to use `batch-detect-targeted-sentiment`.

AWS CLI

To detect the sentiment and each named entity for multiple input texts

The following `batch-detect-targeted-sentiment` example analyzes multiple input texts and returns the named entities along with the prevailing sentiment attached to each entity. The pre-trained model's confidence score is also output for each prediction.

```

aws comprehend batch-detect-targeted-sentiment \
  --language-code en \
  --text-list "That movie was really boring, the original was way more
entertaining" "The trail is extra beautiful today." "My meal was just okay."

```

Output:

```

{
  "ResultList": [
    {
      "Index": 0,
      "Entities": [
        {
          "DescriptiveMentionIndex": [
            0
          ],
          "Mentions": [
            {
              "Score": 0.9999009966850281,
              "GroupScore": 1.0,
              "Text": "movie",
              "Type": "MOVIE",

```

```

        "MentionSentiment": {
            "Sentiment": "NEGATIVE",
            "SentimentScore": {
                "Positive": 0.13887299597263336,
                "Negative": 0.8057460188865662,
                "Neutral": 0.05525200068950653,
                "Mixed": 0.00012799999967683107
            }
        },
        "BeginOffset": 5,
        "EndOffset": 10
    }
]
},
{
    "DescriptiveMentionIndex": [
        0
    ],
    "Mentions": [
        {
            "Score": 0.9921110272407532,
            "GroupScore": 1.0,
            "Text": "original",
            "Type": "MOVIE",
            "MentionSentiment": {
                "Sentiment": "POSITIVE",
                "SentimentScore": {
                    "Positive": 0.9999989867210388,
                    "Negative": 9.99999974752427e-07,
                    "Neutral": 0.0,
                    "Mixed": 0.0
                }
            }
        },
        "BeginOffset": 34,
        "EndOffset": 42
    ]
}
]
},
{
    "Index": 1,
    "Entities": [
        {

```

```
"DescriptiveMentionIndex": [
  0
],
"Mentions": [
  {
    "Score": 0.7545599937438965,
    "GroupScore": 1.0,
    "Text": "trail",
    "Type": "OTHER",
    "MentionSentiment": {
      "Sentiment": "POSITIVE",
      "SentimentScore": {
        "Positive": 1.0,
        "Negative": 0.0,
        "Neutral": 0.0,
        "Mixed": 0.0
      }
    },
    "BeginOffset": 4,
    "EndOffset": 9
  }
],
{
  "DescriptiveMentionIndex": [
    0
  ],
  "Mentions": [
    {
      "Score": 0.9999960064888,
      "GroupScore": 1.0,
      "Text": "today",
      "Type": "DATE",
      "MentionSentiment": {
        "Sentiment": "NEUTRAL",
        "SentimentScore": {
          "Positive": 9.000000318337698e-06,
          "Negative": 1.9999999949504854e-06,
          "Neutral": 0.9999859929084778,
          "Mixed": 3.999999989900971e-06
        }
      },
      "BeginOffset": 29,
      "EndOffset": 34
    }
  ]
}
```

```

    }
  ]
}
},
{
  "Index": 2,
  "Entities": [
    {
      "DescriptiveMentionIndex": [
        0
      ],
      "Mentions": [
        {
          "Score": 0.9999880194664001,
          "GroupScore": 1.0,
          "Text": "My",
          "Type": "PERSON",
          "MentionSentiment": {
            "Sentiment": "NEUTRAL",
            "SentimentScore": {
              "Positive": 0.0,
              "Negative": 0.0,
              "Neutral": 1.0,
              "Mixed": 0.0
            }
          }
        },
        {
          "BeginOffset": 0,
          "EndOffset": 2
        }
      ]
    }
  ],
  "DescriptiveMentionIndex": [
    0
  ],
  "Mentions": [
    {
      "Score": 0.9995260238647461,
      "GroupScore": 1.0,
      "Text": "meal",
      "Type": "OTHER",
      "MentionSentiment": {
        "Sentiment": "NEUTRAL",

```



```

        "SentimentScore": {
            "Positive": 0.04695599898695946,
            "Negative": 0.003226999891921878,
            "Neutral": 0.6091709733009338,
            "Mixed": 0.34064599871635437
        }
    },
    "BeginOffset": 3,
    "EndOffset": 7
}
]
}
],
"ErrorList": []
}

```

For more information, see [Targeted Sentiment](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [BatchDetectTargetedSentiment](#) in *AWS CLI Command Reference*.

classify-document

The following code example shows how to use `classify-document`.

AWS CLI

To classify document with model-specific endpoint

The following `classify-document` example classifies a document with an endpoint of a custom model. The model in this example was trained on a dataset containing sms messages labeled as spam or non-spam, or "ham".

```

aws comprehend classify-document \
  --endpoint-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier-
  endpoint/example-classifier-endpoint \
  --text "CONGRATULATIONS! TXT 1235550100 to win $5000"

```

Output:

```
{
```

```
"Classes": [  
  {  
    "Name": "spam",  
    "Score": 0.9998599290847778  
  },  
  {  
    "Name": "ham",  
    "Score": 0.00014001205272506922  
  }  
]
```

For more information, see [Custom Classification](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ClassifyDocument](#) in *AWS CLI Command Reference*.

contains-pii-entities

The following code example shows how to use contains-pii-entities.

AWS CLI

To analyze the input text for the presence of PII information

The following contains-pii-entities example analyzes the input text for the presence of personally identifiable information (PII) and returns the labels of identified PII entity types such as name, address, bank account number, or phone number.

```
aws comprehend contains-pii-entities \  
  --language-code en \  
  --text "Hello Zhang Wei, I am John. Your AnyCompany Financial Services, LLC  
credit card  
  account 1111-XXXX-1111-XXXX has a minimum payment of $24.53 that is due by  
July 31st. Based on your autopay settings,  
  we will withdraw your payment on the due date from your bank account number  
XXXXXX1111 with the routing number XXXXX0000.  
  Customer feedback for Sunshine Spa, 100 Main St, Anywhere. Send comments to  
Alice at AnySpa@example.com."
```

Output:

```
{
```

```
"Labels": [  
  {  
    "Name": "NAME",  
    "Score": 1.0  
  },  
  {  
    "Name": "EMAIL",  
    "Score": 1.0  
  },  
  {  
    "Name": "BANK_ACCOUNT_NUMBER",  
    "Score": 0.9995794296264648  
  },  
  {  
    "Name": "BANK_ROUTING",  
    "Score": 0.9173126816749573  
  },  
  {  
    "Name": "CREDIT_DEBIT_NUMBER",  
    "Score": 1.0  
  }  
]
```

For more information, see [Personally Identifiable Information \(PII\)](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ContainsPiiEntities](#) in *AWS CLI Command Reference*.

create-dataset

The following code example shows how to use create-dataset.

AWS CLI

To create a flywheel dataset

The following create-dataset example creates a dataset for a flywheel. This dataset will be used as additional training data as specified by the --dataset-type tag.

```
aws comprehend create-dataset \  
  --flywheel-arn arn:aws:comprehend:us-west-2:111122223333:flywheel/flywheel-  
entity \  
  --dataset-name example-dataset \  

```

```
--dataset-type "TRAIN" \  
--input-data-config file://inputConfig.json
```

Contents of file://inputConfig.json:

```
{  
  "DataFormat": "COMPREHEND_CSV",  
  "DocumentClassifierInputDataConfig": {  
    "S3Uri": "s3://DOC-EXAMPLE-BUCKET/training-data.csv"  
  }  
}
```

Output:

```
{  
  "DatasetArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/flywheel-  
entity/dataset/example-dataset"  
}
```

For more information, see [Flywheel Overview](#) in *Amazon Comprehend Developer Guide*.

- For API details, see [CreateDataset](#) in *AWS CLI Command Reference*.

create-document-classifier

The following code example shows how to use `create-document-classifier`.

AWS CLI

To create a document classifier to categorize documents

The following `create-document-classifier` example begins the training process for a document classifier model. The training data file, `training.csv`, is located at the `--input-data-config` tag. `training.csv` is a two column document where the labels, or, classifications are provided in the first column and the documents are provided in the second column.

```
aws comprehend create-document-classifier \  
  --document-classifier-name example-classifier \  
  --data-access-arn arn:aws:comprehend:us-west-2:111122223333:pii-entities-  
detection-job/123456abcdeb0e11022f22a11EXAMPLE \  
  --dataset-type "TRAIN" \  
  --input-data-config file://inputConfig.json
```

```
--input-data-config "S3Uri=s3://DOC-EXAMPLE-BUCKET/" \  
--language-code en
```

Output:

```
{  
  "DocumentClassifierArn": "arn:aws:comprehend:us-west-2:111122223333:document-  
classifier/example-classifier"  
}
```

For more information, see [Custom Classification](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [CreateDocumentClassifier](#) in *AWS CLI Command Reference*.

create-endpoint

The following code example shows how to use `create-endpoint`.

AWS CLI

To create an endpoint for a custom model

The following `create-endpoint` example creates an endpoint for synchronous inference for a previously trained custom model.

```
aws comprehend create-endpoint \  
  --endpoint-name example-classifier-endpoint-1 \  
  --model-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/  
example-classifier \  
  --desired-inference-units 1
```

Output:

```
{  
  "EndpointArn": "arn:aws:comprehend:us-west-2:111122223333:document-classifier-  
endpoint/example-classifier-endpoint-1"  
}
```

For more information, see [Managing Amazon Comprehend endpoints](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [CreateEndpoint](#) in *AWS CLI Command Reference*.

create-entity-recognizer

The following code example shows how to use `create-entity-recognizer`.

AWS CLI

To create a custom entity recognizer

The following `create-entity-recognizer` example begins the training process for a custom entity recognizer model. This example uses a CSV file containing training documents, `raw_text.csv`, and a CSV entity list, `entity_list.csv` to train the model. `entity_list.csv` contains the following columns: `text` and `type`.

```
aws comprehend create-entity-recognizer \  
  --recognizer-name example-entity-recognizer \  
  --data-access-role-arn arn:aws:iam::111122223333:role/service-role/  
AmazonComprehendServiceRole-example-role \  
  --input-data-config "EntityTypes=[{Type=DEVICE}],Documents={S3Uri=s3://DOC-  
EXAMPLE-BUCKET/trainingdata/raw_text.csv},EntityList={S3Uri=s3://DOC-EXAMPLE-BUCKET/  
trainingdata/entity_list.csv}" \  
  --language-code en
```

Output:

```
{  
  "EntityRecognizerArn": "arn:aws:comprehend:us-west-2:111122223333:example-  
entity-recognizer/entityrecognizer1"  
}
```

For more information, see [Custom entity recognition](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [CreateEntityRecognizer](#) in *AWS CLI Command Reference*.

create-flywheel

The following code example shows how to use `create-flywheel`.

AWS CLI

To create a flywheel

The following `create-flywheel` example creates a flywheel to orchestrate the ongoing training of either a document classification or entity recognition model. The flywheel in this example is created to manage an existing trained model specified by the `--active-model-arn` tag. When the flywheel is created, a data lake is created at the `--input-data-lake` tag.

```
aws comprehend create-flywheel \  
  --flywheel-name example-flywheel \  
  --active-model-arn arn:aws:comprehend:us-west-2:111122223333:document-  
classifier/example-model/version/1 \  
  --data-access-role-arn arn:aws:iam::111122223333:role/service-role/  
AmazonComprehendServiceRole-example-role \  
  --data-lake-s3-uri "s3://DOC-EXAMPLE-BUCKET"
```

Output:

```
{  
  "FlywheelArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/example-  
flywheel"  
}
```

For more information, see [Flywheel Overview](#) in *Amazon Comprehend Developer Guide*.

- For API details, see [CreateFlywheel](#) in *AWS CLI Command Reference*.

delete-document-classifier

The following code example shows how to use `delete-document-classifier`.

AWS CLI

To delete a custom document classifier

The following `delete-document-classifier` example deletes a custom document classifier model.

```
aws comprehend delete-document-classifier \  
  --document-classifier-arn arn:aws:comprehend:us-west-2:111122223333:document-  
classifier/example-classifier-1
```

This command produces no output.

For more information, see [Managing Amazon Comprehend endpoints](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DeleteDocumentClassifier](#) in *AWS CLI Command Reference*.

delete-endpoint

The following code example shows how to use `delete-endpoint`.

AWS CLI

To delete an endpoint for a custom model

The following `delete-endpoint` example deletes a model-specific endpoint. All endpoints must be deleted in order for the model to be deleted.

```
aws comprehend delete-endpoint \  
  --endpoint-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier-  
  endpoint/example-classifier-endpoint-1
```

This command produces no output.

For more information, see [Managing Amazon Comprehend endpoints](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DeleteEndpoint](#) in *AWS CLI Command Reference*.

delete-entity-recognizer

The following code example shows how to use `delete-entity-recognizer`.

AWS CLI

To delete a custom entity recognizer model

The following `delete-entity-recognizer` example deletes a custom entity recognizer model.

```
aws comprehend delete-entity-recognizer \  
  --entity-recognizer-arn arn:aws:comprehend:us-west-2:111122223333:entity-  
  recognizer/example-entity-recognizer-1
```


This command produces no output.

For more information, see [Managing Amazon Comprehend endpoints](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DeleteEntityRecognizer](#) in *AWS CLI Command Reference*.

delete-flywheel

The following code example shows how to use `delete-flywheel`.

AWS CLI

To delete a flywheel

The following `delete-flywheel` example deletes a flywheel. The data lake or the model associated with the flywheel is not deleted.

```
aws comprehend delete-flywheel \  
  --flywheel-arn arn:aws:comprehend:us-west-2:111122223333:flywheel/example-  
flywheel-1
```

This command produces no output.

For more information, see [Flywheel overview](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DeleteFlywheel](#) in *AWS CLI Command Reference*.

delete-resource-policy

The following code example shows how to use `delete-resource-policy`.

AWS CLI

To delete a resource-based policy

The following `delete-resource-policy` example deletes a resource-based policy from an Amazon Comprehend resource.

```
aws comprehend delete-resource-policy \  
  --resource-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/  
example-classifier-1/version/1
```

This command produces no output.

For more information, see [Copying custom models between AWS accounts](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DeleteResourcePolicy](#) in *AWS CLI Command Reference*.

describe-dataset

The following code example shows how to use describe-dataset.

AWS CLI

To describe a flywheel dataset

The following describe-dataset example gets the properties of a flywheel dataset.

```
aws comprehend describe-dataset \  
  --dataset-arn arn:aws:comprehend:us-west-2:111122223333:flywheel/flywheel-  
entity/dataset/example-dataset
```

Output:

```
{  
  "DatasetProperties": {  
    "DatasetArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/flywheel-  
entity/dataset/example-dataset",  
    "DatasetName": "example-dataset",  
    "DatasetType": "TRAIN",  
    "DatasetS3Uri": "s3://DOC-EXAMPLE-BUCKET/flywheel-entity/  
schemaVersion=1/12345678A123456Z/datasets/example-dataset/20230616T203710Z/",  
    "Status": "CREATING",  
    "CreationTime": "2023-06-16T20:37:10.400000+00:00"  
  }  
}
```

For more information, see [Flywheel Overview](#) in *Amazon Comprehend Developer Guide*.

- For API details, see [DescribeDataset](#) in *AWS CLI Command Reference*.

describe-document-classification-job

The following code example shows how to use describe-document-classification-job.

AWS CLI

To describe a document classification job

The following `describe-document-classification-job` example gets the properties of an asynchronous document classification job.

```
aws comprehend describe-document-classification-job \  
  --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

Output:

```
{  
  "DocumentClassificationJobProperties": {  
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE",  
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:document-  
classification-job/123456abcdeb0e11022f22a11EXAMPLE",  
    "JobName": "exampleclassificationjob",  
    "JobStatus": "COMPLETED",  
    "SubmitTime": "2023-06-14T17:09:51.788000+00:00",  
    "EndTime": "2023-06-14T17:15:58.582000+00:00",  
    "DocumentClassifierArn": "arn:aws:comprehend:us-  
west-2:111122223333:document-classifier/mymodel/version/1",  
    "InputDataConfig": {  
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/jobdata/",  
      "InputFormat": "ONE_DOC_PER_LINE"  
    },  
    "OutputDataConfig": {  
      "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/testfolder/111122223333-  
CLN-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"  
    },  
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/  
AmazonComprehendServiceRole-servicerole"  
  }  
}
```

For more information, see [Custom Classification](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DescribeDocumentClassificationJob](#) in *AWS CLI Command Reference*.

describe-document-classifier

The following code example shows how to use `describe-document-classifier`.

AWS CLI

To describe a document classifier

The following `describe-document-classifier` example gets the properties of a custom document classifier model.

```
aws comprehend describe-document-classifier \  
  --document-classifier-arn arn:aws:comprehend:us-west-2:111122223333:document-  
  classifier/example-classifier-1
```

Output:

```
{  
  "DocumentClassifierProperties": {  
    "DocumentClassifierArn": "arn:aws:comprehend:us-  
west-2:111122223333:document-classifier/example-classifier-1",  
    "LanguageCode": "en",  
    "Status": "TRAINED",  
    "SubmitTime": "2023-06-13T19:04:15.735000+00:00",  
    "EndTime": "2023-06-13T19:42:31.752000+00:00",  
    "TrainingStartTime": "2023-06-13T19:08:20.114000+00:00",  
    "TrainingEndTime": "2023-06-13T19:41:35.080000+00:00",  
    "InputDataConfig": {  
      "DataFormat": "COMPREHEND_CSV",  
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/trainingdata"  
    },  
    "OutputDataConfig": {},  
    "ClassifierMetadata": {  
      "NumberOfLabels": 3,  
      "NumberOfTrainedDocuments": 5016,  
      "NumberOfTestDocuments": 557,  
      "EvaluationMetrics": {  
        "Accuracy": 0.9856,  
        "Precision": 0.9919,  
        "Recall": 0.9459,  
        "F1Score": 0.9673,  
        "MicroPrecision": 0.9856,  
        "MicroRecall": 0.9856,  
        "MicroF1Score": 0.9856,  
        "HammingLoss": 0.0144  
      }  
    }  
  },  
}
```

```
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role",
    "Mode": "MULTI_CLASS"
  }
}
```

For more information, see [Creating and managing custom models](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DescribeDocumentClassifier](#) in *AWS CLI Command Reference*.

describe-dominant-language-detection-job

The following code example shows how to use `describe-dominant-language-detection-job`.

AWS CLI

To describe a dominant language detection job.

The following `describe-dominant-language-detection-job` example gets the properties of an asynchronous dominant language detection job.

```
aws comprehend describe-dominant-language-detection-job \
  --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

Output:

```
{
  "DominantLanguageDetectionJobProperties": {
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:dominant-language-
detection-job/123456abcdeb0e11022f22a11EXAMPLE",
    "JobName": "languageanalysis1",
    "JobStatus": "IN_PROGRESS",
    "SubmitTime": "2023-06-09T18:10:38.037000+00:00",
    "InputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET",
      "InputFormat": "ONE_DOC_PER_LINE"
    },
    "OutputDataConfig": {
```

```

        "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/testfolder/111122223333-
LANGUAGE-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
    },
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
}
}

```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DescribeDominantLanguageDetectionJob](#) in *AWS CLI Command Reference*.

describe-endpoint

The following code example shows how to use describe-endpoint.

AWS CLI

To describe a specific endpoint

The following describe-endpoint example gets the properties of a model-specific endpoint.

```

aws comprehend describe-endpoint \
  --endpoint-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier-
endpoint/example-classifier-endpoint

```

Output:

```

{
  "EndpointProperties": {
    "EndpointArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier-endpoint/example-classifier-endpoint",
    "Status": "IN_SERVICE",
    "ModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-classifier/
exampleclassifier1",
    "DesiredModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier/exampleclassifier1",
    "DesiredInferenceUnits": 1,
    "CurrentInferenceUnits": 1,
    "CreationTime": "2023-06-13T20:32:54.526000+00:00",
    "LastModifiedTime": "2023-06-13T20:32:54.526000+00:00"
  }
}

```

```
}  
}
```

For more information, see [Managing Amazon Comprehend endpoints](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DescribeEndpoint](#) in *AWS CLI Command Reference*.

describe-entities-detection-job

The following code example shows how to use `describe-entities-detection-job`.

AWS CLI

To describe an entities detection job

The following `describe-entities-detection-job` example gets the properties of an asynchronous entities detection job.

```
aws comprehend describe-entities-detection-job \  
  --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

Output:

```
{  
  "EntitiesDetectionJobProperties": {  
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE",  
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:entities-detection-  
job/123456abcdeb0e11022f22a11EXAMPLE",  
    "JobName": "example-entity-detector",  
    "JobStatus": "COMPLETED",  
    "SubmitTime": "2023-06-08T21:30:15.323000+00:00",  
    "EndTime": "2023-06-08T21:40:23.509000+00:00",  
    "InputDataConfig": {  
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/AsyncBatchJobs/",  
      "InputFormat": "ONE_DOC_PER_LINE"  
    },  
    "OutputDataConfig": {  
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/thefolder/111122223333-  
NER-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"  
    },  
    "LanguageCode": "en",  
  }  
}
```

```
    "DataAccessRoleArn": "arn:aws:iam::12345678012:role/service-role/
AmazonComprehendServiceRole-example-role"
  }
}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DescribeEntitiesDetectionJob](#) in *AWS CLI Command Reference*.

describe-entity-recognizer

The following code example shows how to use describe-entity-recognizer.

AWS CLI

To describe an entity recognizer

The following describe-entity-recognizer example gets the properties of a custom entity recognizer model.

```
aws comprehend describe-entity-recognizer \
  entity-recognizer-arn arn:aws:comprehend:us-west-2:111122223333:entity-
recognizer/business-recongizer-1/version/1
```

Output:

```
{
  "EntityRecognizerProperties": {
    "EntityRecognizerArn": "arn:aws:comprehend:us-west-2:111122223333:entity-
recognizer/business-recongizer-1/version/1",
    "LanguageCode": "en",
    "Status": "TRAINED",
    "SubmitTime": "2023-06-14T20:44:59.631000+00:00",
    "EndTime": "2023-06-14T20:59:19.532000+00:00",
    "TrainingStartTime": "2023-06-14T20:48:52.811000+00:00",
    "TrainingEndTime": "2023-06-14T20:58:11.473000+00:00",
    "InputDataConfig": {
      "DataFormat": "COMPREHEND_CSV",
      "EntityTypes": [
        {
          "Type": "BUSINESS"
        }
      ]
    }
  }
}
```



```

    ],
    "Documents": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/trainingdata/dataset/",
      "InputFormat": "ONE_DOC_PER_LINE"
    },
    "EntityList": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/trainingdata/entity.csv"
    }
  },
  "RecognizerMetadata": {
    "NumberOfTrainedDocuments": 1814,
    "NumberOfTestDocuments": 486,
    "EvaluationMetrics": {
      "Precision": 100.0,
      "Recall": 100.0,
      "F1Score": 100.0
    },
    "EntityTypes": [
      {
        "Type": "BUSINESS",
        "EvaluationMetrics": {
          "Precision": 100.0,
          "Recall": 100.0,
          "F1Score": 100.0
        },
        "NumberOfTrainMentions": 1520
      }
    ]
  },
  "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/AmazonComprehendServiceRole-example-role",
  "VersionName": "1"
}

```

For more information, see [Custom entity recognition](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DescribeEntityRecognizer](#) in *AWS CLI Command Reference*.

describe-events-detection-job

The following code example shows how to use `describe-events-detection-job`.

AWS CLI

To describe an events detection job.

The following `describe-events-detection-job` example gets the properties of an asynchronous events detection job.

```
aws comprehend describe-events-detection-job \  
  --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

Output:

```
{  
  "EventsDetectionJobProperties": {  
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE",  
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:events-detection-  
job/123456abcdeb0e11022f22a11EXAMPLE",  
    "JobName": "events_job_1",  
    "JobStatus": "IN_PROGRESS",  
    "SubmitTime": "2023-06-12T18:45:56.054000+00:00",  
    "InputDataConfig": {  
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/EventsData",  
      "InputFormat": "ONE_DOC_PER_LINE"  
    },  
    "OutputDataConfig": {  
      "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/testfolder/111122223333-  
EVENTS-123456abcdeb0e11022f22a11EXAMPLE/output/"  
    },  
    "LanguageCode": "en",  
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/  
AmazonComprehendServiceRole-example-role",  
    "TargetEventTypes": [  
      "BANKRUPTCY",  
      "EMPLOYMENT",  
      "CORPORATE_ACQUISITION",  
      "CORPORATE_MERGER",  
      "INVESTMENT_GENERAL"  
    ]  
  }  
}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DescribeEventsDetectionJob](#) in *AWS CLI Command Reference*.

describe-flywheel-iteration

The following code example shows how to use `describe-flywheel-iteration`.

AWS CLI

To describe a flywheel iteration

The following `describe-flywheel-iteration` example gets the properties of a flywheel iteration.

```
aws comprehend describe-flywheel-iteration \  
  --flywheel-arn arn:aws:comprehend:us-west-2:111122223333:flywheel/example-  
flywheel \  
  --flywheel-iteration-id 20232222AEXAMPLE
```

Output:

```
{  
  "FlywheelIterationProperties": {  
    "FlywheelArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/flywheel-  
entity",  
    "FlywheelIterationId": "20232222AEXAMPLE",  
    "CreationTime": "2023-06-16T21:10:26.385000+00:00",  
    "EndTime": "2023-06-16T23:33:16.827000+00:00",  
    "Status": "COMPLETED",  
    "Message": "FULL_ITERATION: Flywheel iteration performed all functions  
successfully.",  
    "EvaluatedModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-  
classifier/example-classifier/version/1",  
    "EvaluatedModelMetrics": {  
      "AverageF1Score": 0.7742663922375772,  
      "AveragePrecision": 0.8287636394041166,  
      "AverageRecall": 0.7427084833645399,  
      "AverageAccuracy": 0.8795394154118689  
    },  
    "TrainedModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-  
classifier/example-classifier/version/Comprehend-Generated-v1-bb52d585",  
    "TrainedModelMetrics": {  
      "AverageF1Score": 0.9767700253081214,  

```

```

        "AveragePrecision": 0.9767700253081214,
        "AverageRecall": 0.9767700253081214,
        "AverageAccuracy": 0.9858281665190434
    },
    "EvaluationManifestS3Prefix": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/flywheel-
entity/schemaVersion=1/20230616T200543Z/evaluation/20230616T211026Z/"
}
}

```

For more information, see [Flywheel overview](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DescribeFlywheelIteration](#) in *AWS CLI Command Reference*.

describe-flywheel

The following code example shows how to use `describe-flywheel`.

AWS CLI

To describe a flywheel

The following `describe-flywheel` example gets the properties of a flywheel. In this example, the model associated with the flywheel is a custom classifier model that is trained to classify documents as either spam or nonspam, or, "ham".

```

aws comprehend describe-flywheel \
  --flywheel-arn arn:aws:comprehend:us-west-2:111122223333:flywheel/example-
flywheel

```

Output:

```

{
  "FlywheelProperties": {
    "FlywheelArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/example-
flywheel",
    "ActiveModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier/example-model/version/1",
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role",
    "TaskConfig": {
      "LanguageCode": "en",
      "DocumentClassificationConfig": {
        "Mode": "MULTI_CLASS",

```

```

        "Labels": [
            "ham",
            "spam"
        ]
    },
    "DataLakeS3Uri": "s3://DOC-EXAMPLE-BUCKET/example-flywheel/
schemaVersion=1/20230616T200543Z/",
    "DataSecurityConfig": {},
    "Status": "ACTIVE",
    "ModelType": "DOCUMENT_CLASSIFIER",
    "CreationTime": "2023-06-16T20:05:43.242000+00:00",
    "LastModifiedTime": "2023-06-16T20:21:43.567000+00:00"
}
}

```

For more information, see [Flywheel Overview](#) in *Amazon Comprehend Developer Guide*.

- For API details, see [DescribeFlywheel](#) in *AWS CLI Command Reference*.

describe-key-phrases-detection-job

The following code example shows how to use `describe-key-phrases-detection-job`.

AWS CLI

To describe a key phrases detection job

The following `describe-key-phrases-detection-job` example gets the properties of an asynchronous key phrases detection job.

```
aws comprehend describe-key-phrases-detection-job \
  --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

Output:

```
{
  "KeyPhrasesDetectionJobProperties": {
    "JobId": "69aa080c00fc68934a6a98f10EXAMPLE",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:key-phrases-detection-
job/69aa080c00fc68934a6a98f10EXAMPLE",
    "JobName": "example-key-phrases-detection-job",
    "JobStatus": "COMPLETED",
  }
}
```

```

    "SubmitTime": 1686606439.177,
    "EndTime": 1686606806.157,
    "InputDataConfig": {
      "S3Uri": "s3://dereksbucket1001/EventsData/",
      "InputFormat": "ONE_DOC_PER_LINE"
    },
    "OutputDataConfig": {
      "S3Uri": "s3://dereksbucket1002/testfolder/111122223333-
KP-69aa080c00fc68934a6a98f10EXAMPLE/output/output.tar.gz"
    },
    "LanguageCode": "en",
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-testrole"
  }
}

```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DescribeKeyPhrasesDetectionJob](#) in *AWS CLI Command Reference*.

describe-pii-entities-detection-job

The following code example shows how to use `describe-pii-entities-detection-job`.

AWS CLI

To describe a PII entities detection job

The following `describe-pii-entities-detection-job` example gets the properties of an asynchronous pii entities detection job.

```

aws comprehend describe-pii-entities-detection-job \
  --job-id 123456abcdeb0e11022f22a11EXAMPLE

```

Output:

```

{
  "PiiEntitiesDetectionJobProperties": {
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:pii-entities-detection-
job/123456abcdeb0e11022f22a11EXAMPLE",
    "JobName": "example-pii-entities-job",

```

```

    "JobStatus": "IN_PROGRESS",
    "SubmitTime": "2023-06-08T21:30:15.323000+00:00",
    "EndTime": "2023-06-08T21:40:23.509000+00:00",
    "InputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/AsyncBatchJobs/",
      "InputFormat": "ONE_DOC_PER_LINE"
    },
    "OutputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/thefolder/111122223333-
NER-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
    },
    "LanguageCode": "en",
    "DataAccessRoleArn": "arn:aws:iam::12345678012:role/service-role/
AmazonComprehendServiceRole-example-role"
  }
}

```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DescribePiiEntitiesDetectionJob](#) in *AWS CLI Command Reference*.

describe-resource-policy

The following code example shows how to use `describe-resource-policy`.

AWS CLI

To describe a resource policy attached to a model

The following `describe-resource-policy` example gets the properties of a resource-based policy attached to a model.

```

aws comprehend describe-resource-policy \
  --resource-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/
example-classifier/version/1

```

Output:

```

{
  "ResourcePolicy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":
\"Allow\",\"Principal\":{\"AWS\":\"arn:aws:iam::444455556666:root\"},\"Action\":
\"comprehend:ImportModel\",\"Resource\":\"*\"}]}",

```

```
"CreationTime": "2023-06-19T18:44:26.028000+00:00",  
"LastModifiedTime": "2023-06-19T18:53:02.002000+00:00",  
"PolicyRevisionId": "baa675d069d07afaa2aa3106ae280f61"  
}
```

For more information, see [Copying custom models between AWS accounts](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DescribeResourcePolicy](#) in *AWS CLI Command Reference*.

describe-sentiment-detection-job

The following code example shows how to use `describe-sentiment-detection-job`.

AWS CLI

To describe a sentiment detection job

The following `describe-sentiment-detection-job` example gets the properties of an asynchronous sentiment detection job.

```
aws comprehend describe-sentiment-detection-job \  
  --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

Output:

```
{  
  "SentimentDetectionJobProperties": {  
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE",  
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:sentiment-detection-  
job/123456abcdeb0e11022f22a11EXAMPLE",  
    "JobName": "movie_review_analysis",  
    "JobStatus": "IN_PROGRESS",  
    "SubmitTime": "2023-06-09T23:16:15.956000+00:00",  
    "InputDataConfig": {  
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/MovieData",  
      "InputFormat": "ONE_DOC_PER_LINE"  
    },  
    "OutputDataConfig": {  
      "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/testfolder/111122223333-  
TS-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"  
    },  
  },  
}
```



```
    "LanguageCode": "en",
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-servicerole"
  }
}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DescribeSentimentDetectionJob](#) in *AWS CLI Command Reference*.

describe-targeted-sentiment-detection-job

The following code example shows how to use `describe-targeted-sentiment-detection-job`.

AWS CLI

To describe a targeted sentiment detection job

The following `describe-targeted-sentiment-detection-job` example gets the properties of an asynchronous targeted sentiment detection job.

```
aws comprehend describe-targeted-sentiment-detection-job \
  --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

Output:

```
{
  "TargetedSentimentDetectionJobProperties": {
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:targeted-sentiment-
detection-job/123456abcdeb0e11022f22a11EXAMPLE",
    "JobName": "movie_review_analysis",
    "JobStatus": "IN_PROGRESS",
    "SubmitTime": "2023-06-09T23:16:15.956000+00:00",
    "InputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/MovieData",
      "InputFormat": "ONE_DOC_PER_LINE"
    },
    "OutputDataConfig": {
```

```

        "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/testfolder/111122223333-
TS-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
    },
    "LanguageCode": "en",
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-servicerole"
}
}

```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DescribeTargetedSentimentDetectionJob](#) in *AWS CLI Command Reference*.

describe-topics-detection-job

The following code example shows how to use `describe-topics-detection-job`.

AWS CLI

To describe a topics detection job

The following `describe-topics-detection-job` example gets the properties of an asynchronous topics detection job.

```

aws comprehend describe-topics-detection-job \
  --job-id 123456abcdeb0e11022f22a11EXAMPLE

```

Output:

```

{
  "TopicsDetectionJobProperties": {
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:topics-detection-
job/123456abcdeb0e11022f22a11EXAMPLE",
    "JobName": "example_topics_detection",
    "JobStatus": "IN_PROGRESS",
    "SubmitTime": "2023-06-09T18:44:43.414000+00:00",
    "InputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET",
      "InputFormat": "ONE_DOC_PER_LINE"
    }
  },

```

```
    "OutputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/testfolder/111122223333-
TOPICS-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
    },
    "NumberOfTopics": 10,
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-examplerole"
  }
}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DescribeTopicsDetectionJob](#) in *AWS CLI Command Reference*.

detect-dominant-language

The following code example shows how to use detect-dominant-language.

AWS CLI

To detect the dominant language of input text

The following detect-dominant-language analyzes the input text and identifies the dominant language. The pre-trained model's confidence score is also output.

```
aws comprehend detect-dominant-language \
  --text "It is a beautiful day in Seattle."
```

Output:

```
{
  "Languages": [
    {
      "LanguageCode": "en",
      "Score": 0.9877256155014038
    }
  ]
}
```

For more information, see [Dominant Language](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DetectDominantLanguage](#) in *AWS CLI Command Reference*.

detect-entities

The following code example shows how to use detect-entities.

AWS CLI

To detect named entities in input text

The following detect-entities example analyzes the input text and returns the named entities. The pre-trained model's confidence score is also output for each prediction.

```
aws comprehend detect-entities \  
  --language-code en \  
  --text "Hello Zhang Wei, I am John. Your AnyCompany Financial Services, LLC  
credit card \  
  account 1111-XXXX-1111-XXXX has a minimum payment of $24.53 that is due by July  
31st. Based on your autopay settings, \  
  we will withdraw your payment on the due date from your bank account number  
XXXXXX1111 with the routing number XXXXX0000. \  
  Customer feedback for Sunshine Spa, 123 Main St, Anywhere. Send comments to  
Alice at AnySpa@example.com."
```

Output:

```
{  
  "Entities": [  
    {  
      "Score": 0.9994556307792664,  
      "Type": "PERSON",  
      "Text": "Zhang Wei",  
      "BeginOffset": 6,  
      "EndOffset": 15  
    },  
    {  
      "Score": 0.9981022477149963,  
      "Type": "PERSON",  
      "Text": "John",  
      "BeginOffset": 22,  
      "EndOffset": 26  
    },  
    {  
      "Score": 0.9986887574195862,  
      "Type": "ORGANIZATION",
```

```
    "Text": "AnyCompany Financial Services, LLC",
    "BeginOffset": 33,
    "EndOffset": 67
  },
  {
    "Score": 0.9959119558334351,
    "Type": "OTHER",
    "Text": "1111-XXXX-1111-XXXX",
    "BeginOffset": 88,
    "EndOffset": 107
  },
  {
    "Score": 0.9708039164543152,
    "Type": "QUANTITY",
    "Text": ".53",
    "BeginOffset": 133,
    "EndOffset": 136
  },
  {
    "Score": 0.9987268447875977,
    "Type": "DATE",
    "Text": "July 31st",
    "BeginOffset": 152,
    "EndOffset": 161
  },
  {
    "Score": 0.9858865737915039,
    "Type": "OTHER",
    "Text": "XXXXXX1111",
    "BeginOffset": 271,
    "EndOffset": 281
  },
  {
    "Score": 0.9700471758842468,
    "Type": "OTHER",
    "Text": "XXXXX0000",
    "BeginOffset": 306,
    "EndOffset": 315
  },
  {
    "Score": 0.9591118693351746,
    "Type": "ORGANIZATION",
    "Text": "Sunshine Spa",
    "BeginOffset": 340,
```

```
    "EndOffset": 352
  },
  {
    "Score": 0.9797496795654297,
    "Type": "LOCATION",
    "Text": "123 Main St",
    "BeginOffset": 354,
    "EndOffset": 365
  },
  {
    "Score": 0.994929313659668,
    "Type": "PERSON",
    "Text": "Alice",
    "BeginOffset": 394,
    "EndOffset": 399
  },
  {
    "Score": 0.9949769377708435,
    "Type": "OTHER",
    "Text": "AnySpa@example.com",
    "BeginOffset": 403,
    "EndOffset": 418
  }
]
}
```

For more information, see [Entities](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DetectEntities](#) in *AWS CLI Command Reference*.

detect-key-phrases

The following code example shows how to use detect-key-phrases.

AWS CLI

To detect key phrases in input text

The following detect-key-phrases example analyzes the input text and identifies the key noun phrases. The pre-trained model's confidence score is also output for each prediction.

```
aws comprehend detect-key-phrases \
  --language-code en \
```

```
--text "Hello Zhang Wei, I am John. Your AnyCompany Financial Services, LLC
credit card \
    account 1111-XXXX-1111-XXXX has a minimum payment of $24.53 that is due by
July 31st. Based on your autopay settings, \
    we will withdraw your payment on the due date from your bank account number
XXXXXXXX1111 with the routing number XXXXX0000. \
    Customer feedback for Sunshine Spa, 123 Main St, Anywhere. Send comments to
Alice at AnySpa@example.com."
```

Output:

```
{
  "KeyPhrases": [
    {
      "Score": 0.8996376395225525,
      "Text": "Zhang Wei",
      "BeginOffset": 6,
      "EndOffset": 15
    },
    {
      "Score": 0.9992469549179077,
      "Text": "John",
      "BeginOffset": 22,
      "EndOffset": 26
    },
    {
      "Score": 0.988385021686554,
      "Text": "Your AnyCompany Financial Services",
      "BeginOffset": 28,
      "EndOffset": 62
    },
    {
      "Score": 0.8740853071212769,
      "Text": "LLC credit card account 1111-XXXX-1111-XXXX",
      "BeginOffset": 64,
      "EndOffset": 107
    },
    {
      "Score": 0.9999437928199768,
      "Text": "a minimum payment",
      "BeginOffset": 112,
      "EndOffset": 129
    }
  ]
}
```

```
{
  "Score": 0.9998900890350342,
  "Text": ".53",
  "BeginOffset": 133,
  "EndOffset": 136
},
{
  "Score": 0.9979453086853027,
  "Text": "July 31st",
  "BeginOffset": 152,
  "EndOffset": 161
},
{
  "Score": 0.9983011484146118,
  "Text": "your autopay settings",
  "BeginOffset": 172,
  "EndOffset": 193
},
{
  "Score": 0.9996572136878967,
  "Text": "your payment",
  "BeginOffset": 211,
  "EndOffset": 223
},
{
  "Score": 0.9995037317276001,
  "Text": "the due date",
  "BeginOffset": 227,
  "EndOffset": 239
},
{
  "Score": 0.9702621698379517,
  "Text": "your bank account number XXXXXX1111",
  "BeginOffset": 245,
  "EndOffset": 280
},
{
  "Score": 0.9179925918579102,
  "Text": "the routing number XXXXX0000.Customer feedback",
  "BeginOffset": 286,
  "EndOffset": 332
},
{
  "Score": 0.9978160858154297,
```



```
    "Text": "Sunshine Spa",
    "BeginOffset": 337,
    "EndOffset": 349
  },
  {
    "Score": 0.9706913232803345,
    "Text": "123 Main St",
    "BeginOffset": 351,
    "EndOffset": 362
  },
  {
    "Score": 0.9941995143890381,
    "Text": "comments",
    "BeginOffset": 379,
    "EndOffset": 387
  },
  {
    "Score": 0.9759287238121033,
    "Text": "Alice",
    "BeginOffset": 391,
    "EndOffset": 396
  },
  {
    "Score": 0.8376792669296265,
    "Text": "AnySpa@example.com",
    "BeginOffset": 400,
    "EndOffset": 415
  }
]
}
```

For more information, see [Key Phrases](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DetectKeyPhrases](#) in *AWS CLI Command Reference*.

detect-pii-entities

The following code example shows how to use `detect-pii-entities`.

AWS CLI

To detect pii entities in input text

The following `detect-pii-entities` example analyzes the input text and identifies entities that contain personally identifiable information (PII). The pre-trained model's confidence score is also output for each prediction.

```
aws comprehend detect-pii-entities \  
  --language-code en \  
  --text "Hello Zhang Wei, I am John. Your AnyCompany Financial Services, LLC  
credit card \  
  account 1111-XXXX-1111-XXXX has a minimum payment of $24.53 that is due by  
July 31st. Based on your autopay settings, \  
  we will withdraw your payment on the due date from your bank account number  
XXXXXX1111 with the routing number XXXXX0000. \  
  Customer feedback for Sunshine Spa, 123 Main St, Anywhere. Send comments to  
Alice at AnySpa@example.com."
```

Output:

```
{  
  "Entities": [  
    {  
      "Score": 0.9998322129249573,  
      "Type": "NAME",  
      "BeginOffset": 6,  
      "EndOffset": 15  
    },  
    {  
      "Score": 0.9998878240585327,  
      "Type": "NAME",  
      "BeginOffset": 22,  
      "EndOffset": 26  
    },  
    {  
      "Score": 0.9994089603424072,  
      "Type": "CREDIT_DEBIT_NUMBER",  
      "BeginOffset": 88,  
      "EndOffset": 107  
    },  
    {  
      "Score": 0.9999760985374451,  
      "Type": "DATE_TIME",  
      "BeginOffset": 152,  
      "EndOffset": 161  
    }  
  ],  
}
```

```
{
  "Score": 0.9999449253082275,
  "Type": "BANK_ACCOUNT_NUMBER",
  "BeginOffset": 271,
  "EndOffset": 281
},
{
  "Score": 0.9999847412109375,
  "Type": "BANK_ROUTING",
  "BeginOffset": 306,
  "EndOffset": 315
},
{
  "Score": 0.999925434589386,
  "Type": "ADDRESS",
  "BeginOffset": 354,
  "EndOffset": 365
},
{
  "Score": 0.9989161491394043,
  "Type": "NAME",
  "BeginOffset": 394,
  "EndOffset": 399
},
{
  "Score": 0.9994171857833862,
  "Type": "EMAIL",
  "BeginOffset": 403,
  "EndOffset": 418
}
]
```

For more information, see [Personally Identifiable Information \(PII\)](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DetectPiiEntities](#) in *AWS CLI Command Reference*.

detect-sentiment

The following code example shows how to use `detect-sentiment`.

AWS CLI

To detect the sentiment of an input text

The following `detect-sentiment` example analyzes the input text and returns an inference of the prevailing sentiment (POSITIVE, NEUTRAL, MIXED, or NEGATIVE).

```
aws comprehend detect-sentiment \  
  --language-code en \  
  --text "It is a beautiful day in Seattle"
```

Output:

```
{  
  "Sentiment": "POSITIVE",  
  "SentimentScore": {  
    "Positive": 0.9976957440376282,  
    "Negative": 9.653854067437351e-05,  
    "Neutral": 0.002169104292988777,  
    "Mixed": 3.857641786453314e-05  
  }  
}
```

For more information, see [Sentiment](#) in the *Amazon Comprehend Developer Guide*

- For API details, see [DetectSentiment](#) in *AWS CLI Command Reference*.

detect-syntax

The following code example shows how to use `detect-syntax`.

AWS CLI

To detect the parts of speech in an input text

The following `detect-syntax` example analyzes the syntax of the input text and returns the different parts of speech. The pre-trained model's confidence score is also output for each prediction.

```
aws comprehend detect-syntax \  
  --language-code en \  
  --text "It is a beautiful day in Seattle."
```

Output:

```
{
  "SyntaxTokens": [
    {
      "TokenId": 1,
      "Text": "It",
      "BeginOffset": 0,
      "EndOffset": 2,
      "PartOfSpeech": {
        "Tag": "PRON",
        "Score": 0.9999740719795227
      }
    },
    {
      "TokenId": 2,
      "Text": "is",
      "BeginOffset": 3,
      "EndOffset": 5,
      "PartOfSpeech": {
        "Tag": "VERB",
        "Score": 0.999901294708252
      }
    },
    {
      "TokenId": 3,
      "Text": "a",
      "BeginOffset": 6,
      "EndOffset": 7,
      "PartOfSpeech": {
        "Tag": "DET",
        "Score": 0.9999938607215881
      }
    },
    {
      "TokenId": 4,
      "Text": "beautiful",
      "BeginOffset": 8,
      "EndOffset": 17,
      "PartOfSpeech": {
        "Tag": "ADJ",
        "Score": 0.9987351894378662
      }
    }
  ],
}
```

```
{
  "TokenId": 5,
  "Text": "day",
  "BeginOffset": 18,
  "EndOffset": 21,
  "PartOfSpeech": {
    "Tag": "NOUN",
    "Score": 0.9999796748161316
  }
},
{
  "TokenId": 6,
  "Text": "in",
  "BeginOffset": 22,
  "EndOffset": 24,
  "PartOfSpeech": {
    "Tag": "ADP",
    "Score": 0.9998047947883606
  }
},
{
  "TokenId": 7,
  "Text": "Seattle",
  "BeginOffset": 25,
  "EndOffset": 32,
  "PartOfSpeech": {
    "Tag": "PROPN",
    "Score": 0.9940530061721802
  }
}
]
```

For more information, see [Syntax Analysis](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DetectSyntax](#) in *AWS CLI Command Reference*.

detect-targeted-sentiment

The following code example shows how to use `detect-targeted-sentiment`.

AWS CLI

To detect the targeted sentiment of named entities in an input text

The following `detect-targeted-sentiment` example analyzes the input text and returns the named entities in addition to the targeted sentiment associated with each entity. The pre-trained models confidence score for each prediction is also output.

```
aws comprehend detect-targeted-sentiment \  
  --language-code en \  
  --text "I do not enjoy January because it is too cold but August is the perfect  
  temperature"
```

Output:

```
{  
  "Entities": [  
    {  
      "DescriptiveMentionIndex": [  
        0  
      ],  
      "Mentions": [  
        {  
          "Score": 0.9999979734420776,  
          "GroupScore": 1.0,  
          "Text": "I",  
          "Type": "PERSON",  
          "MentionSentiment": {  
            "Sentiment": "NEUTRAL",  
            "SentimentScore": {  
              "Positive": 0.0,  
              "Negative": 0.0,  
              "Neutral": 1.0,  
              "Mixed": 0.0  
            }  
          },  
          "BeginOffset": 0,  
          "EndOffset": 1  
        }  
      ]  
    },  
    {  
      "DescriptiveMentionIndex": [  
        0  
      ],  
      "Mentions": [  
        {
```

```
        "Score": 0.9638869762420654,
        "GroupScore": 1.0,
        "Text": "January",
        "Type": "DATE",
        "MentionSentiment": {
            "Sentiment": "NEGATIVE",
            "SentimentScore": {
                "Positive": 0.0031610000878572464,
                "Negative": 0.9967250227928162,
                "Neutral": 0.00011100000119768083,
                "Mixed": 1.9999999949504854e-06
            }
        },
        "BeginOffset": 15,
        "EndOffset": 22
    }
]
},
{
    "DescriptiveMentionIndex": [
        0
    ],
    "Mentions": [
        {
            "Score": 0.9664419889450073,
            "GroupScore": 1.0,
            "Text": "August",
            "Type": "DATE",
            "MentionSentiment": {
                "Sentiment": "POSITIVE",
                "SentimentScore": {
                    "Positive": 0.9999549984931946,
                    "Negative": 3.999999989900971e-06,
                    "Neutral": 4.099999932805076e-05,
                    "Mixed": 0.0
                }
            },
            "BeginOffset": 50,
            "EndOffset": 56
        }
    ]
},
{
```



```

    "DescriptiveMentionIndex": [
      0
    ],
    "Mentions": [
      {
        "Score": 0.9803199768066406,
        "GroupScore": 1.0,
        "Text": "temperature",
        "Type": "ATTRIBUTE",
        "MentionSentiment": {
          "Sentiment": "POSITIVE",
          "SentimentScore": {
            "Positive": 1.0,
            "Negative": 0.0,
            "Neutral": 0.0,
            "Mixed": 0.0
          }
        },
        "BeginOffset": 77,
        "EndOffset": 88
      }
    ]
  }
}

```

For more information, see [Targeted Sentiment](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [DetectTargetedSentiment](#) in *AWS CLI Command Reference*.

import-model

The following code example shows how to use `import-model`.

AWS CLI

To import a model

The following `import-model` example imports a model from a different AWS account. The document classifier model in account 444455556666 has a resource-based policy allowing account 111122223333 to import the model.

```
aws comprehend import-model \
```

```
--source-model-arn arn:aws:comprehend:us-west-2:444455556666:document-
classifier/example-classifier
```

Output:

```
{
  "ModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-classifier/
example-classifier"
}
```

For more information, see [Copying custom models between AWS accounts](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ImportModel](#) in *AWS CLI Command Reference*.

list-datasets

The following code example shows how to use `list-datasets`.

AWS CLI

To list all flywheel datasets

The following `list-datasets` example lists all datasets associated with a flywheel.

```
aws comprehend list-datasets \
  --flywheel-arn arn:aws:comprehend:us-west-2:111122223333:flywheel/flywheel-
entity
```

Output:

```
{
  "DatasetPropertiesList": [
    {
      "DatasetArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/
flywheel-entity/dataset/example-dataset-1",
      "DatasetName": "example-dataset-1",
      "DatasetType": "TRAIN",
      "DatasetS3Uri": "s3://DOC-EXAMPLE-BUCKET/flywheel-entity/
schemaVersion=1/20230616T200543Z/datasets/example-dataset-1/20230616T203710Z/",
      "Status": "CREATING",
    }
  ]
}
```

```

        "CreationTime": "2023-06-16T20:37:10.400000+00:00"
    },
    {
        "DatasetArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/
flywheel-entity/dataset/example-dataset-2",
        "DatasetName": "example-dataset-2",
        "DatasetType": "TRAIN",
        "DatasetS3Uri": "s3://DOC-EXAMPLE-BUCKET/flywheel-entity/
schemaVersion=1/20230616T200543Z/datasets/example-dataset-2/20230616T200607Z/",
        "Description": "TRAIN Dataset created by Flywheel creation.",
        "Status": "COMPLETED",
        "NumberOfDocuments": 5572,
        "CreationTime": "2023-06-16T20:06:07.722000+00:00"
    }
]
}

```

For more information, see [Flywheel Overview](#) in *Amazon Comprehend Developer Guide*.

- For API details, see [ListDatasets](#) in *AWS CLI Command Reference*.

list-document-classification-jobs

The following code example shows how to use `list-document-classification-jobs`.

AWS CLI

To list of all document classification jobs

The following `list-document-classification-jobs` example lists all document classification jobs.

```
aws comprehend list-document-classification-jobs
```

Output:

```

{
  "DocumentClassificationJobPropertiesList": [
    {
      "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
      "JobArn": "arn:aws:comprehend:us-west-2:1234567890101:document-
classification-job/123456abcdeb0e11022f22a11EXAMPLE",

```

```

    "JobName": "exampleclassificationjob",
    "JobStatus": "COMPLETED",
    "SubmitTime": "2023-06-14T17:09:51.788000+00:00",
    "EndTime": "2023-06-14T17:15:58.582000+00:00",
    "DocumentClassifierArn": "arn:aws:comprehend:us-
west-2:1234567890101:document-classifier/mymodel/version/12",
    "InputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/jobdata/",
      "InputFormat": "ONE_DOC_PER_LINE"
    },
    "OutputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/
thefolder/1234567890101-CLN-e758dd56b824aa717ceab551f11749fb/output/output.tar.gz"
    },
    "DataAccessRoleArn": "arn:aws:iam::1234567890101:role/service-role/
AmazonComprehendServiceRole-example-role"
  },
  {
    "JobId": "123456abcdeb0e11022f22a1EXAMPLE2",
    "JobArn": "arn:aws:comprehend:us-west-2:1234567890101:document-
classification-job/123456abcdeb0e11022f22a1EXAMPLE2",
    "JobName": "exampleclassificationjob2",
    "JobStatus": "COMPLETED",
    "SubmitTime": "2023-06-14T17:22:39.829000+00:00",
    "EndTime": "2023-06-14T17:28:46.107000+00:00",
    "DocumentClassifierArn": "arn:aws:comprehend:us-
west-2:1234567890101:document-classifier/mymodel/version/12",
    "InputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/jobdata/",
      "InputFormat": "ONE_DOC_PER_LINE"
    },
    "OutputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/
thefolder/1234567890101-CLN-123456abcdeb0e11022f22a1EXAMPLE2/output/output.tar.gz"
    },
    "DataAccessRoleArn": "arn:aws:iam::1234567890101:role/service-role/
AmazonComprehendServiceRole-example-role"
  }
]
}

```

For more information, see [Custom Classification](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ListDocumentClassificationJobs](#) in *AWS CLI Command Reference*.

list-document-classifier-summaries

The following code example shows how to use `list-document-classifier-summaries`.

AWS CLI

To list the summaries of all created document classifiers

The following `list-document-classifier-summaries` example lists all created document classifier summaries.

```
aws comprehend list-document-classifier-summaries
```

Output:

```
{
  "DocumentClassifierSummariesList": [
    {
      "DocumentClassifierName": "example-classifier-1",
      "NumberOfVersions": 1,
      "LatestVersionCreatedAt": "2023-06-13T22:07:59.825000+00:00",
      "LatestVersionName": "1",
      "LatestVersionStatus": "TRAINED"
    },
    {
      "DocumentClassifierName": "example-classifier-2",
      "NumberOfVersions": 2,
      "LatestVersionCreatedAt": "2023-06-13T21:54:59.589000+00:00",
      "LatestVersionName": "2",
      "LatestVersionStatus": "TRAINED"
    }
  ]
}
```

For more information, see [Creating and managing custom models](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ListDocumentClassifierSummaries](#) in *AWS CLI Command Reference*.

list-document-classifiers

The following code example shows how to use `list-document-classifiers`.

AWS CLI

To list of all document classifiers

The following `list-document-classifiers` example lists all trained and in-training document classifier models.

```
aws comprehend list-document-classifiers
```

Output:

```
{
  "DocumentClassifierPropertiesList": [
    {
      "DocumentClassifierArn": "arn:aws:comprehend:us-
west-2:111122223333:document-classifier/exampleclassifier1",
      "LanguageCode": "en",
      "Status": "TRAINED",
      "SubmitTime": "2023-06-13T19:04:15.735000+00:00",
      "EndTime": "2023-06-13T19:42:31.752000+00:00",
      "TrainingStartTime": "2023-06-13T19:08:20.114000+00:00",
      "TrainingEndTime": "2023-06-13T19:41:35.080000+00:00",
      "InputDataConfig": {
        "DataFormat": "COMPREHEND_CSV",
        "S3Uri": "s3://DOC-EXAMPLE-BUCKET/trainingdata"
      },
      "OutputDataConfig": {},
      "ClassifierMetadata": {
        "NumberOfLabels": 3,
        "NumberOfTrainedDocuments": 5016,
        "NumberOfTestDocuments": 557,
        "EvaluationMetrics": {
          "Accuracy": 0.9856,
          "Precision": 0.9919,
          "Recall": 0.9459,
          "F1Score": 0.9673,
          "MicroPrecision": 0.9856,
          "MicroRecall": 0.9856,
          "MicroF1Score": 0.9856,
          "HammingLoss": 0.0144
        }
      }
    }
  ],
}
```

```

        "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-testorle",
        "Mode": "MULTI_CLASS"
    },
    {
        "DocumentClassifierArn": "arn:aws:comprehend:us-
west-2:111122223333:document-classifier/exampleclassifier2",
        "LanguageCode": "en",
        "Status": "TRAINING",
        "SubmitTime": "2023-06-13T21:20:28.690000+00:00",
        "InputDataConfig": {
            "DataFormat": "COMPREHEND_CSV",
            "S3Uri": "s3://DOC-EXAMPLE-BUCKET/trainingdata"
        },
        "OutputDataConfig": {},
        "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-testorle",
        "Mode": "MULTI_CLASS"
    }
]
}

```

For more information, see [Creating and managing custom models](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ListDocumentClassifiers](#) in *AWS CLI Command Reference*.

list-dominant-language-detection-jobs

The following code example shows how to use `list-dominant-language-detection-jobs`.

AWS CLI

To list all dominant language detection jobs

The following `list-dominant-language-detection-jobs` example lists all in-progress and completed asynchronous dominant language detection jobs.

```
aws comprehend list-dominant-language-detection-jobs
```

Output:

```
{
```

```
"DominantLanguageDetectionJobPropertiesList": [  
  {  
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE",  
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:dominant-language-  
detection-job/123456abcdeb0e11022f22a11EXAMPLE",  
    "JobName": "languageanalysis1",  
    "JobStatus": "COMPLETED",  
    "SubmitTime": "2023-06-09T18:10:38.037000+00:00",  
    "EndTime": "2023-06-09T18:18:45.498000+00:00",  
    "InputDataConfig": {  
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET",  
      "InputFormat": "ONE_DOC_PER_LINE"  
    },  
    "OutputDataConfig": {  
      "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/  
testfolder/111122223333-LANGUAGE-123456abcdeb0e11022f22a11EXAMPLE/output/  
output.tar.gz"  
    },  
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/  
AmazonComprehendServiceRole-example-role"  
  },  
  {  
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE",  
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:dominant-language-  
detection-job/123456abcdeb0e11022f22a11EXAMPLE",  
    "JobName": "languageanalysis2",  
    "JobStatus": "STOPPED",  
    "SubmitTime": "2023-06-09T18:16:33.690000+00:00",  
    "EndTime": "2023-06-09T18:24:40.608000+00:00",  
    "InputDataConfig": {  
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET",  
      "InputFormat": "ONE_DOC_PER_LINE"  
    },  
    "OutputDataConfig": {  
      "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/  
testfolder/111122223333-LANGUAGE-123456abcdeb0e11022f22a11EXAMPLE/output/  
output.tar.gz"  
    },  
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/  
AmazonComprehendServiceRole-example-role"  
  }  
]
```


For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ListDominantLanguageDetectionJobs](#) in *AWS CLI Command Reference*.

list-endpoints

The following code example shows how to use `list-endpoints`.

AWS CLI

To list of all endpoints

The following `list-endpoints` example lists all active model-specific endpoints.

```
aws comprehend list-endpoints
```

Output:

```
{
  "EndpointPropertiesList": [
    {
      "EndpointArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier-endpoint/ExampleClassifierEndpoint",
      "Status": "IN_SERVICE",
      "ModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier/exampleclassifier1",
      "DesiredModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier/exampleclassifier1",
      "DesiredInferenceUnits": 1,
      "CurrentInferenceUnits": 1,
      "CreationTime": "2023-06-13T20:32:54.526000+00:00",
      "LastModifiedTime": "2023-06-13T20:32:54.526000+00:00"
    },
    {
      "EndpointArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier-endpoint/ExampleClassifierEndpoint2",
      "Status": "IN_SERVICE",
      "ModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier/exampleclassifier2",
      "DesiredModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier/exampleclassifier2",
      "DesiredInferenceUnits": 1,

```

```
        "CurrentInferenceUnits": 1,  
        "CreationTime": "2023-06-13T20:32:54.526000+00:00",  
        "LastModifiedTime": "2023-06-13T20:32:54.526000+00:00"  
    }  
]  
}
```

For more information, see [Managing Amazon Comprehend endpoints](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ListEndpoints](#) in *AWS CLI Command Reference*.

list-entities-detection-jobs

The following code example shows how to use `list-entities-detection-jobs`.

AWS CLI

To list all entities detection jobs

The following `list-entities-detection-jobs` example lists all asynchronous entities detection jobs.

```
aws comprehend list-entities-detection-jobs
```

Output:

```
{  
  "EntitiesDetectionJobPropertiesList": [  
    {  
      "JobId": "468af39c28ab45b83eb0c4ab9EXAMPLE",  
      "JobArn": "arn:aws:comprehend:us-west-2:111122223333:entities-detection-job/468af39c28ab45b83eb0c4ab9EXAMPLE",  
      "JobName": "example-entities-detection",  
      "JobStatus": "COMPLETED",  
      "SubmitTime": "2023-06-08T20:57:46.476000+00:00",  
      "EndTime": "2023-06-08T21:05:53.718000+00:00",  
      "InputDataConfig": {  
        "S3Uri": "s3://DOC-EXAMPLE-BUCKET/AsyncBatchJobs/",  
        "InputFormat": "ONE_DOC_PER_LINE"  
      },  
      "OutputDataConfig": {
```

```
        "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/
thefolder/111122223333-NER-468af39c28ab45b83eb0c4ab9EXAMPLE/output/output.tar.gz"
    },
    "LanguageCode": "en",
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
  },
  {
    "JobId": "809691caeaab0e71406f80a28EXAMPLE",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:entities-detection-
job/809691caeaab0e71406f80a28EXAMPLE",
    "JobName": "example-entities-detection-2",
    "JobStatus": "COMPLETED",
    "SubmitTime": "2023-06-08T21:30:15.323000+00:00",
    "EndTime": "2023-06-08T21:40:23.509000+00:00",
    "InputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/AsyncBatchJobs/",
      "InputFormat": "ONE_DOC_PER_LINE"
    },
    "OutputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/
thefolder/111122223333-NER-809691caeaab0e71406f80a28EXAMPLE/output/output.tar.gz"
    },
    "LanguageCode": "en",
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
  },
  {
    "JobId": "e00597c36b448b91d70dea165EXAMPLE",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:entities-detection-
job/e00597c36b448b91d70dea165EXAMPLE",
    "JobName": "example-entities-detection-3",
    "JobStatus": "STOPPED",
    "SubmitTime": "2023-06-08T22:19:28.528000+00:00",
    "EndTime": "2023-06-08T22:27:33.991000+00:00",
    "InputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/AsyncBatchJobs/",
      "InputFormat": "ONE_DOC_PER_LINE"
    },
    "OutputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/
thefolder/111122223333-NER-e00597c36b448b91d70dea165EXAMPLE/output/output.tar.gz"
    },
    "LanguageCode": "en",
```

```
        "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
    }
]
}
```

For more information, see [Entities](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ListEntitiesDetectionJobs](#) in *AWS CLI Command Reference*.

list-entity-recognizer-summaries

The following code example shows how to use `list-entity-recognizer-summaries`.

AWS CLI

To list of summaries for all created entity recognizers

The following `list-entity-recognizer-summaries` example lists all entity recognizer summaries.

```
aws comprehend list-entity-recognizer-summaries
```

Output:

```
{
  "EntityRecognizerSummariesList": [
    {
      "RecognizerName": "entity-recognizer-3",
      "NumberOfVersions": 2,
      "LatestVersionCreatedAt": "2023-06-15T23:15:07.621000+00:00",
      "LatestVersionName": "2",
      "LatestVersionStatus": "STOP_REQUESTED"
    },
    {
      "RecognizerName": "entity-recognizer-2",
      "NumberOfVersions": 1,
      "LatestVersionCreatedAt": "2023-06-14T22:55:27.805000+00:00",
      "LatestVersionName": "2",
      "LatestVersionStatus": "TRAINED"
    },
    {
      "RecognizerName": "entity-recognizer-1",
```

```

        "NumberOfVersions": 1,
        "LatestVersionCreatedAt": "2023-06-14T20:44:59.631000+00:00",
        "LatestVersionName": "1",
        "LatestVersionStatus": "TRAINED"
    }
]
}

```

For more information, see [Custom entity recognition](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ListEntityRecognizerSummaries](#) in *AWS CLI Command Reference*.

list-entity-recognizers

The following code example shows how to use `list-entity-recognizers`.

AWS CLI

To list of all custom entity recognizers

The following `list-entity-recognizers` example lists all created custom entity recognizers.

```
aws comprehend list-entity-recognizers
```

Output:

```

{
  "EntityRecognizerPropertiesList": [
    {
      "EntityRecognizerArn": "arn:aws:comprehend:us-
west-2:111122223333:entity-recognizer/EntityRecognizer/version/1",
      "LanguageCode": "en",
      "Status": "TRAINED",
      "SubmitTime": "2023-06-14T20:44:59.631000+00:00",
      "EndTime": "2023-06-14T20:59:19.532000+00:00",
      "TrainingStartTime": "2023-06-14T20:48:52.811000+00:00",
      "TrainingEndTime": "2023-06-14T20:58:11.473000+00:00",
      "InputDataConfig": {
        "DataFormat": "COMPREHEND_CSV",
        "EntityTypes": [
          {

```

```

        "Type": "BUSINESS"
      }
    ],
    "Documents": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/trainingdata/dataset/",
      "InputFormat": "ONE_DOC_PER_LINE"
    },
    "EntityList": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/trainingdata/entity.csv"
    }
  },
  "RecognizerMetadata": {
    "NumberOfTrainedDocuments": 1814,
    "NumberOfTestDocuments": 486,
    "EvaluationMetrics": {
      "Precision": 100.0,
      "Recall": 100.0,
      "F1Score": 100.0
    },
    "EntityTypes": [
      {
        "Type": "BUSINESS",
        "EvaluationMetrics": {
          "Precision": 100.0,
          "Recall": 100.0,
          "F1Score": 100.0
        },
        "NumberOfTrainMentions": 1520
      }
    ]
  },
  "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/AmazonComprehendServiceRole-servicerole",
  "VersionName": "1"
},
{
  "EntityRecognizerArn": "arn:aws:comprehend:us-west-2:111122223333:entity-recognizer/entityrecognizer3",
  "LanguageCode": "en",
  "Status": "TRAINED",
  "SubmitTime": "2023-06-14T22:57:51.056000+00:00",
  "EndTime": "2023-06-14T23:14:13.894000+00:00",
  "TrainingStartTime": "2023-06-14T23:01:33.984000+00:00",
  "TrainingEndTime": "2023-06-14T23:13:02.984000+00:00",

```

```
"InputDataConfig": {
  "DataFormat": "COMPREHEND_CSV",
  "EntityTypes": [
    {
      "Type": "DEVICE"
    }
  ],
  "Documents": {
    "S3Uri": "s3://DOC-EXAMPLE-BUCKET/trainingdata/raw_txt.csv",
    "InputFormat": "ONE_DOC_PER_LINE"
  },
  "EntityList": {
    "S3Uri": "s3://DOC-EXAMPLE-BUCKET/trainingdata/entity_list.csv"
  }
},
"RecognizerMetadata": {
  "NumberOfTrainedDocuments": 4616,
  "NumberOfTestDocuments": 3489,
  "EvaluationMetrics": {
    "Precision": 98.54227405247813,
    "Recall": 100.0,
    "F1Score": 99.26578560939794
  },
  "EntityTypes": [
    {
      "Type": "DEVICE",
      "EvaluationMetrics": {
        "Precision": 98.54227405247813,
        "Recall": 100.0,
        "F1Score": 99.26578560939794
      },
      "NumberOfTrainMentions": 2764
    }
  ]
},
"DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/AmazonComprehendServiceRole-servicerole"
}
]
```

For more information, see [Custom entity recognition](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ListEntityRecognizers](#) in *AWS CLI Command Reference*.

list-events-detection-jobs

The following code example shows how to use `list-events-detection-jobs`.

AWS CLI

To list all events detection jobs

The following `list-events-detection-jobs` example lists all asynchronous events detection jobs.

```
aws comprehend list-events-detection-jobs
```

Output:

```
{
  "EventsDetectionJobPropertiesList": [
    {
      "JobId": "aa9593f9203e84f3ef032ce18EXAMPLE",
      "JobArn": "arn:aws:comprehend:us-west-2:1111222233333:events-detection-
job/aa9593f9203e84f3ef032ce18EXAMPLE",
      "JobName": "events_job_1",
      "JobStatus": "COMPLETED",
      "SubmitTime": "2023-06-12T19:14:57.751000+00:00",
      "EndTime": "2023-06-12T19:21:04.962000+00:00",
      "InputDataConfig": {
        "S3Uri": "s3://DOC-EXAMPLE-SOURCE-BUCKET/EventsData/",
        "InputFormat": "ONE_DOC_PER_LINE"
      },
      "OutputDataConfig": {
        "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/
testfolder/1111222233333-EVENTS-aa9593f9203e84f3ef032ce18EXAMPLE/output/"
      },
      "LanguageCode": "en",
      "DataAccessRoleArn": "arn:aws:iam::1111222233333:role/service-role/
AmazonComprehendServiceRole-example-role",
      "TargetEventTypes": [
        "BANKRUPTCY",
        "EMPLOYMENT",
        "CORPORATE_ACQUISITION",

```



```

        "CORPORATE_MERGER",
        "INVESTMENT_GENERAL"
    ]
},
{
    "JobId": "4a990a2f7e82adfca6e171135EXAMPLE",
    "JobArn": "arn:aws:comprehend:us-west-2:1111222233333:events-detection-
job/4a990a2f7e82adfca6e171135EXAMPLE",
    "JobName": "events_job_2",
    "JobStatus": "COMPLETED",
    "SubmitTime": "2023-06-12T19:55:43.702000+00:00",
    "EndTime": "2023-06-12T20:03:49.893000+00:00",
    "InputDataConfig": {
        "S3Uri": "s3://DOC-EXAMPLE-SOURCE-BUCKET/EventsData/",
        "InputFormat": "ONE_DOC_PER_LINE"
    },
    "OutputDataConfig": {
        "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/
testfolder/1111222233333-EVENTS-4a990a2f7e82adfca6e171135EXAMPLE/output/"
    },
    "LanguageCode": "en",
    "DataAccessRoleArn": "arn:aws:iam::1111222233333:role/service-role/
AmazonComprehendServiceRole-example-role",
    "TargetEventTypes": [
        "BANKRUPTCY",
        "EMPLOYMENT",
        "CORPORATE_ACQUISITION",
        "CORPORATE_MERGER",
        "INVESTMENT_GENERAL"
    ]
}
]
}

```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ListEventsDetectionJobs](#) in *AWS CLI Command Reference*.

list-flywheel-iteration-history

The following code example shows how to use `list-flywheel-iteration-history`.

AWS CLI

To list all flywheel iteration history

The following `list-flywheel-iteration-history` example lists all iterations of a flywheel.

```
aws comprehend list-flywheel-iteration-history
  --flywheel-arn arn:aws:comprehend:us-west-2:111122223333:flywheel/example-
  flywheel
```

Output:

```
{
  "FlywheelIterationPropertiesList": [
    {
      "FlywheelArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/
example-flywheel",
      "FlywheelIterationId": "20230619TEXAMPLE",
      "CreationTime": "2023-06-19T04:00:32.594000+00:00",
      "EndTime": "2023-06-19T04:00:49.248000+00:00",
      "Status": "COMPLETED",
      "Message": "FULL_ITERATION: Flywheel iteration performed all functions
successfully.",
      "EvaluatedModelArn": "arn:aws:comprehend:us-
west-2:111122223333:document-classifier/example-classifier/version/1",
      "EvaluatedModelMetrics": {
        "AverageF1Score": 0.7742663922375772,
        "AverageF1Score": 0.9876464664646313,
        "AveragePrecision": 0.9800000253081214,
        "AverageRecall": 0.9445600253081214,
        "AverageAccuracy": 0.9997281665190434
      },
      "EvaluationManifestS3Prefix": "s3://DOC-EXAMPLE-BUCKET/example-flywheel/
schemaVersion=1/20230619TEXAMPLE/evaluation/20230619TEXAMPLE/"
    },
    {
      "FlywheelArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/
example-flywheel-2",
      "FlywheelIterationId": "20230616TEXAMPLE",
      "CreationTime": "2023-06-16T21:10:26.385000+00:00",
      "EndTime": "2023-06-16T23:33:16.827000+00:00",
      "Status": "COMPLETED",
```

```

    "Message": "FULL_ITERATION: Flywheel iteration performed all functions
successfully.",
    "EvaluatedModelArn": "arn:aws:comprehend:us-
west-2:111122223333:document-classifier/spamvshamclassify/version/1",
    "EvaluatedModelMetrics": {
        "AverageF1Score": 0.7742663922375772,
        "AverageF1Score": 0.9767700253081214,
        "AveragePrecision": 0.9767700253081214,
        "AverageRecall": 0.9767700253081214,
        "AverageAccuracy": 0.9858281665190434
    },
    "EvaluationManifestS3Prefix": "s3://DOC-EXAMPLE-BUCKET/example-
flywheel-2/schemaVersion=1/20230616TEXAMPLE/evaluation/20230616TEXAMPLE/"
    }
]
}

```

For more information, see [Flywheel overview](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ListFlywheelIterationHistory](#) in *AWS CLI Command Reference*.

list-flywheels

The following code example shows how to use `list-flywheels`.

AWS CLI

To list all flywheels

The following `list-flywheels` example lists all created flywheels.

```
aws comprehend list-flywheels
```

Output:

```

{
  "FlywheelSummaryList": [
    {
      "FlywheelArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/
example-flywheel-1",
      "ActiveModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier/exampleclassifier/version/1",

```

```

        "DataLakeS3Uri": "s3://DOC-EXAMPLE-BUCKET/example-flywheel-1/
schemaVersion=1/20230616T200543Z/",
        "Status": "ACTIVE",
        "ModelType": "DOCUMENT_CLASSIFIER",
        "CreationTime": "2023-06-16T20:05:43.242000+00:00",
        "LastModifiedTime": "2023-06-19T04:00:43.027000+00:00",
        "LatestFlywheelIteration": "20230619T040032Z"
    },
    {
        "FlywheelArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/
example-flywheel-2",
        "ActiveModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-
classifier/exampleclassifier2/version/1",
        "DataLakeS3Uri": "s3://DOC-EXAMPLE-BUCKET/example-flywheel-2/
schemaVersion=1/20220616T200543Z/",
        "Status": "ACTIVE",
        "ModelType": "DOCUMENT_CLASSIFIER",
        "CreationTime": "2022-06-16T20:05:43.242000+00:00",
        "LastModifiedTime": "2022-06-19T04:00:43.027000+00:00",
        "LatestFlywheelIteration": "20220619T040032Z"
    }
]
}

```

For more information, see [Flywheel overview](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ListFlywheels](#) in *AWS CLI Command Reference*.

list-key-phrases-detection-jobs

The following code example shows how to use `list-key-phrases-detection-jobs`.

AWS CLI

To list all key phrases detection jobs

The following `list-key-phrases-detection-jobs` example lists all in-progress and completed asynchronous key phrases detection jobs.

```
aws comprehend list-key-phrases-detection-jobs
```

Output:

```
{
  "KeyPhrasesDetectionJobPropertiesList": [
    {
      "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
      "JobArn": "arn:aws:comprehend:us-west-2:111122223333:key-phrases-
detection-job/123456abcdeb0e11022f22a11EXAMPLE",
      "JobName": "keyphrasesanalysis1",
      "JobStatus": "COMPLETED",
      "SubmitTime": "2023-06-08T22:31:43.767000+00:00",
      "EndTime": "2023-06-08T22:39:52.565000+00:00",
      "InputDataConfig": {
        "S3Uri": "s3://DOC-EXAMPLE-SOURCE-BUCKET/AsyncBatchJobs/",
        "InputFormat": "ONE_DOC_PER_LINE"
      },
      "OutputDataConfig": {
        "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/
testfolder/111122223333-KP-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
      },
      "LanguageCode": "en",
      "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
    },
    {
      "JobId": "123456abcdeb0e11022f22a33EXAMPLE",
      "JobArn": "arn:aws:comprehend:us-west-2:111122223333:key-phrases-
detection-job/123456abcdeb0e11022f22a33EXAMPLE",
      "JobName": "keyphrasesanalysis2",
      "JobStatus": "STOPPED",
      "SubmitTime": "2023-06-08T22:57:52.154000+00:00",
      "EndTime": "2023-06-08T23:05:48.385000+00:00",
      "InputDataConfig": {
        "S3Uri": "s3://DOC-EXAMPLE-BUCKET/AsyncBatchJobs/",
        "InputFormat": "ONE_DOC_PER_LINE"
      },
      "OutputDataConfig": {
        "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/
testfolder/111122223333-KP-123456abcdeb0e11022f22a33EXAMPLE/output/output.tar.gz"
      },
      "LanguageCode": "en",
      "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
    },
    {
```

```

    "JobId": "123456abcdeb0e11022f22a44EXAMPLE",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:key-phrases-
detection-job/123456abcdeb0e11022f22a44EXAMPLE",
    "JobName": "keyphrasesanalysis3",
    "JobStatus": "FAILED",
    "Message": "NO_READ_ACCESS_TO_INPUT: The provided data access role does
not have proper access to the input data.",
    "SubmitTime": "2023-06-09T16:47:04.029000+00:00",
    "EndTime": "2023-06-09T16:47:18.413000+00:00",
    "InputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET",
      "InputFormat": "ONE_DOC_PER_LINE"
    },
    "OutputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/
testfolder/111122223333-KP-123456abcdeb0e11022f22a44EXAMPLE/output/output.tar.gz"
    },
    "LanguageCode": "en",
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
  }
]
}

```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ListKeyPhrasesDetectionJobs](#) in *AWS CLI Command Reference*.

list-pii-entities-detection-jobs

The following code example shows how to use `list-pii-entities-detection-jobs`.

AWS CLI

To list all pii entities detection jobs

The following `list-pii-entities-detection-jobs` example lists all in-progress and completed asynchronous pii detection jobs.

```
aws comprehend list-pii-entities-detection-jobs
```

Output:

```
{
  "PiiEntitiesDetectionJobPropertiesList": [
    {
      "JobId": "6f9db0c42d0c810e814670ee4EXAMPLE",
      "JobArn": "arn:aws:comprehend:us-west-2:111122223333:pii-entities-
detection-job/6f9db0c42d0c810e814670ee4EXAMPLE",
      "JobName": "example-pii-detection-job",
      "JobStatus": "COMPLETED",
      "SubmitTime": "2023-06-09T21:02:46.241000+00:00",
      "EndTime": "2023-06-09T21:12:52.602000+00:00",
      "InputDataConfig": {
        "S3Uri": "s3://DOC-EXAMPLE-BUCKET/AsyncBatchJobs/",
        "InputFormat": "ONE_DOC_PER_LINE"
      },
      "OutputDataConfig": {
        "S3Uri": "s3://DOC-EXAMPLE-SOURCE-BUCKET/111122223333-
PII-6f9db0c42d0c810e814670ee4EXAMPLE/output/"
      },
      "LanguageCode": "en",
      "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role",
      "Mode": "ONLY_OFFSETS"
    },
    {
      "JobId": "d927562638cfa739331a99b3cEXAMPLE",
      "JobArn": "arn:aws:comprehend:us-west-2:111122223333:pii-entities-
detection-job/d927562638cfa739331a99b3cEXAMPLE",
      "JobName": "example-pii-detection-job-2",
      "JobStatus": "COMPLETED",
      "SubmitTime": "2023-06-09T21:20:58.211000+00:00",
      "EndTime": "2023-06-09T21:31:06.027000+00:00",
      "InputDataConfig": {
        "S3Uri": "s3://DOC-EXAMPLE-BUCKET/AsyncBatchJobs/",
        "InputFormat": "ONE_DOC_PER_LINE"
      },
      "OutputDataConfig": {
        "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/
thefolder/111122223333-PII-d927562638cfa739331a99b3cEXAMPLE/output/"
      },
      "LanguageCode": "en",
      "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role",
      "Mode": "ONLY_OFFSETS"
    }
  ]
}
```

```

    }
  ]
}

```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ListPiiEntitiesDetectionJobs](#) in *AWS CLI Command Reference*.

list-sentiment-detection-jobs

The following code example shows how to use `list-sentiment-detection-jobs`.

AWS CLI

To list all sentiment detection jobs

The following `list-sentiment-detection-jobs` example lists all in-progress and completed asynchronous sentiment detection jobs.

```
aws comprehend list-sentiment-detection-jobs
```

Output:

```

{
  "SentimentDetectionJobPropertiesList": [
    {
      "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
      "JobArn": "arn:aws:comprehend:us-west-2:111122223333:sentiment-
detection-job/123456abcdeb0e11022f22a11EXAMPLE",
      "JobName": "example-sentiment-detection-job",
      "JobStatus": "IN_PROGRESS",
      "SubmitTime": "2023-06-09T22:42:20.545000+00:00",
      "EndTime": "2023-06-09T22:52:27.416000+00:00",
      "InputDataConfig": {
        "S3Uri": "s3://DOC-EXAMPLE-BUCKET/MovieData",
        "InputFormat": "ONE_DOC_PER_LINE"
      },
      "OutputDataConfig": {
        "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/
testfolder/111122223333-TS-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
      },
      "LanguageCode": "en",
    }
  ]
}

```



```

        "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
    },
    {
        "JobId": "123456abcdeb0e11022f22a1EXAMPLE2",
        "JobArn": "arn:aws:comprehend:us-west-2:111122223333:sentiment-
detection-job/123456abcdeb0e11022f22a1EXAMPLE2",
        "JobName": "example-sentiment-detection-job-2",
        "JobStatus": "COMPLETED",
        "SubmitTime": "2023-06-09T23:16:15.956000+00:00",
        "EndTime": "2023-06-09T23:26:00.168000+00:00",
        "InputDataConfig": {
            "S3Uri": "s3://DOC-EXAMPLE-BUCKET/MovieData2",
            "InputFormat": "ONE_DOC_PER_LINE"
        },
        "OutputDataConfig": {
            "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/
testfolder/111122223333-TS-123456abcdeb0e11022f22a1EXAMPLE2/output/output.tar.gz"
        },
        "LanguageCode": "en",
        "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
    }
]
}

```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ListSentimentDetectionJobs](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags for resource

The following `list-tags-for-resource` example lists the tags for an Amazon Comprehend resource.

```
aws comprehend list-tags-for-resource \
```

```
--resource-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/  
example-classifier/version/1
```

Output:

```
{  
  "ResourceArn": "arn:aws:comprehend:us-west-2:111122223333:document-classifier/  
example-classifier/version/1",  
  "Tags": [  
    {  
      "Key": "Department",  
      "Value": "Finance"  
    },  
    {  
      "Key": "location",  
      "Value": "Seattle"  
    }  
  ]  
}
```

For more information, see [Tagging your resources](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

list-targeted-sentiment-detection-jobs

The following code example shows how to use `list-targeted-sentiment-detection-jobs`.

AWS CLI

To list all targeted sentiment detection jobs

The following `list-targeted-sentiment-detection-jobs` example lists all in-progress and completed asynchronous targeted sentiment detection jobs.

```
aws comprehend list-targeted-sentiment-detection-jobs
```

Output:

```
{  
  "TargetedSentimentDetectionJobPropertiesList": [  
    {
```

```
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:targeted-sentiment-
detection-job/123456abcdeb0e11022f22a11EXAMPLE",
    "JobName": "example-targeted-sentiment-detection-job",
    "JobStatus": "COMPLETED",
    "SubmitTime": "2023-06-09T22:42:20.545000+00:00",
    "EndTime": "2023-06-09T22:52:27.416000+00:00",
    "InputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/MovieData",
      "InputFormat": "ONE_DOC_PER_LINE"
    },
    "OutputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/
testfolder/111122223333-TS-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
    },
    "LanguageCode": "en",
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-I0role"
  },
  {
    "JobId": "123456abcdeb0e11022f22a11EXAMPLE2",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:targeted-sentiment-
detection-job/123456abcdeb0e11022f22a11EXAMPLE2",
    "JobName": "example-targeted-sentiment-detection-job-2",
    "JobStatus": "COMPLETED",
    "SubmitTime": "2023-06-09T23:16:15.956000+00:00",
    "EndTime": "2023-06-09T23:26:00.168000+00:00",
    "InputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/MovieData2",
      "InputFormat": "ONE_DOC_PER_LINE"
    },
    "OutputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/
testfolder/111122223333-TS-123456abcdeb0e11022f22a11EXAMPLE2/output/output.tar.gz"
    },
    "LanguageCode": "en",
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
  }
]
}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ListTargetedSentimentDetectionJobs](#) in *AWS CLI Command Reference*.

list-topics-detection-jobs

The following code example shows how to use `list-topics-detection-jobs`.

AWS CLI

To list all topic detection jobs

The following `list-topics-detection-jobs` example lists all in-progress and completed asynchronous topics detection jobs.

```
aws comprehend list-topics-detection-jobs
```

Output:

```
{
  "TopicsDetectionJobPropertiesList": [
    {
      "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
      "JobArn": "arn:aws:comprehend:us-west-2:111122223333:topics-detection-job/123456abcdeb0e11022f22a11EXAMPLE",
      "JobName": "topic-analysis-1",
      "JobStatus": "IN_PROGRESS",
      "SubmitTime": "2023-06-09T18:40:35.384000+00:00",
      "EndTime": "2023-06-09T18:46:41.936000+00:00",
      "InputDataConfig": {
        "S3Uri": "s3://DOC-EXAMPLE-BUCKET",
        "InputFormat": "ONE_DOC_PER_LINE"
      },
      "OutputDataConfig": {
        "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/
thefolder/111122223333-TOPICS-123456abcdeb0e11022f22a11EXAMPLE/output/output.tar.gz"
      },
      "NumberOfTopics": 10,
      "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
    },
    {
```

```

    "JobId": "123456abcdeb0e11022f22a1EXAMPLE2",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:topics-detection-
job/123456abcdeb0e11022f22a1EXAMPLE2",
    "JobName": "topic-analysis-2",
    "JobStatus": "COMPLETED",
    "SubmitTime": "2023-06-09T18:44:43.414000+00:00",
    "EndTime": "2023-06-09T18:50:50.872000+00:00",
    "InputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET",
      "InputFormat": "ONE_DOC_PER_LINE"
    },
    "OutputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/
thefolder/111122223333-TOPICS-123456abcdeb0e11022f22a1EXAMPLE2/output/output.tar.gz"
    },
    "NumberOfTopics": 10,
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
  },
  {
    "JobId": "123456abcdeb0e11022f22a1EXAMPLE3",
    "JobArn": "arn:aws:comprehend:us-west-2:111122223333:topics-detection-
job/123456abcdeb0e11022f22a1EXAMPLE3",
    "JobName": "topic-analysis-2",
    "JobStatus": "IN_PROGRESS",
    "SubmitTime": "2023-06-09T18:50:56.737000+00:00",
    "InputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET",
      "InputFormat": "ONE_DOC_PER_LINE"
    },
    "OutputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/
thefolder/111122223333-TOPICS-123456abcdeb0e11022f22a1EXAMPLE3/output/output.tar.gz"
    },
    "NumberOfTopics": 10,
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role"
  }
]
}

```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [ListTopicsDetectionJobs](#) in *AWS CLI Command Reference*.

put-resource-policy

The following code example shows how to use `put-resource-policy`.

AWS CLI

To attach a resource-based policy

The following `put-resource-policy` example attaches a resource-based policy to a model so that can be imported by another AWS account. The policy is attached to the model in account 111122223333 and allows account 444455556666 import the model.

```
aws comprehend put-resource-policy \  
  --resource-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/  
example-classifier/version/1 \  
  --resource-policy '{"Version":"2012-10-17","Statement":  
[{"Effect":"Allow","Action":"comprehend:ImportModel","Resource":"*","Principal":  
{"AWS":["arn:aws:iam::444455556666:root"]}]}'
```

Output:

```
{  
  "PolicyRevisionId": "aaa111d069d07afaa2aa3106aEXAMPLE"  
}
```

For more information, see [Copying custom models between AWS accounts](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [PutResourcePolicy](#) in *AWS CLI Command Reference*.

start-document-classification-job

The following code example shows how to use `start-document-classification-job`.

AWS CLI

To start document classification job

The following `start-document-classification-job` example starts a document classification job with a custom model on all of the files at the address specified

by the `--input-data-config` tag. In this example, the input S3 bucket contains `SampleSMStext1.txt`, `SampleSMStext2.txt`, and `SampleSMStext3.txt`. The model was previously trained on document classifications of spam and non-spam, or, "ham", SMS messages. When the job is complete, `output.tar.gz` is put at the location specified by the `--output-data-config` tag. `output.tar.gz` contains `predictions.jsonl` which lists the classification of each document. The Json output is printed on one line per file, but is formatted here for readability.

```
aws comprehend start-document-classification-job \  
  --job-name exampleclassificationjob \  
  --input-data-config "S3Uri=s3://DOC-EXAMPLE-BUCKET-INPUT/jobdata/" \  
  --output-data-config "S3Uri=s3://DOC-EXAMPLE-DESTINATION-BUCKET/testfolder/" \  
  --data-access-role-arn arn:aws:iam::111122223333:role/service-role/  
AmazonComprehendServiceRole-example-role \  
  --document-classifier-arn arn:aws:comprehend:us-west-2:111122223333:document-  
classifier/mymodel/version/12
```

Contents of `SampleSMStext1.txt`:

```
"CONGRATULATIONS! TXT 2155550100 to win $5000"
```

Contents of `SampleSMStext2.txt`:

```
"Hi, when do you want me to pick you up from practice?"
```

Contents of `SampleSMStext3.txt`:

```
"Plz send bank account # to 2155550100 to claim prize!!"
```

Output:

```
{  
  "JobId": "e758dd56b824aa717ceab551fEXAMPLE",  
  "JobArn": "arn:aws:comprehend:us-west-2:111122223333:document-classification-  
job/e758dd56b824aa717ceab551fEXAMPLE",  
  "JobStatus": "SUBMITTED"  
}
```

Contents of `predictions.jsonl`:

```

{"File": "SampleSMSText1.txt", "Line": "0", "Classes": [{"Name": "spam", "Score": 0.9999}, {"Name": "ham", "Score": 0.0001}]}
{"File": "SampleSMStext2.txt", "Line": "0", "Classes": [{"Name": "ham", "Score": 0.9994}, {"Name": "spam", "Score": 0.0006}]}
{"File": "SampleSMSText3.txt", "Line": "0", "Classes": [{"Name": "spam", "Score": 0.9999}, {"Name": "ham", "Score": 0.0001}]}

```

For more information, see [Custom Classification](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StartDocumentClassificationJob](#) in *AWS CLI Command Reference*.

start-dominant-language-detection-job

The following code example shows how to use `start-dominant-language-detection-job`.

AWS CLI

To start an asynchronous language detection job

The following `start-dominant-language-detection-job` example starts an asynchronous language detection job for all of the files located at the address specified by the `--input-data-config` tag. The S3 bucket in this example contains `Sampletext1.txt`. When the job is complete, the folder, `output`, is placed in the location specified by the `--output-data-config` tag. The folder contains `output.txt` which contains the dominant language of each of the text files as well as the pre-trained model's confidence score for each prediction.

```

aws comprehend start-dominant-language-detection-job \
  --job-name example_language_analysis_job \
  --language-code en \
  --input-data-config "S3Uri=s3://DOC-EXAMPLE-BUCKET/" \
  --output-data-config "S3Uri=s3://DOC-EXAMPLE-DESTINATION-BUCKET/testfolder/" \
  --data-access-role-arn arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role \
  --language-code en

```

Contents of `Sampletext1.txt`:

```

"Physics is the natural science that involves the study of matter and its motion and
behavior through space and time, along with related concepts such as energy and
force."

```


Output:

```
{
  "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
  "JobArn": "arn:aws:comprehend:us-west-2:111122223333:dominant-language-
detection-job/123456abcdeb0e11022f22a11EXAMPLE",
  "JobStatus": "SUBMITTED"
}
```

Contents of output.txt:

```
{"File": "Sampletext1.txt", "Languages": [{"LanguageCode": "en", "Score":
0.9913753867149353}], "Line": 0}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StartDominantLanguageDetectionJob](#) in *AWS CLI Command Reference*.

start-entities-detection-job

The following code example shows how to use `start-entities-detection-job`.

AWS CLI

Example 1: To start a standard entity detection job using the pre-trained model

The following `start-entities-detection-job` example starts an asynchronous entities detection job for all files located at the address specified by the `--input-data-config` tag. The S3 bucket in this example contains `Sampletext1.txt`, `Sampletext2.txt`, and `Sampletext3.txt`. When the job is complete, the folder, `output`, is placed in the location specified by the `--output-data-config` tag. The folder contains `output.txt` which lists all of the named entities detected within each text file as well as the pre-trained model's confidence score for each prediction. The Json output is printed on one line per input file, but is formatted here for readability.

```
aws comprehend start-entities-detection-job \
  --job-name entitiestest \
  --language-code en \
  --input-data-config "S3Uri=s3://DOC-EXAMPLE-BUCKET/" \
  --output-data-config "S3Uri=s3://DOC-EXAMPLE-DESTINATION-BUCKET/testfolder/" \
```

```
--data-access-role-arn arn:aws:iam::111122223333:role/service-role/  
AmazonComprehendServiceRole-example-role \  
--language-code en
```

Contents of Sampletext1.txt:

```
"Hello Zhang Wei, I am John. Your AnyCompany Financial Services, LLC credit card  
account 1111-XXXX-1111-XXXX has a minimum payment of $24.53 that is due by July  
31st."
```

Contents of Sampletext2.txt:

```
"Dear Max, based on your autopay settings for your account example1.org account, we  
will withdraw your payment on the due date from your bank account number XXXXXX1111  
with the routing number XXXXX0000. "
```

Contents of Sampletext3.txt:

```
"Jane, please submit any customer feedback from this weekend to AnySpa, 123 Main St,  
Anywhere and send comments to Alice at AnySpa@example.com."
```

Output:

```
{  
  "JobId": "123456abcdeb0e11022f22a11EXAMPLE",  
  "JobArn": "arn:aws:comprehend:us-west-2:111122223333:entities-detection-  
job/123456abcdeb0e11022f22a11EXAMPLE",  
  "JobStatus": "SUBMITTED"  
}
```

Contents of output.txt with line indents for readability:

```
{  
  "Entities": [  
    {  
      "BeginOffset": 6,  
      "EndOffset": 15,  
      "Score": 0.9994006636420306,  
      "Text": "Zhang Wei",  
      "Type": "PERSON"  
    },  
  ],  
}
```

```
{
  "BeginOffset": 22,
  "EndOffset": 26,
  "Score": 0.9976647915128143,
  "Text": "John",
  "Type": "PERSON"
},
{
  "BeginOffset": 33,
  "EndOffset": 67,
  "Score": 0.9984608700836206,
  "Text": "AnyCompany Financial Services, LLC",
  "Type": "ORGANIZATION"
},
{
  "BeginOffset": 88,
  "EndOffset": 107,
  "Score": 0.9868521019555556,
  "Text": "1111-XXXX-1111-XXXX",
  "Type": "OTHER"
},
{
  "BeginOffset": 133,
  "EndOffset": 139,
  "Score": 0.998242565709204,
  "Text": "$24.53",
  "Type": "QUANTITY"
},
{
  "BeginOffset": 155,
  "EndOffset": 164,
  "Score": 0.9993039263159287,
  "Text": "July 31st",
  "Type": "DATE"
}
],
"File": "SampleText1.txt",
"Line": 0
}
{
  "Entities": [
    {
      "BeginOffset": 5,
      "EndOffset": 8,
```

```
"Score": 0.9866232147545232,
"Text": "Max",
"Type": "PERSON"
},
{
  "BeginOffset": 156,
  "EndOffset": 166,
  "Score": 0.9797723450933329,
  "Text": "XXXXXX1111",
  "Type": "OTHER"
},
{
  "BeginOffset": 191,
  "EndOffset": 200,
  "Score": 0.9247838572396843,
  "Text": "XXXXX0000",
  "Type": "OTHER"
}
],
"File": "SampleText2.txt",
"Line": 0
}
{
  "Entities": [
    {
      "Score": 0.9990532994270325,
      "Type": "PERSON",
      "Text": "Jane",
      "BeginOffset": 0,
      "EndOffset": 4
    },
    {
      "Score": 0.9519651532173157,
      "Type": "DATE",
      "Text": "this weekend",
      "BeginOffset": 47,
      "EndOffset": 59
    },
    {
      "Score": 0.5566426515579224,
      "Type": "ORGANIZATION",
      "Text": "AnySpa",
      "BeginOffset": 63,
      "EndOffset": 69
    }
  ]
}
```

```
  },
  {
    "Score": 0.8059805631637573,
    "Type": "LOCATION",
    "Text": "123 Main St, Anywhere",
    "BeginOffset": 71,
    "EndOffset": 92
  },
  {
    "Score": 0.998830258846283,
    "Type": "PERSON",
    "Text": "Alice",
    "BeginOffset": 114,
    "EndOffset": 119
  },
  {
    "Score": 0.997818112373352,
    "Type": "OTHER",
    "Text": "AnySpa@example.com",
    "BeginOffset": 123,
    "EndOffset": 138
  }
],
"File": "SampleText3.txt",
"Line": 0
}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

Example 2: To start a custom entity detection job

The following `start-entities-detection-job` example starts an asynchronous custom entities detection job for all files located at the address specified by the `--input-data-config` tag. In this example, the S3 bucket in this example contains `SampleFeedback1.txt`, `SampleFeedback2.txt`, and `SampleFeedback3.txt`. The entity recognizer model was trained on customer support Feedbacks to recognize device names. When the job is complete, an the folder, `output`, is put at the location specified by the `--output-data-config` tag. The folder contains `output.txt`, which lists all of the named entities detected within each text file as well as the pre-trained model's confidence score for each prediction. The Json output is printed on one line per file, but is formatted here for readability.

```
aws comprehend start-entities-detection-job \  
  --job-name customentitiestest \  
  --entity-recognizer-arn "arn:aws:comprehend:us-west-2:111122223333:entity-recognizer/entityrecognizer" \  
  --language-code en \  
  --input-data-config "S3Uri=s3://DOC-EXAMPLE-BUCKET/jobdata/" \  
  --output-data-config "S3Uri=s3://DOC-EXAMPLE-DESTINATION-BUCKET/testfolder/" \  
  --data-access-role-arn "arn:aws:iam::111122223333:role/service-role/AmazonComprehendServiceRole-I0role"
```

Contents of SampleFeedback1.txt:

```
"I've been on the AnyPhone app have had issues for 24 hours when trying to pay bill. Cannot make payment. Sigh. | Oh man! Lets get that app up and running. DM me, and we can get to work!"
```

Contents of SampleFeedback2.txt:

```
"Hi, I have a discrepancy with my new bill. Could we get it sorted out? A rep added stuff I didnt sign up for when I did my AnyPhone 10 upgrade. | We can absolutely get this sorted!"
```

Contents of SampleFeedback3.txt:

```
"Is the by 1 get 1 free AnySmartPhone promo still going on? | Hi Christian! It ended yesterday, send us a DM if you have any questions and we can take a look at your options!"
```

Output:

```
{  
  "JobId": "019ea9edac758806850fa8a79ff83021",  
  "JobArn": "arn:aws:comprehend:us-west-2:111122223333:entities-detection-job/019ea9edac758806850fa8a79ff83021",  
  "JobStatus": "SUBMITTED"  
}
```

Contents of output.txt with line indents for readability:

```
{
```

```
"Entities": [  
  {  
    "BeginOffset": 17,  
    "EndOffset": 25,  
    "Score": 0.9999728210205924,  
    "Text": "AnyPhone",  
    "Type": "DEVICE"  
  }  
],  
"File": "SampleFeedback1.txt",  
"Line": 0  
}  
{  
"Entities": [  
  {  
    "BeginOffset": 123,  
    "EndOffset": 133,  
    "Score": 0.9999892116761524,  
    "Text": "AnyPhone 10",  
    "Type": "DEVICE"  
  }  
],  
"File": "SampleFeedback2.txt",  
"Line": 0  
}  
{  
"Entities": [  
  {  
    "BeginOffset": 23,  
    "EndOffset": 35,  
    "Score": 0.9999971389852362,  
    "Text": "AnySmartPhone",  
    "Type": "DEVICE"  
  }  
],  
"File": "SampleFeedback3.txt",  
"Line": 0  
}
```

For more information, see [Custom entity recognition](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StartEntitiesDetectionJob](#) in *AWS CLI Command Reference*.

start-events-detection-job

The following code example shows how to use `start-events-detection-job`.

AWS CLI

To start an asynchronous events detection job

The following `start-events-detection-job` example starts an asynchronous events detection job for all files located at the address specified by the `--input-data-config` tag. Possible target event types include `BANKRUPTCY`, `EMPLOYMENT`, `CORPORATE_ACQUISITION`, `INVESTMENT_GENERAL`, `CORPORATE_MERGER`, `IPO`, `RIGHTS_ISSUE`, `SECONDARY_OFFERING`, `SHELF_OFFERING`, `TENDER_OFFERING`, and `STOCK_SPLIT`. The S3 bucket in this example contains `SampleText1.txt`, `SampleText2.txt`, and `SampleText3.txt`. When the job is complete, the folder, `output`, is placed in the location specified by the `--output-data-config` tag. The folder contains `SampleText1.txt.out`, `SampleText2.txt.out`, and `SampleText3.txt.out`. The JSON output is printed on one line per file, but is formatted here for readability.

```
aws comprehend start-events-detection-job \  
  --job-name events-detection-1 \  
  --input-data-config "S3Uri=s3://DOC-EXAMPLE-BUCKET/EventsData" \  
  --output-data-config "S3Uri=s3://DOC-EXAMPLE-DESTINATION-BUCKET/testfolder/" \  
  --data-access-role-arn arn:aws:iam::111122223333:role/service-role/  
AmazonComprehendServiceRole-servicerole \  
  --language-code en \  
  --target-event-types "BANKRUPTCY" "EMPLOYMENT" "CORPORATE_ACQUISITION"  
"CORPORATE_MERGER" "INVESTMENT_GENERAL"
```

Contents of `SampleText1.txt`:

```
"Company AnyCompany grew by increasing sales and through acquisitions. After  
purchasing competing firms in 2020, AnyBusiness, a part of the AnyBusinessGroup,  
gave Jane Does firm a going rate of one cent a gallon or forty-two cents a barrel."
```

Contents of `SampleText2.txt`:

```
"In 2021, AnyCompany officially purchased AnyBusiness for 100 billion dollars,  
surprising and exciting the shareholders."
```

Contents of `SampleText3.txt`:


```
"In 2022, AnyCompany stock crashed 50. Eventually later that year they filed for bankruptcy."
```

Output:

```
{
  "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
  "JobArn": "arn:aws:comprehend:us-west-2:111122223333:events-detection-job/123456abcdeb0e11022f22a11EXAMPLE",
  "JobStatus": "SUBMITTED"
}
```

Contents of SampleText1.txt.out with line indents for readability:

```
{
  "Entities": [
    {
      "Mentions": [
        {
          "BeginOffset": 8,
          "EndOffset": 18,
          "Score": 0.99977,
          "Text": "AnyCompany",
          "Type": "ORGANIZATION",
          "GroupScore": 1
        },
        {
          "BeginOffset": 112,
          "EndOffset": 123,
          "Score": 0.999747,
          "Text": "AnyBusiness",
          "Type": "ORGANIZATION",
          "GroupScore": 0.979826
        },
        {
          "BeginOffset": 171,
          "EndOffset": 175,
          "Score": 0.999615,
          "Text": "firm",
          "Type": "ORGANIZATION",
          "GroupScore": 0.871647
        }
      ]
    }
  ]
}
```

```
]
},
{
  "Mentions": [
    {
      "BeginOffset": 97,
      "EndOffset": 102,
      "Score": 0.987687,
      "Text": "firms",
      "Type": "ORGANIZATION",
      "GroupScore": 1
    }
  ]
},
{
  "Mentions": [
    {
      "BeginOffset": 103,
      "EndOffset": 110,
      "Score": 0.999458,
      "Text": "in 2020",
      "Type": "DATE",
      "GroupScore": 1
    }
  ]
},
{
  "Mentions": [
    {
      "BeginOffset": 160,
      "EndOffset": 168,
      "Score": 0.999649,
      "Text": "John Doe",
      "Type": "PERSON",
      "GroupScore": 1
    }
  ]
}
],
"Events": [
  {
    "Type": "CORPORATE_ACQUISITION",
    "Arguments": [
      {
```

```
    "EntityIndex": 0,  
    "Role": "INVESTOR",  
    "Score": 0.99977  
  }  
],  
"Triggers": [  
  {  
    "BeginOffset": 56,  
    "EndOffset": 68,  
    "Score": 0.999967,  
    "Text": "acquisitions",  
    "Type": "CORPORATE_ACQUISITION",  
    "GroupScore": 1  
  }  
]  
},  
{  
  "Type": "CORPORATE_ACQUISITION",  
  "Arguments": [  
    {  
      "EntityIndex": 1,  
      "Role": "INVESTEES",  
      "Score": 0.987687  
    },  
    {  
      "EntityIndex": 2,  
      "Role": "DATE",  
      "Score": 0.999458  
    },  
    {  
      "EntityIndex": 3,  
      "Role": "INVESTOR",  
      "Score": 0.999649  
    }  
  ],  
  "Triggers": [  
    {  
      "BeginOffset": 76,  
      "EndOffset": 86,  
      "Score": 0.999973,  
      "Text": "purchasing",  
      "Type": "CORPORATE_ACQUISITION",  
      "GroupScore": 1  
    }  
  ]  
}
```

```
    ]
  }
],
"File": "SampleText1.txt",
"Line": 0
}
```

Contents of SampleText2.txt.out:

```
{
  "Entities": [
    {
      "Mentions": [
        {
          "BeginOffset": 0,
          "EndOffset": 7,
          "Score": 0.999473,
          "Text": "In 2021",
          "Type": "DATE",
          "GroupScore": 1
        }
      ]
    },
    {
      "Mentions": [
        {
          "BeginOffset": 9,
          "EndOffset": 19,
          "Score": 0.999636,
          "Text": "AnyCompany",
          "Type": "ORGANIZATION",
          "GroupScore": 1
        }
      ]
    },
    {
      "Mentions": [
        {
          "BeginOffset": 45,
          "EndOffset": 56,
          "Score": 0.999712,
          "Text": "AnyBusiness",
          "Type": "ORGANIZATION",

```

```
        "GroupScore": 1
      }
    ]
  },
  {
    "Mentions": [
      {
        "BeginOffset": 61,
        "EndOffset": 80,
        "Score": 0.998886,
        "Text": "100 billion dollars",
        "Type": "MONETARY_VALUE",
        "GroupScore": 1
      }
    ]
  }
],
"Events": [
  {
    "Type": "CORPORATE_ACQUISITION",
    "Arguments": [
      {
        "EntityIndex": 3,
        "Role": "AMOUNT",
        "Score": 0.998886
      },
      {
        "EntityIndex": 2,
        "Role": "INVESTEES",
        "Score": 0.999712
      },
      {
        "EntityIndex": 0,
        "Role": "DATE",
        "Score": 0.999473
      },
      {
        "EntityIndex": 1,
        "Role": "INVESTOR",
        "Score": 0.999636
      }
    ]
  },
  {
    "Type": "CORPORATE_ACQUISITION",
    "Arguments": [
      {
        "EntityIndex": 3,
        "Role": "AMOUNT",
        "Score": 0.998886
      },
      {
        "EntityIndex": 2,
        "Role": "INVESTEES",
        "Score": 0.999712
      },
      {
        "EntityIndex": 0,
        "Role": "DATE",
        "Score": 0.999473
      },
      {
        "EntityIndex": 1,
        "Role": "INVESTOR",
        "Score": 0.999636
      }
    ]
  }
],
"Triggers": [
  {
```

```
        "BeginOffset": 31,
        "EndOffset": 40,
        "Score": 0.99995,
        "Text": "purchased",
        "Type": "CORPORATE_ACQUISITION",
        "GroupScore": 1
      }
    ]
  }
],
"File": "SampleText2.txt",
"Line": 0
}
```

Contents of SampleText3.txt.out:

```
{
  "Entities": [
    {
      "Mentions": [
        {
          "BeginOffset": 9,
          "EndOffset": 19,
          "Score": 0.999774,
          "Text": "AnyCompany",
          "Type": "ORGANIZATION",
          "GroupScore": 1
        },
        {
          "BeginOffset": 66,
          "EndOffset": 70,
          "Score": 0.995717,
          "Text": "they",
          "Type": "ORGANIZATION",
          "GroupScore": 0.997626
        }
      ]
    },
    {
      "Mentions": [
        {
          "BeginOffset": 50,
          "EndOffset": 65,
```

```
        "Score": 0.999656,
        "Text": "later that year",
        "Type": "DATE",
        "GroupScore": 1
      }
    ]
  }
],
"Events": [
  {
    "Type": "BANKRUPTCY",
    "Arguments": [
      {
        "EntityIndex": 1,
        "Role": "DATE",
        "Score": 0.999656
      },
      {
        "EntityIndex": 0,
        "Role": "FILER",
        "Score": 0.995717
      }
    ],
    "Triggers": [
      {
        "BeginOffset": 81,
        "EndOffset": 91,
        "Score": 0.999936,
        "Text": "bankruptcy",
        "Type": "BANKRUPTCY",
        "GroupScore": 1
      }
    ]
  }
],
"File": "SampleText3.txt",
"Line": 0
}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StartEventsDetectionJob](#) in *AWS CLI Command Reference*.

start-flywheel-iteration

The following code example shows how to use `start-flywheel-iteration`.

AWS CLI

To start a flywheel iteration

The following `start-flywheel-iteration` example starts a flywheel iteration. This operation uses any new datasets in the flywheel to train a new model version.

```
aws comprehend start-flywheel-iteration \  
  --flywheel-arn arn:aws:comprehend:us-west-2:111122223333:flywheel/example-  
flywheel
```

Output:

```
{  
  "FlywheelArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/example-  
flywheel",  
  "FlywheelIterationId": "12345123TEXAMPLE"  
}
```

For more information, see [Flywheel overview](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StartFlywheelIteration](#) in *AWS CLI Command Reference*.

start-key-phrases-detection-job

The following code example shows how to use `start-key-phrases-detection-job`.

AWS CLI

To start a key phrases detection job

The following `start-key-phrases-detection-job` example starts an asynchronous key phrases detection job for all files located at the address specified by the `--input-data-config` tag. The S3 bucket in this example contains `Sampletext1.txt`, `Sampletext2.txt`, and `Sampletext3.txt`. When the job is completed, the folder, `output`, is placed in the location specified by the `--output-data-config` tag. The folder contains the file `output.txt` which contains all the key phrases detected within each text file and the pre-

trained model's confidence score for each prediction. The Json output is printed on one line per file, but is formatted here for readability.

```
aws comprehend start-key-phrases-detection-job \  
  --job-name keyphrasesanalysistest1 \  
  --language-code en \  
  --input-data-config "S3Uri=s3://DOC-EXAMPLE-BUCKET/" \  
  --output-data-config "S3Uri=s3://DOC-EXAMPLE-DESTINATION-BUCKET/testfolder/" \  
  --data-access-role-arn "arn:aws:iam::111122223333:role/service-role/  
AmazonComprehendServiceRole-example-role" \  
  --language-code en
```

Contents of Sampletext1.txt:

```
"Hello Zhang Wei, I am John. Your AnyCompany Financial Services, LLC credit card  
account 1111-XXXX-1111-XXXX has a minimum payment of $24.53 that is due by July  
31st."
```

Contents of Sampletext2.txt:

```
"Dear Max, based on your autopay settings for your account Internet.org account, we  
will withdraw your payment on the due date from your bank account number XXXXXX1111  
with the routing number XXXXX0000. "
```

Contents of Sampletext3.txt:

```
"Jane, please submit any customer feedback from this weekend to Sunshine Spa, 123  
Main St, Anywhere and send comments to Alice at AnySpa@example.com."
```

Output:

```
{  
  "JobId": "123456abcdeb0e11022f22a11EXAMPLE",  
  "JobArn": "arn:aws:comprehend:us-west-2:111122223333:key-phrases-detection-  
job/123456abcdeb0e11022f22a11EXAMPLE",  
  "JobStatus": "SUBMITTED"  
}
```

Contents of output.txt with line indents for readability:

```
{
```

```
"File": "SampleText1.txt",
"KeyPhrases": [
  {
    "BeginOffset": 6,
    "EndOffset": 15,
    "Score": 0.9748965572679326,
    "Text": "Zhang Wei"
  },
  {
    "BeginOffset": 22,
    "EndOffset": 26,
    "Score": 0.9997344722354619,
    "Text": "John"
  },
  {
    "BeginOffset": 28,
    "EndOffset": 62,
    "Score": 0.9843791074032948,
    "Text": "Your AnyCompany Financial Services"
  },
  {
    "BeginOffset": 64,
    "EndOffset": 107,
    "Score": 0.8976122401721824,
    "Text": "LLC credit card account 1111-XXXX-1111-XXXX"
  },
  {
    "BeginOffset": 112,
    "EndOffset": 129,
    "Score": 0.9999612982629748,
    "Text": "a minimum payment"
  },
  {
    "BeginOffset": 133,
    "EndOffset": 139,
    "Score": 0.99975728947036,
    "Text": "$24.53"
  },
  {
    "BeginOffset": 155,
    "EndOffset": 164,
    "Score": 0.9940866241449973,
    "Text": "July 31st"
  }
]
```

```
],
"Line": 0
}
{
"File": "SampleText2.txt",
"KeyPhrases": [
  {
    "BeginOffset": 0,
    "EndOffset": 8,
    "Score": 0.9974021100118472,
    "Text": "Dear Max"
  },
  {
    "BeginOffset": 19,
    "EndOffset": 40,
    "Score": 0.9961120519515884,
    "Text": "your autopay settings"
  },
  {
    "BeginOffset": 45,
    "EndOffset": 78,
    "Score": 0.9980620070116009,
    "Text": "your account Internet.org account"
  },
  {
    "BeginOffset": 97,
    "EndOffset": 109,
    "Score": 0.999919660140754,
    "Text": "your payment"
  },
  {
    "BeginOffset": 113,
    "EndOffset": 125,
    "Score": 0.9998370719754205,
    "Text": "the due date"
  },
  {
    "BeginOffset": 131,
    "EndOffset": 166,
    "Score": 0.9955068678502509,
    "Text": "your bank account number XXXXXX1111"
  },
  {
    "BeginOffset": 172,
```

```
    "EndOffset": 200,
    "Score": 0.8653433315829526,
    "Text": "the routing number XXXXX0000"
  }
],
"Line": 0
}
{
"File": "SampleText3.txt",
"KeyPhrases": [
  {
    "BeginOffset": 0,
    "EndOffset": 4,
    "Score": 0.9142947833681668,
    "Text": "Jane"
  },
  {
    "BeginOffset": 20,
    "EndOffset": 41,
    "Score": 0.9984325676596763,
    "Text": "any customer feedback"
  },
  {
    "BeginOffset": 47,
    "EndOffset": 59,
    "Score": 0.9998782448150636,
    "Text": "this weekend"
  },
  {
    "BeginOffset": 63,
    "EndOffset": 75,
    "Score": 0.99866741830757,
    "Text": "Sunshine Spa"
  },
  {
    "BeginOffset": 77,
    "EndOffset": 88,
    "Score": 0.9695803485466054,
    "Text": "123 Main St"
  },
  {
    "BeginOffset": 108,
    "EndOffset": 116,
    "Score": 0.9997065928550928,
```

```
    "Text": "comments"
  },
  {
    "BeginOffset": 120,
    "EndOffset": 125,
    "Score": 0.9993466833825161,
    "Text": "Alice"
  },
  {
    "BeginOffset": 129,
    "EndOffset": 144,
    "Score": 0.9654563612885667,
    "Text": "AnySpa@example.com"
  }
],
"Line": 0
}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StartKeyPhrasesDetectionJob](#) in *AWS CLI Command Reference*.

start-pii-entities-detection-job

The following code example shows how to use `start-pii-entities-detection-job`.

AWS CLI

To start an asynchronous PII detection job

The following `start-pii-entities-detection-job` example starts an asynchronous personal identifiable information (PII) entities detection job for all files located at the address specified by the `--input-data-config` tag. The S3 bucket in this example contains `Sampletext1.txt`, `Sampletext2.txt`, and `Sampletext3.txt`. When the job is complete, the folder, `output`, is placed in the location specified by the `--output-data-config` tag. The folder contains `SampleText1.txt.out`, `SampleText2.txt.out`, and `SampleText3.txt.out` which list the named entities within each text file. The Json output is printed on one line per file, but is formatted here for readability.

```
aws comprehend start-pii-entities-detection-job \
  --job-name entities_test \
```

```
--language-code en \  
--input-data-config "S3Uri=s3://DOC-EXAMPLE-BUCKET/" \  
--output-data-config "S3Uri=s3://DOC-EXAMPLE-DESTINATION-BUCKET/testfolder/" \  
--data-access-role-arn arn:aws:iam::111122223333:role/service-role/  
AmazonComprehendServiceRole-example-role \  
--language-code en \  
--mode ONLY_OFFSETS
```

Contents of Sampletext1.txt:

```
"Hello Zhang Wei, I am John. Your AnyCompany Financial Services, LLC credit card  
account 1111-XXXX-1111-XXXX has a minimum payment of $24.53 that is due by July  
31st."
```

Contents of Sampletext2.txt:

```
"Dear Max, based on your autopay settings for your account Internet.org account, we  
will withdraw your payment on the due date from your bank account number XXXXXX1111  
with the routing number XXXXX0000. "
```

Contents of Sampletext3.txt:

```
"Jane, please submit any customer feedback from this weekend to Sunshine Spa, 123  
Main St, Anywhere and send comments to Alice at AnySpa@example.com."
```

Output:

```
{  
  "JobId": "123456abcdeb0e11022f22a11EXAMPLE",  
  "JobArn": "arn:aws:comprehend:us-west-2:111122223333:pii-entities-detection-  
job/123456abcdeb0e11022f22a11EXAMPLE",  
  "JobStatus": "SUBMITTED"  
}
```

Contents of SampleText1.txt.out with line indents for readability:

```
{  
  "Entities": [  
    {  
      "BeginOffset": 6,
```

```

    "EndOffset": 15,
    "Type": "NAME",
    "Score": 0.9998490510222595
  },
  {
    "BeginOffset": 22,
    "EndOffset": 26,
    "Type": "NAME",
    "Score": 0.9998937958019426
  },
  {
    "BeginOffset": 88,
    "EndOffset": 107,
    "Type": "CREDIT_DEBIT_NUMBER",
    "Score": 0.9554297245278491
  },
  {
    "BeginOffset": 155,
    "EndOffset": 164,
    "Type": "DATE_TIME",
    "Score": 0.9999720462925257
  }
],
"File": "SampleText1.txt",
"Line": 0
}

```

Contents of SampleText2.txt.out with line indents for readability:

```

{
  "Entities": [
    {
      "BeginOffset": 5,
      "EndOffset": 8,
      "Type": "NAME",
      "Score": 0.9994390774924007
    },
    {
      "BeginOffset": 58,
      "EndOffset": 70,
      "Type": "URL",
      "Score": 0.9999958276922101
    },
  ],
}

```

```
{
  "BeginOffset": 156,
  "EndOffset": 166,
  "Type": "BANK_ACCOUNT_NUMBER",
  "Score": 0.9999721058045592
},
{
  "BeginOffset": 191,
  "EndOffset": 200,
  "Type": "BANK_ROUTING",
  "Score": 0.9998968945989909
}
],
"File": "SampleText2.txt",
"Line": 0
}
```

Contents of SampleText3.txt.out with line indents for readability:

```
{
  "Entities": [
    {
      "BeginOffset": 0,
      "EndOffset": 4,
      "Type": "NAME",
      "Score": 0.999949934606805
    },
    {
      "BeginOffset": 77,
      "EndOffset": 88,
      "Type": "ADDRESS",
      "Score": 0.9999035300466904
    },
    {
      "BeginOffset": 120,
      "EndOffset": 125,
      "Type": "NAME",
      "Score": 0.9998203838716296
    },
    {
      "BeginOffset": 129,
      "EndOffset": 144,
      "Type": "EMAIL",

```



```
    "Score": 0.9998313473105228
  }
],
"File": "SampleText3.txt",
"Line": 0
}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StartPiiEntitiesDetectionJob](#) in *AWS CLI Command Reference*.

start-sentiment-detection-job

The following code example shows how to use `start-sentiment-detection-job`.

AWS CLI

To start an asynchronous sentiment analysis job

The following `start-sentiment-detection-job` example starts an asynchronous sentiment analysis detection job for all files located at the address specified by the `--input-data-config` tag. The S3 bucket folder in this example contains `SampleMovieReview1.txt`, `SampleMovieReview2.txt`, and `SampleMovieReview3.txt`. When the job is complete, the folder, `output`, is placed at the location specified by the `--output-data-config` tag. The folder contains the file, `output.txt`, which contains the prevailing sentiments for each text file and the pre-trained model's confidence score for each prediction. The Json output is printed on one line per file, but is formatted here for readability.

```
aws comprehend start-sentiment-detection-job \
  --job-name example-sentiment-detection-job \
  --language-code en \
  --input-data-config "S3Uri=s3://DOC-EXAMPLE-BUCKET/MovieData" \
  --output-data-config "S3Uri=s3://DOC-EXAMPLE-DESTINATION-BUCKET/testfolder/" \
  --data-access-role-arn arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role
```

Contents of `SampleMovieReview1.txt`:

```
"The film, AnyMovie2, is fairly predictable and just okay."
```

Contents of SampleMovieReview2.txt:

```
"AnyMovie2 is the essential sci-fi film that I grew up watching when I was a kid. I highly recommend this movie."
```

Contents of SampleMovieReview3.txt:

```
"Don't get fooled by the 'awards' for AnyMovie2. All parts of the film were poorly stolen from other modern directors."
```

Output:

```
{
  "JobId": "0b5001e25f62ebb40631a9a1a7fde7b3",
  "JobArn": "arn:aws:comprehend:us-west-2:111122223333:sentiment-detection-job/0b5001e25f62ebb40631a9a1a7fde7b3",
  "JobStatus": "SUBMITTED"
}
```

Contents of output.txt with line of indents for readability:

```
{
  "File": "SampleMovieReview1.txt",
  "Line": 0,
  "Sentiment": "MIXED",
  "SentimentScore": {
    "Mixed": 0.6591159105300903,
    "Negative": 0.26492202281951904,
    "Neutral": 0.035430654883384705,
    "Positive": 0.04053137078881264
  }
}
{
  "File": "SampleMovieReview2.txt",
  "Line": 0,
  "Sentiment": "POSITIVE",
  "SentimentScore": {
    "Mixed": 0.000008718466233403888,
    "Negative": 0.00006134175055194646,
    "Neutral": 0.0002941041602753103,
    "Positive": 0.9996358156204224
  }
}
```

```
    }
  {
    "File": "SampleMovieReview3.txt",
    "Line": 0,
    "Sentiment": "NEGATIVE",
    "SentimentScore": {
      "Mixed": 0.004146667663007975,
      "Negative": 0.9645107984542847,
      "Neutral": 0.016559595242142677,
      "Positive": 0.014782938174903393
    }
  }
}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StartSentimentDetectionJob](#) in *AWS CLI Command Reference*.

start-targeted-sentiment-detection-job

The following code example shows how to use `start-targeted-sentiment-detection-job`.

AWS CLI

To start an asynchronous targeted sentiment analysis job

The following `start-targeted-sentiment-detection-job` example starts an asynchronous targeted sentiment analysis detection job for all files located at the address specified by the `--input-data-config` tag. The S3 bucket folder in this example contains `SampleMovieReview1.txt`, `SampleMovieReview2.txt`, and `SampleMovieReview3.txt`. When the job is complete, `output.tar.gz` is placed at the location specified by the `--output-data-config` tag. `output.tar.gz` contains the files `SampleMovieReview1.txt.out`, `SampleMovieReview2.txt.out`, and `SampleMovieReview3.txt.out`, which each contain all of the named entities and associated sentiments for a single input text file.

```
aws comprehend start-targeted-sentiment-detection-job \
  --job-name targeted_movie_review_analysis1 \
  --language-code en \
  --input-data-config "S3Uri=s3://DOC-EXAMPLE-BUCKET/MovieData" \
  --output-data-config "S3Uri=s3://DOC-EXAMPLE-DESTINATION-BUCKET/testfolder/" \
```

```
--data-access-role-arn arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role
```

Contents of SampleMovieReview1.txt:

```
"The film, AnyMovie, is fairly predictable and just okay."
```

Contents of SampleMovieReview2.txt:

```
"AnyMovie is the essential sci-fi film that I grew up watching when I was a kid. I
highly recommend this movie."
```

Contents of SampleMovieReview3.txt:

```
"Don't get fooled by the 'awards' for AnyMovie. All parts of the film were poorly
stolen from other modern directors."
```

Output:

```
{
  "JobId": "0b5001e25f62ebb40631a9a1a7fde7b3",
  "JobArn": "arn:aws:comprehend:us-west-2:111122223333:targeted-sentiment-
detection-job/0b5001e25f62ebb40631a9a1a7fde7b3",
  "JobStatus": "SUBMITTED"
}
```

Contents of SampleMovieReview1.txt.out with line indents for readability:

```
{
  "Entities": [
    {
      "DescriptiveMentionIndex": [
        0
      ],
      "Mentions": [
        {
          "BeginOffset": 4,
          "EndOffset": 8,
          "Score": 0.994972,
          "GroupScore": 1,
          "Text": "film",

```

```

        "Type": "MOVIE",
        "MentionSentiment": {
            "Sentiment": "NEUTRAL",
            "SentimentScore": {
                "Mixed": 0,
                "Negative": 0,
                "Neutral": 1,
                "Positive": 0
            }
        }
    ],
},
{
    "DescriptiveMentionIndex": [
        0
    ],
    "Mentions": [
        {
            "BeginOffset": 10,
            "EndOffset": 18,
            "Score": 0.631368,
            "GroupScore": 1,
            "Text": "AnyMovie",
            "Type": "ORGANIZATION",
            "MentionSentiment": {
                "Sentiment": "POSITIVE",
                "SentimentScore": {
                    "Mixed": 0.001729,
                    "Negative": 0.000001,
                    "Neutral": 0.000318,
                    "Positive": 0.997952
                }
            }
        }
    ]
}
],
"File": "SampleMovieReview1.txt",
"Line": 0
}

```

Contents of SampleMovieReview2.txt.out line indents for readability:

```
{
  "Entities": [
    {
      "DescriptiveMentionIndex": [
        0
      ],
      "Mentions": [
        {
          "BeginOffset": 0,
          "EndOffset": 8,
          "Score": 0.854024,
          "GroupScore": 1,
          "Text": "AnyMovie",
          "Type": "MOVIE",
          "MentionSentiment": {
            "Sentiment": "POSITIVE",
            "SentimentScore": {
              "Mixed": 0,
              "Negative": 0,
              "Neutral": 0.000007,
              "Positive": 0.999993
            }
          }
        }
      ],
        {
          "BeginOffset": 104,
          "EndOffset": 109,
          "Score": 0.999129,
          "GroupScore": 0.502937,
          "Text": "movie",
          "Type": "MOVIE",
          "MentionSentiment": {
            "Sentiment": "POSITIVE",
            "SentimentScore": {
              "Mixed": 0,
              "Negative": 0,
              "Neutral": 0,
              "Positive": 1
            }
          }
        }
      ],
        {
          "BeginOffset": 33,
```

```
    "EndOffset": 37,
    "Score": 0.999823,
    "GroupScore": 0.999252,
    "Text": "film",
    "Type": "MOVIE",
    "MentionSentiment": {
      "Sentiment": "POSITIVE",
      "SentimentScore": {
        "Mixed": 0,
        "Negative": 0,
        "Neutral": 0.000001,
        "Positive": 0.999999
      }
    }
  }
],
},
{
  "DescriptiveMentionIndex": [
    0,
    1,
    2
  ],
  "Mentions": [
    {
      "BeginOffset": 43,
      "EndOffset": 44,
      "Score": 0.999997,
      "GroupScore": 1,
      "Text": "I",
      "Type": "PERSON",
      "MentionSentiment": {
        "Sentiment": "NEUTRAL",
        "SentimentScore": {
          "Mixed": 0,
          "Negative": 0,
          "Neutral": 1,
          "Positive": 0
        }
      }
    }
  ],
  {
    "BeginOffset": 80,
    "EndOffset": 81,
```

```
    "Score": 0.999996,
    "GroupScore": 0.52523,
    "Text": "I",
    "Type": "PERSON",
    "MentionSentiment": {
      "Sentiment": "NEUTRAL",
      "SentimentScore": {
        "Mixed": 0,
        "Negative": 0,
        "Neutral": 1,
        "Positive": 0
      }
    }
  },
  {
    "BeginOffset": 67,
    "EndOffset": 68,
    "Score": 0.999994,
    "GroupScore": 0.999499,
    "Text": "I",
    "Type": "PERSON",
    "MentionSentiment": {
      "Sentiment": "NEUTRAL",
      "SentimentScore": {
        "Mixed": 0,
        "Negative": 0,
        "Neutral": 1,
        "Positive": 0
      }
    }
  }
],
{
  "DescriptiveMentionIndex": [
    0
  ],
  "Mentions": [
    {
      "BeginOffset": 75,
      "EndOffset": 78,
      "Score": 0.999978,
      "GroupScore": 1,
      "Text": "kid",
```



```
    "Type": "PERSON",
    "MentionSentiment": {
      "Sentiment": "NEUTRAL",
      "SentimentScore": {
        "Mixed": 0,
        "Negative": 0,
        "Neutral": 1,
        "Positive": 0
      }
    }
  ]
}
],
"File": "SampleMovieReview2.txt",
"Line": 0
}
```

Contents of SampleMovieReview3.txt.out with line indents for readability:

```
{
  "Entities": [
    {
      "DescriptiveMentionIndex": [
        1
      ],
      "Mentions": [
        {
          "BeginOffset": 64,
          "EndOffset": 68,
          "Score": 0.992953,
          "GroupScore": 0.999814,
          "Text": "film",
          "Type": "MOVIE",
          "MentionSentiment": {
            "Sentiment": "NEUTRAL",
            "SentimentScore": {
              "Mixed": 0.000004,
              "Negative": 0.010425,
              "Neutral": 0.989543,
              "Positive": 0.000027
            }
          }
        }
      ]
    }
  ]
}
```

```
    },
    {
      "BeginOffset": 37,
      "EndOffset": 45,
      "Score": 0.999782,
      "GroupScore": 1,
      "Text": "AnyMovie",
      "Type": "ORGANIZATION",
      "MentionSentiment": {
        "Sentiment": "POSITIVE",
        "SentimentScore": {
          "Mixed": 0.000095,
          "Negative": 0.039847,
          "Neutral": 0.000673,
          "Positive": 0.959384
        }
      }
    }
  ]
},
{
  "DescriptiveMentionIndex": [
    0
  ],
  "Mentions": [
    {
      "BeginOffset": 47,
      "EndOffset": 50,
      "Score": 0.999991,
      "GroupScore": 1,
      "Text": "All",
      "Type": "QUANTITY",
      "MentionSentiment": {
        "Sentiment": "NEUTRAL",
        "SentimentScore": {
          "Mixed": 0.000001,
          "Negative": 0.000001,
          "Neutral": 0.999998,
          "Positive": 0
        }
      }
    }
  ]
},
},
```

```
{
  "DescriptiveMentionIndex": [
    0
  ],
  "Mentions": [
    {
      "BeginOffset": 106,
      "EndOffset": 115,
      "Score": 0.542083,
      "GroupScore": 1,
      "Text": "directors",
      "Type": "PERSON",
      "MentionSentiment": {
        "Sentiment": "NEUTRAL",
        "SentimentScore": {
          "Mixed": 0,
          "Negative": 0,
          "Neutral": 1,
          "Positive": 0
        }
      }
    }
  ]
},
{
  "File": "SampleMovieReview3.txt",
  "Line": 0
}
}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StartTargetedSentimentDetectionJob](#) in *AWS CLI Command Reference*.

start-topics-detection-job

The following code example shows how to use `start-topics-detection-job`.

AWS CLI

To start a topics detection analysis job

The following `start-topics-detection-job` example starts an asynchronous topics detection job for all files located at the address specified by the `--input-data-config` tag. When the job is complete, the folder, output, is placed at the location specified by the `--output-data-config` tag. output contains `topic-terms.csv` and `doc-topics.csv`. The first output file, `topic-terms.csv`, is a list of topics in the collection. For each topic, the list includes, by default, the top terms by topic according to their weight. The second file, `doc-topics.csv`, lists the documents associated with a topic and the proportion of the document that is concerned with the topic.

```
aws comprehend start-topics-detection-job \
  --job-name example_topics_detection_job \
  --language-code en \
  --input-data-config "S3Uri=s3://DOC-EXAMPLE-BUCKET/" \
  --output-data-config "S3Uri=s3://DOC-EXAMPLE-DESTINATION-BUCKET/testfolder/" \
  --data-access-role-arn arn:aws:iam::111122223333:role/service-role/
AmazonComprehendServiceRole-example-role \
  --language-code en
```

Output:

```
{
  "JobId": "123456abcdeb0e11022f22a11EXAMPLE",
  "JobArn": "arn:aws:comprehend:us-west-2:111122223333:key-phrases-detection-
job/123456abcdeb0e11022f22a11EXAMPLE",
  "JobStatus": "SUBMITTED"
}
```

For more information, see [Topic Modeling](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StartTopicsDetectionJob](#) in *AWS CLI Command Reference*.

stop-dominant-language-detection-job

The following code example shows how to use `stop-dominant-language-detection-job`.

AWS CLI

To stop an asynchronous dominant language detection job

The following `stop-dominant-language-detection-job` example stops an in-progress, asynchronous dominant language detection job. If the current job state is `IN_PROGRESS` the

job is marked for termination and put into the `STOP_REQUESTED` state. If the job completes before it can be stopped, it is put into the `COMPLETED` state.

```
aws comprehend stop-dominant-language-detection-job \  
  --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

Output:

```
{  
  "JobId": "123456abcdeb0e11022f22a11EXAMPLE",  
  "JobStatus": "STOP_REQUESTED"  
}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StopDominantLanguageDetectionJob](#) in *AWS CLI Command Reference*.

stop-entities-detection-job

The following code example shows how to use `stop-entities-detection-job`.

AWS CLI

To stop an asynchronous entities detection job

The following `stop-entities-detection-job` example stops an in-progress, asynchronous entities detection job. If the current job state is `IN_PROGRESS` the job is marked for termination and put into the `STOP_REQUESTED` state. If the job completes before it can be stopped, it is put into the `COMPLETED` state.

```
aws comprehend stop-entities-detection-job \  
  --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

Output:

```
{  
  "JobId": "123456abcdeb0e11022f22a11EXAMPLE",  
  "JobStatus": "STOP_REQUESTED"  
}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StopEntitiesDetectionJob](#) in *AWS CLI Command Reference*.

stop-events-detection-job

The following code example shows how to use `stop-events-detection-job`.

AWS CLI

To stop an asynchronous events detection job

The following `stop-events-detection-job` example stops an in-progress, asynchronous events detection job. If the current job state is `IN_PROGRESS` the job is marked for termination and put into the `STOP_REQUESTED` state. If the job completes before it can be stopped, it is put into the `COMPLETED` state.

```
aws comprehend stop-events-detection-job \  
  --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

Output:

```
{  
  "JobId": "123456abcdeb0e11022f22a11EXAMPLE",  
  "JobStatus": "STOP_REQUESTED"  
}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StopEventsDetectionJob](#) in *AWS CLI Command Reference*.

stop-key-phrases-detection-job

The following code example shows how to use `stop-key-phrases-detection-job`.

AWS CLI

To stop an asynchronous key phrases detection job

The following `stop-key-phrases-detection-job` example stops an in-progress, asynchronous key phrases detection job. If the current job state is `IN_PROGRESS` the job is marked for termination and put into the `STOP_REQUESTED` state. If the job completes before it can be stopped, it is put into the `COMPLETED` state.

```
aws comprehend stop-key-phrases-detection-job \  
  --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

Output:

```
{  
  "JobId": "123456abcdeb0e11022f22a11EXAMPLE",  
  "JobStatus": "STOP_REQUESTED"  
}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StopKeyPhrasesDetectionJob](#) in *AWS CLI Command Reference*.

stop-pii-entities-detection-job

The following code example shows how to use `stop-pii-entities-detection-job`.

AWS CLI

To stop an asynchronous pii entities detection job

The following `stop-pii-entities-detection-job` example stops an in-progress, asynchronous pii entities detection job. If the current job state is `IN_PROGRESS` the job is marked for termination and put into the `STOP_REQUESTED` state. If the job completes before it can be stopped, it is put into the `COMPLETED` state.

```
aws comprehend stop-pii-entities-detection-job \  
  --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

Output:

```
{  
  "JobId": "123456abcdeb0e11022f22a11EXAMPLE",  
  "JobStatus": "STOP_REQUESTED"  
}
```

```
}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StopPiiEntitiesDetectionJob](#) in *AWS CLI Command Reference*.

stop-sentiment-detection-job

The following code example shows how to use stop-sentiment-detection-job.

AWS CLI

To stop an asynchronous sentiment detection job

The following stop-sentiment-detection-job example stops an in-progress, asynchronous sentiment detection job. If the current job state is IN_PROGRESS the job is marked for termination and put into the STOP_REQUESTED state. If the job completes before it can be stopped, it is put into the COMPLETED state.

```
aws comprehend stop-sentiment-detection-job \  
  --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

Output:

```
{  
  "JobId": "123456abcdeb0e11022f22a11EXAMPLE",  
  "JobStatus": "STOP_REQUESTED"  
}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StopSentimentDetectionJob](#) in *AWS CLI Command Reference*.

stop-targeted-sentiment-detection-job

The following code example shows how to use stop-targeted-sentiment-detection-job.

AWS CLI

To stop an asynchronous targeted sentiment detection job

The following `stop-targeted-sentiment-detection-job` example stops an in-progress, asynchronous targeted sentiment detection job. If the current job state is `IN_PROGRESS` the job is marked for termination and put into the `STOP_REQUESTED` state. If the job completes before it can be stopped, it is put into the `COMPLETED` state.

```
aws comprehend stop-targeted-sentiment-detection-job \  
  --job-id 123456abcdeb0e11022f22a11EXAMPLE
```

Output:

```
{  
  "JobId": "123456abcdeb0e11022f22a11EXAMPLE",  
  "JobStatus": "STOP_REQUESTED"  
}
```

For more information, see [Async analysis for Amazon Comprehend insights](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StopTargetedSentimentDetectionJob](#) in *AWS CLI Command Reference*.

stop-training-document-classifier

The following code example shows how to use `stop-training-document-classifier`.

AWS CLI

To stop the training of a document classifier model

The following `stop-training-document-classifier` example stops the training of a document classifier model while in-progress.

```
aws comprehend stop-training-document-classifier  
  --document-classifier-arn arn:aws:comprehend:us-west-2:111122223333:document-  
  classifier/example-classifier
```

This command produces no output.

For more information, see [Creating and managing custom models](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StopTrainingDocumentClassifier](#) in *AWS CLI Command Reference*.

stop-training-entity-recognizer

The following code example shows how to use stop-training-entity-recognizer.

AWS CLI

To stop the training of an entity recognizer model

The following stop-training-entity-recognizer example stops the training of an entity recognizer model while in-progress.

```
aws comprehend stop-training-entity-recognizer
  --entity-recognizer-arn "arn:aws:comprehend:us-west-2:111122223333:entity-
recognizer/examplerrecognizer1"
```

This command produces no output.

For more information, see [Creating and managing custom models](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [StopTrainingEntityRecognizer](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

Example 1: To tag a resource

The following tag-resource example adds a single tag to an Amazon Comprehend resource.

```
aws comprehend tag-resource \
  --resource-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/
example-classifier/version/1 \
  --tags Key=Location,Value=Seattle
```

This command has no output.

For more information, see [Tagging your resources](#) in the *Amazon Comprehend Developer Guide*.

Example 2: To add multiple tags to a resource

The following tag-resource example adds multiple tags to an Amazon Comprehend resource.

```
aws comprehend tag-resource \  
  --resource-arn "arn:aws:comprehend:us-west-2:111122223333:document-classifier/  
example-classifier/version/1" \  
  --tags Key=location,Value=Seattle Key=Department,Value=Finance
```

This command has no output.

For more information, see [Tagging your resources](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

Example 1: To remove a single tag from a resource

The following `untag-resource` example removes a single tag from an Amazon Comprehend resource.

```
aws comprehend untag-resource \  
  --resource-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/  
example-classifier/version/1  
  --tag-keys Location
```

This command produces no output.

For more information, see [Tagging your resources](#) in the *Amazon Comprehend Developer Guide*.

Example 2: To remove multiple tags from a resource

The following `untag-resource` example removes multiple tags from an Amazon Comprehend resource.

```
aws comprehend untag-resource \  
  --resource-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier/  
example-classifier/version/1  
  --tag-keys Location Department
```

This command produces no output.

For more information, see [Tagging your resources](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-endpoint

The following code example shows how to use `update-endpoint`.

AWS CLI

Example 1: To update an endpoint's inference units

The following `update-endpoint` example updates information about an endpoint. In this example, the number of inference units is increased.

```
aws comprehend update-endpoint \  
  --endpoint-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier-  
endpoint/example-classifier-endpoint  
  --desired-inference-units 2
```

This command produces no output.

For more information, see [Managing Amazon Comprehend endpoints](#) in the *Amazon Comprehend Developer Guide*.

Example 2: To update an endpoint's active model

The following `update-endpoint` example updates information about an endpoint. In this example, the active model is changed.

```
aws comprehend update-endpoint \  
  --endpoint-arn arn:aws:comprehend:us-west-2:111122223333:document-classifier-  
endpoint/example-classifier-endpoint  
  --active-model-arn arn:aws:comprehend:us-west-2:111122223333:document-  
classifier/example-classifier-new
```

This command produces no output.

For more information, see [Managing Amazon Comprehend endpoints](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [UpdateEndpoint](#) in *AWS CLI Command Reference*.

update-flywheel

The following code example shows how to use `update-flywheel`.

AWS CLI

To update a flywheel configuration

The following `update-flywheel` example updates a flywheel configuration. In this example, the active model for the flywheel is updated.

```
aws comprehend update-flywheel \  
  --flywheel-arn arn:aws:comprehend:us-west-2:111122223333:flywheel/example-  
flywheel-1 \  
  --active-model-arn arn:aws:comprehend:us-west-2:111122223333:document-  
classifier/example-classifier/version/new-example-classifier-model
```

Output:

```
{  
  "FlywheelProperties": {  
    "FlywheelArn": "arn:aws:comprehend:us-west-2:111122223333:flywheel/flywheel-  
entity",  
    "ActiveModelArn": "arn:aws:comprehend:us-west-2:111122223333:document-  
classifier/example-classifier/version/new-example-classifier-model",  
    "DataAccessRoleArn": "arn:aws:iam::111122223333:role/service-role/  
AmazonComprehendServiceRole-example-role",  
    "TaskConfig": {  
      "LanguageCode": "en",  
      "DocumentClassificationConfig": {  
        "Mode": "MULTI_CLASS"  
      }  
    },  
    "DataLakeS3Uri": "s3://DOC-EXAMPLE-BUCKET/flywheel-entity/  
schemaVersion=1/20230616T200543Z/",  
    "DataSecurityConfig": {},  
    "Status": "ACTIVE",  
    "ModelType": "DOCUMENT_CLASSIFIER",  
    "CreationTime": "2023-06-16T20:05:43.242000+00:00",  
    "LastModifiedTime": "2023-06-19T04:00:43.027000+00:00",  
    "LatestFlywheelIteration": "20230619T040032Z"  
  }  
}
```

For more information, see [Flywheel overview](#) in the *Amazon Comprehend Developer Guide*.

- For API details, see [UpdateFlywheel](#) in *AWS CLI Command Reference*.

Amazon Comprehend Medical examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon Comprehend Medical.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

describe-entities-detection-v2-job

The following code example shows how to use `describe-entities-detection-v2-job`.

AWS CLI

To describe an entities detection job

The following `describe-entities-detection-v2-job` example displays the properties associated with an asynchronous entity detection job.

```
aws comprehendmedical describe-entities-detection-v2-job \  
  --job-id "ab9887877365fe70299089371c043b96"
```

Output:

```
{
```

```

"ComprehendMedicalAsyncJobProperties": {
  "JobId": "ab9887877365fe70299089371c043b96",
  "JobStatus": "COMPLETED",
  "SubmitTime": "2020-03-18T21:20:15.614000+00:00",
  "EndTime": "2020-03-18T21:27:07.350000+00:00",
  "ExpirationTime": "2020-07-16T21:20:15+00:00",
  "InputDataConfig": {
    "S3Bucket": "comp-med-input",
    "S3Key": ""
  },
  "OutputDataConfig": {
    "S3Bucket": "comp-med-output",
    "S3Key": "867139942017-EntitiesDetection-
ab9887877365fe70299089371c043b96/"
  },
  "LanguageCode": "en",
  "DataAccessRoleArn": "arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole",
  "ModelVersion": "DetectEntitiesModelV20190930"
}
}

```

For more information, see [Batch APIs](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [DescribeEntitiesDetectionV2Job](#) in *AWS CLI Command Reference*.

describe-icd10-cm-inference-job

The following code example shows how to use `describe-icd10-cm-inference-job`.

AWS CLI

To describe an ICD-10-CM inference job

The following `describe-icd10-cm-inference-job` example describes the properties of the requested inference job with the specified `job-id`.

```

aws comprehendmedical describe-icd10-cm-inference-job \
  --job-id "5780034166536cdb52ffa3295a1b00a7"

```

Output:

```
{
```

```

"ComprehendMedicalAsyncJobProperties": {
  "JobId": "5780034166536cdb52ffa3295a1b00a7",
  "JobStatus": "COMPLETED",
  "SubmitTime": "2020-05-18T21:20:15.614000+00:00",
  "EndTime": "2020-05-18T21:27:07.350000+00:00",
  "ExpirationTime": "2020-09-16T21:20:15+00:00",
  "InputDataConfig": {
    "S3Bucket": "comp-med-input",
    "S3Key": "AKIAIOSFODNN7EXAMPLE"
  },
  "OutputDataConfig": {
    "S3Bucket": "comp-med-output",
    "S3Key": "AKIAIOSFODNN7EXAMPLE"
  },
  "LanguageCode": "en",
  "DataAccessRoleArn": "arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole",
  "ModelVersion": "0.1.0"
}
}

```

For more information, see [Ontology linking batch analysis](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [DescribeLcd10CmInferenceJob](#) in *AWS CLI Command Reference*.

describe-phi-detection-job

The following code example shows how to use `describe-phi-detection-job`.

AWS CLI

To describe a PHI detection job

The following `describe-phi-detection-job` example displays the properties associated with an asynchronous protected health information (PHI) detection job.

```

aws comprehendmedical describe-phi-detection-job \
  --job-id "4750034166536cdb52ffa3295a1b00a3"

```

Output:

```
{
```



```

    "ComprehendMedicalAsyncJobProperties": {
      "JobId": "4750034166536cdb52ffa3295a1b00a3",
      "JobStatus": "COMPLETED",
      "SubmitTime": "2020-03-19T20:38:37.594000+00:00",
      "EndTime": "2020-03-19T20:45:07.894000+00:00",
      "ExpirationTime": "2020-07-17T20:38:37+00:00",
      "InputDataConfig": {
        "S3Bucket": "comp-med-input",
        "S3Key": ""
      },
      "OutputDataConfig": {
        "S3Bucket": "comp-med-output",
        "S3Key": "867139942017-PHIDetection-4750034166536cdb52ffa3295a1b00a3/"
      },
      "LanguageCode": "en",
      "DataAccessRoleArn": "arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole",
      "ModelVersion": "PHIModelV20190903"
    }
  }
}

```

For more information, see [Batch APIs](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [DescribePhiDetectionJob](#) in *AWS CLI Command Reference*.

describe-rx-norm-inference-job

The following code example shows how to use `describe-rx-norm-inference-job`.

AWS CLI

To describe an RxNorm inference job

The following `describe-rx-norm-inference-job` example describes the properties of the requested inference job with the specified job-id.

```

aws comprehendmedical describe-rx-norm-inference-job \
  --job-id "eg8199877365fc70299089371c043b96"

```

Output:

```

{
  "ComprehendMedicalAsyncJobProperties": {

```

```

    "JobId": "g8199877365fc70299089371c043b96",
    "JobStatus": "COMPLETED",
    "SubmitTime": "2020-05-18T21:20:15.614000+00:00",
    "EndTime": "2020-05-18T21:27:07.350000+00:00",
    "ExpirationTime": "2020-09-16T21:20:15+00:00",
    "InputDataConfig": {
      "S3Bucket": "comp-med-input",
      "S3Key": "AKIAIOSFODNN7EXAMPLE"
    },
    "OutputDataConfig": {
      "S3Bucket": "comp-med-output",
      "S3Key": "AKIAIOSFODNN7EXAMPLE"
    },
    "LanguageCode": "en",
    "DataAccessRoleArn": "arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole",
    "ModelVersion": "0.0.0"
  }
}

```

For more information, see [Ontology linking batch analysis](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [DescribeRxNormInferenceJob](#) in *AWS CLI Command Reference*.

describe-snomedct-inference-job

The following code example shows how to use `describe-snomedct-inference-job`.

AWS CLI

To describe an SNOMED CT inference job

The following `describe-snomedct-inference-job` example describes the properties of the requested inference job with the specified job-id.

```

aws comprehendmedical describe-snomedct-inference-job \
  --job-id "2630034166536cdb52ffa3295a1b00a7"

```

Output:

```
{
```

```

"ComprehendMedicalAsyncJobProperties": {
  "JobId": "2630034166536cdb52ffa3295a1b00a7",
  "JobStatus": "COMPLETED",
  "SubmitTime": "2021-12-18T21:20:15.614000+00:00",
  "EndTime": "2021-12-18T21:27:07.350000+00:00",
  "ExpirationTime": "2022-05-16T21:20:15+00:00",
  "InputDataConfig": {
    "S3Bucket": "comp-med-input",
    "S3Key": "AKIAIOSFODNN7EXAMPLE"
  },
  "OutputDataConfig": {
    "S3Bucket": "comp-med-output",
    "S3Key": "AKIAIOSFODNN7EXAMPLE"
  },
  "LanguageCode": "en",
  "DataAccessRoleArn": "arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole",
  "ModelVersion": "0.1.0"
}
}

```

For more information, see [Ontology linking batch analysis](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [DescribeSnomedctInferenceJob](#) in *AWS CLI Command Reference*.

detect-entities-v2

The following code example shows how to use `detect-entities-v2`.

AWS CLI

Example 1: To detect entities directly from text

The following `detect-entities-v2` example shows the detected entities and labels them according to type, directly from input text.

```

aws comprehendmedical detect-entities-v2 \
  --text "Sleeping trouble on present dosage of Clonidine. Severe rash on face and
leg, slightly itchy."

```

Output:

```
{
  "Id": 0,
  "BeginOffset": 38,
  "EndOffset": 47,
  "Score": 0.9942955374717712,
  "Text": "Clonidine",
  "Category": "MEDICATION",
  "Type": "GENERIC_NAME",
  "Traits": []
}
```

For more information, see [Detect Entities Version 2](#) in the *Amazon Comprehend Medical Developer Guide*.

Example 2: To detect entities from a file path

The following `detect-entities-v2` example shows the detected entities and labels them according to type from a file path.

```
aws comprehendmedical detect-entities-v2 \
  --text file://medical_entities.txt
```

Contents of `medical_entities.txt`:

```
{
  "Sleeping trouble on present dosage of Clonidine. Severe rash on face and leg,
  slightly itchy."
}
```

Output:

```
{
  "Id": 0,
  "BeginOffset": 38,
  "EndOffset": 47,
  "Score": 0.9942955374717712,
  "Text": "Clonidine",
  "Category": "MEDICATION",
  "Type": "GENERIC_NAME",
  "Traits": []
}
```

For more information, see [Detect Entities Version 2](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [DetectEntitiesV2](#) in *AWS CLI Command Reference*.

detect-phi

The following code example shows how to use `detect-phi`.

AWS CLI

Example 1: To detect protected health information (PHI) directly from text

The following `detect-phi` example displays the detected protected health information (PHI) entities directly from input text.

```
aws comprehendmedical detect-phi \  
  --text "Patient Carlos Salazar presented with rash on his upper extremities and  
  dry cough. He lives at 100 Main Street, Anytown, USA where he works from his home  
  as a carpenter."
```

Output:

```
{  
  "Entities": [  
    {  
      "Id": 0,  
      "BeginOffset": 8,  
      "EndOffset": 21,  
      "Score": 0.9914507269859314,  
      "Text": "Carlos Salazar",  
      "Category": "PROTECTED_HEALTH_INFORMATION",  
      "Type": "NAME",  
      "Traits": []  
    },  
    {  
      "Id": 1,  
      "BeginOffset": 94,  
      "EndOffset": 109,  
      "Score": 0.871849775314331,  
      "Text": "100 Main Street, Anytown, USA",  
      "Category": "PROTECTED_HEALTH_INFORMATION",  
      "Type": "ADDRESS",
```

```

    "Traits": []
  },
  {
    "Id": 2,
    "BeginOffset": 145,
    "EndOffset": 154,
    "Score": 0.8302185535430908,
    "Text": "carpenter",
    "Category": "PROTECTED_HEALTH_INFORMATION",
    "Type": "PROFESSION",
    "Traits": []
  }
],
"ModelVersion": "0.0.0"
}

```

For more information, see [Detect PHI](#) in the *Amazon Comprehend Medical Developer Guide*.

Example 2: To detect protect health information (PHI) directly from a file path

The following detect-phi example shows the detected protected health information (PHI) entities from a file path.

```
aws comprehendmedical detect-phi \
  --text file://phi.txt
```

Contents of phi.txt:

```
"Patient Carlos Salazar presented with a rash on his upper extremities and a dry cough. He lives at 100 Main Street, Anytown, USA, where he works from his home as a carpenter."
```

Output:

```

{
  "Entities": [
    {
      "Id": 0,
      "BeginOffset": 8,
      "EndOffset": 21,
      "Score": 0.9914507269859314,
      "Text": "Carlos Salazar",
      "Category": "PROTECTED_HEALTH_INFORMATION",

```

```

    "Type": "NAME",
    "Traits": []
  },
  {
    "Id": 1,
    "BeginOffset": 94,
    "EndOffset": 109,
    "Score": 0.871849775314331,
    "Text": "100 Main Street, Anytown, USA",
    "Category": "PROTECTED_HEALTH_INFORMATION",
    "Type": "ADDRESS",
    "Traits": []
  },
  {
    "Id": 2,
    "BeginOffset": 145,
    "EndOffset": 154,
    "Score": 0.8302185535430908,
    "Text": "carpenter",
    "Category": "PROTECTED_HEALTH_INFORMATION",
    "Type": "PROFESSION",
    "Traits": []
  }
],
"ModelVersion": "0.0.0"
}

```

For more information, see [Detect PHI](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [DetectPhi](#) in *AWS CLI Command Reference*.

infer-icd10-cm

The following code example shows how to use `infer-icd10-cm`.

AWS CLI

Example 1: To detect medical condition entities and link to the ICD-10-CM Ontology directly from text

The following `infer-icd10-cm` example labels the detected medical condition entities and links those entities with codes in the 2019 edition of the International Classification of Diseases Clinical Modification (ICD-10-CM).

```
aws comprehendmedical infer-icd10-cm \
  --text "The patient complains of abdominal pain, has a long-standing history of
  diabetes treated with Micronase daily."
```

Output:

```
{
  "Entities": [
    {
      "Id": 0,
      "Text": "abdominal pain",
      "Category": "MEDICAL_CONDITION",
      "Type": "DX_NAME",
      "Score": 0.9475538730621338,
      "BeginOffset": 28,
      "EndOffset": 42,
      "Attributes": [],
      "Traits": [
        {
          "Name": "SYMPTOM",
          "Score": 0.6724207401275635
        }
      ],
      "ICD10CMConcepts": [
        {
          "Description": "Unspecified abdominal pain",
          "Code": "R10.9",
          "Score": 0.6904221177101135
        },
        {
          "Description": "Epigastric pain",
          "Code": "R10.13",
          "Score": 0.1364113688468933
        },
        {
          "Description": "Generalized abdominal pain",
          "Code": "R10.84",
          "Score": 0.12508003413677216
        },
        {
          "Description": "Left lower quadrant pain",
          "Code": "R10.32",
          "Score": 0.10063883662223816
        }
      ]
    }
  ]
}
```



```

    },
    {
      "Description": "Lower abdominal pain, unspecified",
      "Code": "R10.30",
      "Score": 0.09933677315711975
    }
  ]
},
{
  "Id": 1,
  "Text": "diabetes",
  "Category": "MEDICAL_CONDITION",
  "Type": "DX_NAME",
  "Score": 0.9899052977561951,
  "BeginOffset": 75,
  "EndOffset": 83,
  "Attributes": [],
  "Traits": [
    {
      "Name": "DIAGNOSIS",
      "Score": 0.9258432388305664
    }
  ],
  "ICD10CMConcepts": [
    {
      "Description": "Type 2 diabetes mellitus without complications",
      "Code": "E11.9",
      "Score": 0.7158446311950684
    },
    {
      "Description": "Family history of diabetes mellitus",
      "Code": "Z83.3",
      "Score": 0.5704703330993652
    },
    {
      "Description": "Family history of other endocrine, nutritional
and metabolic diseases",
      "Code": "Z83.49",
      "Score": 0.19856023788452148
    },
    {
      "Description": "Type 1 diabetes mellitus with ketoacidosis
without coma",
      "Code": "E10.10",

```

```

        "Score": 0.13285516202449799
      },
      {
        "Description": "Type 2 diabetes mellitus with hyperglycemia",
        "Code": "E11.65",
        "Score": 0.0993388369679451
      }
    ]
  },
  ],
  "ModelVersion": "0.1.0"
}

```

For more information, see [Infer ICD10-CM](#) in the *Amazon Comprehend Medical Developer Guide*.

Example 2: To detect medical condition entities and link to the ICD-10-CM Ontology from a file pathway

The following `infer-icd-10-cm` example labels the detected medical condition entities and links those entities with codes in the 2019 edition of the International Classification of Diseases Clinical Modification (ICD-10-CM).

```
aws comprehendmedical infer-icd10-cm \
  --text file://icd10cm.txt
```

Contents of `icd10cm.txt`:

```
{
  "The patient complains of abdominal pain, has a long-standing history of
  diabetes treated with Micronase daily."
}
```

Output:

```
{
  "Entities": [
    {
      "Id": 0,
      "Text": "abdominal pain",
      "Category": "MEDICAL_CONDITION",
      "Type": "DX_NAME",
      "Score": 0.9475538730621338,

```

```
"BeginOffset": 28,
"EndOffset": 42,
"Attributes": [],
"Traits": [
  {
    "Name": "SYMPTOM",
    "Score": 0.6724207401275635
  }
],
"ICD10CMConcepts": [
  {
    "Description": "Unspecified abdominal pain",
    "Code": "R10.9",
    "Score": 0.6904221177101135
  },
  {
    "Description": "Epigastric pain",
    "Code": "R10.13",
    "Score": 0.1364113688468933
  },
  {
    "Description": "Generalized abdominal pain",
    "Code": "R10.84",
    "Score": 0.12508003413677216
  },
  {
    "Description": "Left lower quadrant pain",
    "Code": "R10.32",
    "Score": 0.10063883662223816
  },
  {
    "Description": "Lower abdominal pain, unspecified",
    "Code": "R10.30",
    "Score": 0.09933677315711975
  }
]
},
{
  "Id": 1,
  "Text": "diabetes",
  "Category": "MEDICAL_CONDITION",
  "Type": "DX_NAME",
  "Score": 0.9899052977561951,
  "BeginOffset": 75,
```

```
"EndOffset": 83,
"Attributes": [],
"Traits": [
  {
    "Name": "DIAGNOSIS",
    "Score": 0.9258432388305664
  }
],
"ICD10CMConcepts": [
  {
    "Description": "Type 2 diabetes mellitus without complications",
    "Code": "E11.9",
    "Score": 0.7158446311950684
  },
  {
    "Description": "Family history of diabetes mellitus",
    "Code": "Z83.3",
    "Score": 0.5704703330993652
  },
  {
    "Description": "Family history of other endocrine, nutritional
and metabolic diseases",
    "Code": "Z83.49",
    "Score": 0.19856023788452148
  },
  {
    "Description": "Type 1 diabetes mellitus with ketoacidosis
without coma",
    "Code": "E10.10",
    "Score": 0.13285516202449799
  },
  {
    "Description": "Type 2 diabetes mellitus with hyperglycemia",
    "Code": "E11.65",
    "Score": 0.0993388369679451
  }
]
}
],
"ModelVersion": "0.1.0"
}
```

For more information, see [Infer-ICD10-CM](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [InferIcd10Cm](#) in *AWS CLI Command Reference*.

infer-rx-norm

The following code example shows how to use `infer-rx-norm`.

AWS CLI

Example 1: To detect medication entities and link to RxNorm directly from text

The following `infer-rx-norm` example shows and labels the detected medication entities and links those entities to concept identifiers (RxCUI) from the National Library of Medicine RxNorm database.

```
aws comprehendmedical infer-rx-norm \  
  --text "Patient reports taking Levothyroxine 125 micrograms p.o. once daily, but  
denies taking Synthroid."
```

Output:

```
{  
  "Entities": [  
    {  
      "Id": 0,  
      "Text": "Levothyroxine",  
      "Category": "MEDICATION",  
      "Type": "GENERIC_NAME",  
      "Score": 0.9996285438537598,  
      "BeginOffset": 23,  
      "EndOffset": 36,  
      "Attributes": [  
        {  
          "Type": "DOSAGE",  
          "Score": 0.9892290830612183,  
          "RelationshipScore": 0.9997978806495667,  
          "Id": 1,  
          "BeginOffset": 37,  
          "EndOffset": 51,  
          "Text": "125 micrograms",  
          "Traits": []  
        },  
        {
```

```

        "Type": "ROUTE_OR_MODE",
        "Score": 0.9988924860954285,
        "RelationshipScore": 0.998291552066803,
        "Id": 2,
        "BeginOffset": 52,
        "EndOffset": 56,
        "Text": "p.o.",
        "Traits": []
    },
    {
        "Type": "FREQUENCY",
        "Score": 0.9953463673591614,
        "RelationshipScore": 0.9999889135360718,
        "Id": 3,
        "BeginOffset": 57,
        "EndOffset": 67,
        "Text": "once daily",
        "Traits": []
    }
],
"Traits": [],
"RxNormConcepts": [
    {
        "Description": "Levothyroxine Sodium 0.125 MG Oral Tablet",
        "Code": "966224",
        "Score": 0.9912070631980896
    },
    {
        "Description": "Levothyroxine Sodium 0.125 MG Oral Capsule",
        "Code": "966405",
        "Score": 0.8698278665542603
    },
    {
        "Description": "Levothyroxine Sodium 0.125 MG Oral Tablet
[Synthroid]",
        "Code": "966191",
        "Score": 0.7448257803916931
    },
    {
        "Description": "levothyroxine",
        "Code": "10582",
        "Score": 0.7050482630729675
    },
    {

```

```

        "Description": "Levothyroxine Sodium 0.125 MG Oral Tablet
[Levoxy1]",
        "Code": "966190",
        "Score": 0.6921631693840027
    }
]
},
{
    "Id": 4,
    "Text": "Synthroid",
    "Category": "MEDICATION",
    "Type": "BRAND_NAME",
    "Score": 0.9946461319923401,
    "BeginOffset": 86,
    "EndOffset": 95,
    "Attributes": [],
    "Traits": [
        {
            "Name": "NEGATION",
            "Score": 0.5167351961135864
        }
    ],
    "RxNormConcepts": [
        {
            "Description": "Synthroid",
            "Code": "224920",
            "Score": 0.9462039470672607
        },
        {
            "Description": "Levothyroxine Sodium 0.088 MG Oral Tablet
[Synthroid]",
            "Code": "966282",
            "Score": 0.8309829235076904
        },
        {
            "Description": "Levothyroxine Sodium 0.125 MG Oral Tablet
[Synthroid]",
            "Code": "966191",
            "Score": 0.4945160448551178
        },
        {
            "Description": "Levothyroxine Sodium 0.05 MG Oral Tablet
[Synthroid]",
            "Code": "966247",

```

```

        "Score": 0.3674522042274475
      },
      {
        "Description": "Levothyroxine Sodium 0.025 MG Oral Tablet
[Synthroid]",
        "Code": "966158",
        "Score": 0.2588822841644287
      }
    ]
  },
  "ModelVersion": "0.0.0"
}

```

For more information, see [Infer RxNorm](#) in the *Amazon Comprehend Medical Developer Guide*.

Example 2: To detect medication entities and link to RxNorm from a file path.

The following `infer-rx-norm` example shows and labels the detected medication entities and links those entities to concept identifiers (RxCUI) from the National Library of Medicine RxNorm database.

```
aws comprehendmedical infer-rx-norm \
  --text file://rxnorm.txt
```

Contents of `rxnorm.txt`:

```
{
  "Patient reports taking Levothyroxine 125 micrograms p.o. once daily, but denies
taking Synthroid."
}
```

Output:

```
{
  "Entities": [
    {
      "Id": 0,
      "Text": "Levothyroxine",
      "Category": "MEDICATION",
      "Type": "GENERIC_NAME",
      "Score": 0.9996285438537598,

```



```
"BeginOffset": 23,
"EndOffset": 36,
"Attributes": [
  {
    "Type": "DOSAGE",
    "Score": 0.9892290830612183,
    "RelationshipScore": 0.9997978806495667,
    "Id": 1,
    "BeginOffset": 37,
    "EndOffset": 51,
    "Text": "125 micrograms",
    "Traits": []
  },
  {
    "Type": "ROUTE_OR_MODE",
    "Score": 0.9988924860954285,
    "RelationshipScore": 0.998291552066803,
    "Id": 2,
    "BeginOffset": 52,
    "EndOffset": 56,
    "Text": "p.o.",
    "Traits": []
  },
  {
    "Type": "FREQUENCY",
    "Score": 0.9953463673591614,
    "RelationshipScore": 0.9999889135360718,
    "Id": 3,
    "BeginOffset": 57,
    "EndOffset": 67,
    "Text": "once daily",
    "Traits": []
  }
],
"Traits": [],
"RxNormConcepts": [
  {
    "Description": "Levothyroxine Sodium 0.125 MG Oral Tablet",
    "Code": "966224",
    "Score": 0.9912070631980896
  },
  {
    "Description": "Levothyroxine Sodium 0.125 MG Oral Capsule",
    "Code": "966405",
```

```

        "Score": 0.8698278665542603
      },
      {
        "Description": "Levothyroxine Sodium 0.125 MG Oral Tablet
[Synthroid]",
        "Code": "966191",
        "Score": 0.7448257803916931
      },
      {
        "Description": "levothyroxine",
        "Code": "10582",
        "Score": 0.7050482630729675
      },
      {
        "Description": "Levothyroxine Sodium 0.125 MG Oral Tablet
[Levoxyl]",
        "Code": "966190",
        "Score": 0.6921631693840027
      }
    ]
  },
  {
    "Id": 4,
    "Text": "Synthroid",
    "Category": "MEDICATION",
    "Type": "BRAND_NAME",
    "Score": 0.9946461319923401,
    "BeginOffset": 86,
    "EndOffset": 95,
    "Attributes": [],
    "Traits": [
      {
        "Name": "NEGATION",
        "Score": 0.5167351961135864
      }
    ],
    "RxNormConcepts": [
      {
        "Description": "Synthroid",
        "Code": "224920",
        "Score": 0.9462039470672607
      },
      {

```

```

    "Description": "Levothyroxine Sodium 0.088 MG Oral Tablet
  [Synthroid]",
    "Code": "966282",
    "Score": 0.8309829235076904
  },
  {
    "Description": "Levothyroxine Sodium 0.125 MG Oral Tablet
  [Synthroid]",
    "Code": "966191",
    "Score": 0.4945160448551178
  },
  {
    "Description": "Levothyroxine Sodium 0.05 MG Oral Tablet
  [Synthroid]",
    "Code": "966247",
    "Score": 0.3674522042274475
  },
  {
    "Description": "Levothyroxine Sodium 0.025 MG Oral Tablet
  [Synthroid]",
    "Code": "966158",
    "Score": 0.2588822841644287
  }
]
}
],
"ModelVersion": "0.0.0"
}

```

For more information, see [Infer RxNorm](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [InferRxNorm](#) in *AWS CLI Command Reference*.

infer-snomedct

The following code example shows how to use `infer-snomedct`.

AWS CLI

Example: To detect entities and link to the SNOMED CT Ontology directly from text

The following `infer-snomedct` example shows how to detect medical entities and link them to concepts from the 2021-03 version of the Systematized Nomenclature of Medicine, Clinical Terms (SNOMED CT).

```
aws comprehendmedical infer-snomedct \  
  --text "The patient complains of abdominal pain, has a long-standing history of  
  diabetes treated with Micronase daily."
```

Output:

```
{  
  "Entities": [  
    {  
      "Id": 3,  
      "BeginOffset": 26,  
      "EndOffset": 40,  
      "Score": 0.9598260521888733,  
      "Text": "abdominal pain",  
      "Category": "MEDICAL_CONDITION",  
      "Type": "DX_NAME",  
      "Traits": [  
        {  
          "Name": "SYMPTOM",  
          "Score": 0.6819021701812744  
        }  
      ]  
    },  
    {  
      "Id": 4,  
      "BeginOffset": 73,  
      "EndOffset": 81,  
      "Score": 0.9905840158462524,  
      "Text": "diabetes",  
      "Category": "MEDICAL_CONDITION",  
      "Type": "DX_NAME",  
      "Traits": [  
        {  
          "Name": "DIAGNOSIS",  
          "Score": 0.9255214333534241  
        }  
      ]  
    },  
  ]  
}
```

```
    "Id": 1,
    "BeginOffset": 95,
    "EndOffset": 104,
    "Score": 0.6371926665306091,
    "Text": "Micronase",
    "Category": "MEDICATION",
    "Type": "BRAND_NAME",
    "Traits": [],
    "Attributes": [
      {
        "Type": "FREQUENCY",
        "Score": 0.9761165380477905,
        "RelationshipScore": 0.9984188079833984,
        "RelationshipType": "FREQUENCY",
        "Id": 2,
        "BeginOffset": 105,
        "EndOffset": 110,
        "Text": "daily",
        "Category": "MEDICATION",
        "Traits": []
      }
    ]
  },
  ],
  "UnmappedAttributes": [],
  "ModelVersion": "1.0.0"
}
```

For more information, see [InferSNOMEDCT](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [InferSnomedct](#) in *AWS CLI Command Reference*.

list-entities-detection-v2-jobs

The following code example shows how to use `list-entities-detection-v2-jobs`.

AWS CLI

To list entities detection jobs

The following `list-entities-detection-v2-jobs` example lists current asynchronous detection jobs.

```
aws comprehendmedical list-entities-detection-v2-jobs
```

Output:

```
{
  "ComprehendMedicalAsyncJobPropertiesList": [
    {
      "JobId": "ab9887877365fe70299089371c043b96",
      "JobStatus": "COMPLETED",
      "SubmitTime": "2020-03-19T20:38:37.594000+00:00",
      "EndTime": "2020-03-19T20:45:07.894000+00:00",
      "ExpirationTime": "2020-07-17T20:38:37+00:00",
      "InputDataConfig": {
        "S3Bucket": "comp-med-input",
        "S3Key": ""
      },
      "OutputDataConfig": {
        "S3Bucket": "comp-med-output",
        "S3Key": "867139942017-EntitiesDetection-ab9887877365fe70299089371c043b96/"
      },
      "LanguageCode": "en",
      "DataAccessRoleArn": "arn:aws:iam::867139942017:role/ComprehendMedicalBatchProcessingRole",
      "ModelVersion": "DetectEntitiesModelV20190930"
    }
  ]
}
```

For more information, see [Batch APIs](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [ListEntitiesDetectionV2Jobs](#) in *AWS CLI Command Reference*.

list-icd10-cm-inference-jobs

The following code example shows how to use `list-icd10-cm-inference-jobs`.

AWS CLI

To list all current ICD-10-CM inference jobs

The following example shows how the `list-icd10-cm-inference-jobs` operation returns a list of current asynchronous ICD-10-CM batch inference jobs.

```
aws comprehendmedical list-icd10-cm-inference-jobs
```

Output:

```
{
  "ComprehendMedicalAsyncJobPropertiesList": [
    {
      "JobId": "5780034166536cdb52ffa3295a1b00a7",
      "JobStatus": "COMPLETED",
      "SubmitTime": "2020-05-19T20:38:37.594000+00:00",
      "EndTime": "2020-05-19T20:45:07.894000+00:00",
      "ExpirationTime": "2020-09-17T20:38:37+00:00",
      "InputDataConfig": {
        "S3Bucket": "comp-med-input",
        "S3Key": "AKIAIOSFODNN7EXAMPLE"
      },
      "OutputDataConfig": {
        "S3Bucket": "comp-med-output",
        "S3Key": "AKIAIOSFODNN7EXAMPLE"
      },
      "LanguageCode": "en",
      "DataAccessRoleArn": "arn:aws:iam::867139942017:role/ComprehendMedicalBatchProcessingRole",
      "ModelVersion": "0.1.0"
    }
  ]
}
```

For more information, see [Ontology linking batch analysis](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [ListIcd10CmInferenceJobs](#) in *AWS CLI Command Reference*.

list-phi-detection-jobs

The following code example shows how to use `list-phi-detection-jobs`.

AWS CLI

To list protected health information (PHI) detection jobs

The following `list-phi-detection-jobs` example lists current protected health information (PHI) detection jobs

```
aws comprehendmedical list-phi-detection-jobs
```

Output:

```
{
  "ComprehendMedicalAsyncJobPropertiesList": [
    {
      "JobId": "4750034166536cdb52ffa3295a1b00a3",
      "JobStatus": "COMPLETED",
      "SubmitTime": "2020-03-19T20:38:37.594000+00:00",
      "EndTime": "2020-03-19T20:45:07.894000+00:00",
      "ExpirationTime": "2020-07-17T20:38:37+00:00",
      "InputDataConfig": {
        "S3Bucket": "comp-med-input",
        "S3Key": ""
      },
      "OutputDataConfig": {
        "S3Bucket": "comp-med-output",
        "S3Key": "867139942017-
PHIDetection-4750034166536cdb52ffa3295a1b00a3/"
      },
      "LanguageCode": "en",
      "DataAccessRoleArn": "arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole",
      "ModelVersion": "PHIModelV20190903"
    }
  ]
}
```

For more information, see [Batch APIs](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [ListPhiDetectionJobs](#) in *AWS CLI Command Reference*.

`list-rx-norm-inference-jobs`

The following code example shows how to use `list-rx-norm-inference-jobs`.

AWS CLI

To list all current Rx-Norm inference jobs

The following example shows how `list-rx-norm-inference-jobs` returns a list of current asynchronous Rx-Norm batch inference jobs.

```
aws comprehendmedical list-rx-norm-inference-jobs
```

Output:

```
{
  "ComprehendMedicalAsyncJobPropertiesList": [
    {
      "JobId": "4980034166536cfb52gga3295a1b00a3",
      "JobStatus": "COMPLETED",
      "SubmitTime": "2020-05-19T20:38:37.594000+00:00",
      "EndTime": "2020-05-19T20:45:07.894000+00:00",
      "ExpirationTime": "2020-09-17T20:38:37+00:00",
      "InputDataConfig": {
        "S3Bucket": "comp-med-input",
        "S3Key": "AKIAIOSFODNN7EXAMPLE"
      },
      "OutputDataConfig": {
        "S3Bucket": "comp-med-output",
        "S3Key": "AKIAIOSFODNN7EXAMPLE"
      },
      "LanguageCode": "en",
      "DataAccessRoleArn": "arn:aws:iam::867139942017:role/ComprehendMedicalBatchProcessingRole",
      "ModelVersion": "0.0.0"
    }
  ]
}
```

For more information, see [Ontology linking batch analysis](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [ListRxNormInferenceJobs](#) in *AWS CLI Command Reference*.

`list-snomedct-inference-jobs`

The following code example shows how to use `list-snomedct-inference-jobs`.

AWS CLI

To list all SNOMED CT inference jobs

The following example shows how the `list-snomedct-inference-jobs` operation returns a list of current asynchronous SNOMED CT batch inference jobs.

```
aws comprehendmedical list-snomedct-inference-jobs
```

Output:

```
{
  "ComprehendMedicalAsyncJobPropertiesList": [
    {
      "JobId": "5780034166536cdb52ffa3295a1b00a7",
      "JobStatus": "COMPLETED",
      "SubmitTime": "2020-05-19T20:38:37.594000+00:00",
      "EndTime": "2020-05-19T20:45:07.894000+00:00",
      "ExpirationTime": "2020-09-17T20:38:37+00:00",
      "InputDataConfig": {
        "S3Bucket": "comp-med-input",
        "S3Key": "AKIAIOSFODNN7EXAMPLE"
      },
      "OutputDataConfig": {
        "S3Bucket": "comp-med-output",
        "S3Key": "AKIAIOSFODNN7EXAMPLE"
      },
      "LanguageCode": "en",
      "DataAccessRoleArn": "arn:aws:iam::867139942017:role/ComprehendMedicalBatchProcessingRole",
      "ModelVersion": "0.1.0"
    }
  ]
}
```

For more information, see [Ontology linking batch analysis](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [ListSnomedctInferenceJobs](#) in *AWS CLI Command Reference*.

start-entities-detection-v2-job

The following code example shows how to use `start-entities-detection-v2-job`.

AWS CLI

To start an entities detection job

The following `start-entities-detection-v2-job` example starts an asynchronous entity detection job.

```
aws comprehendmedical start-entities-detection-v2-job \  
  --input-data-config "S3Bucket=comp-med-input" \  
  --output-data-config "S3Bucket=comp-med-output" \  
  --data-access-role-arn arn:aws:iam::867139942017:role/  
ComprehendMedicalBatchProcessingRole \  
  --language-code en
```

Output:

```
{  
  "JobId": "ab9887877365fe70299089371c043b96"  
}
```

For more information, see [Batch APIs](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [StartEntitiesDetectionV2Job](#) in *AWS CLI Command Reference*.

start-icd10-cm-inference-job

The following code example shows how to use `start-icd10-cm-inference-job`.

AWS CLI

To start an ICD-10-CM inference job

The following `start-icd10-cm-inference-job` example starts an ICD-10-CM inference batch analysis job.

```
aws comprehendmedical start-icd10-cm-inference-job \  
  --input-data-config "S3Bucket=comp-med-input" \  
  --output-data-config "S3Bucket=comp-med-output" \  
  --data-access-role-arn arn:aws:iam::867139942017:role/  
ComprehendMedicalBatchProcessingRole \  
  --language-code en
```

Output:

```
{
  "JobId": "ef7289877365fc70299089371c043b96"
}
```

For more information, see [Ontology linking batch analysis](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [StartIcd10CmInferenceJob](#) in *AWS CLI Command Reference*.

start-phi-detection-job

The following code example shows how to use `start-phi-detection-job`.

AWS CLI**To start a PHI detection job**

The following `start-phi-detection-job` example starts an asynchronous PHI entity detection job.

```
aws comprehendmedical start-phi-detection-job \
  --input-data-config "S3Bucket=comp-med-input" \
  --output-data-config "S3Bucket=comp-med-output" \
  --data-access-role-arn arn:aws:iam::867139942017:role/
ComprehendMedicalBatchProcessingRole \
  --language-code en
```

Output:

```
{
  "JobId": "ab9887877365fe70299089371c043b96"
}
```

For more information, see [Batch APIs](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [StartPhiDetectionJob](#) in *AWS CLI Command Reference*.

start-rx-norm-inference-job

The following code example shows how to use `start-rx-norm-inference-job`.

AWS CLI

To start an RxNorm inference job

The following `start-rx-norm-inference-job` example starts an RxNorm inference batch analysis job.

```
aws comprehendmedical start-rx-norm-inference-job \  
  --input-data-config "S3Bucket=comp-med-input" \  
  --output-data-config "S3Bucket=comp-med-output" \  
  --data-access-role-arn arn:aws:iam::867139942017:role/  
ComprehendMedicalBatchProcessingRole \  
  --language-code en
```

Output:

```
{  
  "JobId": "eg8199877365fc70299089371c043b96"  
}
```

For more information, see [Ontology linking batch analysis](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [StartRxNormInferenceJob](#) in *AWS CLI Command Reference*.

start-snomedct-inference-job

The following code example shows how to use `start-snomedct-inference-job`.

AWS CLI

To start an SNOMED CT inference job

The following `start-snomedct-inference-job` example starts a SNOMED CT inference batch analysis job.

```
aws comprehendmedical start-snomedct-inference-job \  
  --input-data-config "S3Bucket=comp-med-input" \  
  --output-data-config "S3Bucket=comp-med-output" \  
  --data-access-role-arn arn:aws:iam::867139942017:role/  
ComprehendMedicalBatchProcessingRole \  
  --language-code en
```

Output:

```
{
  "JobId": "dg7289877365fc70299089371c043b96"
}
```

For more information, see [Ontology linking batch analysis](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [StartSnomedctInferenceJob](#) in *AWS CLI Command Reference*.

stop-entities-detection-v2-job

The following code example shows how to use `stop-entities-detection-v2-job`.

AWS CLI**To stop an entity detection job**

The following `stop-entities-detection-v2-job` example stops an asynchronous entity detection job.

```
aws comprehendmedical stop-entities-detection-v2-job \
  --job-id "ab9887877365fe70299089371c043b96"
```

Output:

```
{
  "JobId": "ab9887877365fe70299089371c043b96"
}
```

For more information, see [Batch APIs](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [StopEntitiesDetectionV2Job](#) in *AWS CLI Command Reference*.

stop-icd10-cm-inference-job

The following code example shows how to use `stop-icd10-cm-inference-job`.

AWS CLI**To stop an ICD-10-CM inference job**

The following `stop-icd10-cm-inference-job` example stops an ICD-10-CM inference batch analysis job.

```
aws comprehendmedical stop-icd10-cm-inference-job \  
  --job-id "4750034166536cdb52ffa3295a1b00a3"
```

Output:

```
{  
  "JobId": "ef7289877365fc70299089371c043b96",  
}
```

For more information, see [Ontology linking batch analysis](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [StopIcd10CmInferenceJob](#) in *AWS CLI Command Reference*.

stop-phi-detection-job

The following code example shows how to use `stop-phi-detection-job`.

AWS CLI

To stop a protected health information (PHI) detection job

The following `stop-phi-detection-job` example stops an asynchronous protected health information (PHI) detection job.

```
aws comprehendmedical stop-phi-detection-job \  
  --job-id "4750034166536cdb52ffa3295a1b00a3"
```

Output:

```
{  
  "JobId": "ab9887877365fe70299089371c043b96"  
}
```

For more information, see [Batch APIs](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [StopPhiDetectionJob](#) in *AWS CLI Command Reference*.

stop-rx-norm-inference-job

The following code example shows how to use stop-rx-norm-inference-job.

AWS CLI

To stop an RxNorm inference job

The following stop-rx-norm-inference-job example stops an ICD-10-CM inference batch analysis job.

```
aws comprehendmedical stop-rx-norm-inference-job \  
  --job-id "eg8199877365fc70299089371c043b96"
```

Output:

```
{  
  "JobId": "eg8199877365fc70299089371c043b96",  
}
```

For more information, see [Ontology linking batch analysis](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [StopRxNormInferenceJob](#) in *AWS CLI Command Reference*.

stop-snomedct-inference-job

The following code example shows how to use stop-snomedct-inference-job.

AWS CLI

To stop a SNOMED CT inference job

The following stop-snomedct-inference-job example stops a SNOMED CT inference batch analysis job.

```
aws comprehendmedical stop-snomedct-inference-job \  
  --job-id "8750034166436cdb52ffa3295a1b00a1"
```

Output:


```
{
  "JobId": "8750034166436cdb52ffa3295a1b00a1",
}
```

For more information, see [Ontology linking batch analysis](#) in the *Amazon Comprehend Medical Developer Guide*.

- For API details, see [StopSnomedctInferenceJob](#) in *AWS CLI Command Reference*.

AWS Config examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Config.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

delete-config-rule

The following code example shows how to use `delete-config-rule`.

AWS CLI

To delete an AWS Config rule

The following command deletes an AWS Config rule named `MyConfigRule`:

```
aws configservice delete-config-rule --config-rule-name MyConfigRule
```

- For API details, see [DeleteConfigRule](#) in *AWS CLI Command Reference*.

delete-delivery-channel

The following code example shows how to use `delete-delivery-channel`.

AWS CLI

To delete a delivery channel

The following command deletes the default delivery channel:

```
aws configservice delete-delivery-channel --delivery-channel-name default
```

- For API details, see [DeleteDeliveryChannel](#) in *AWS CLI Command Reference*.

delete-evaluation-results

The following code example shows how to use `delete-evaluation-results`.

AWS CLI

To manually delete evaluation results

The following command deletes the current evaluation results for the AWS managed rule `s3-bucket-versioning-enabled`:

```
aws configservice delete-evaluation-results --config-rule-name s3-bucket-versioning-enabled
```

- For API details, see [DeleteEvaluationResults](#) in *AWS CLI Command Reference*.

deliver-config-snapshot

The following code example shows how to use `deliver-config-snapshot`.

AWS CLI

To deliver a configuration snapshot

The following command delivers a configuration snapshot to the Amazon S3 bucket that belongs to the default delivery channel:

```
aws configservice deliver-config-snapshot --delivery-channel-name default
```

Output:

```
{
  "configSnapshotId": "d0333b00-a683-44af-921e-examplefb794"
}
```

- For API details, see [DeliverConfigSnapshot](#) in *AWS CLI Command Reference*.

describe-compliance-by-config-rule

The following code example shows how to use `describe-compliance-by-config-rule`.

AWS CLI

To get compliance information for your AWS Config rules

The following command returns compliance information for each AWS Config rule that is violated by one or more AWS resources:

```
aws configservice describe-compliance-by-config-rule --compliance-types
NON_COMPLIANT
```

In the output, the value for each `CappedCount` attribute indicates how many resources do not comply with the related rule. For example, the following output indicates that 3 resources do not comply with the rule named `InstanceTypesAreT2micro`.

Output:

```
{
  "ComplianceByConfigRules": [
    {
      "Compliance": {
        "ComplianceContributorCount": {
          "CappedCount": 3,
          "CapExceeded": false
        }
      }
    }
  ]
}
```

```

        },
        "ComplianceType": "NON_COMPLIANT"
    },
    "ConfigRuleName": "InstanceTypesAreT2micro"
},
{
    "Compliance": {
        "ComplianceContributorCount": {
            "CappedCount": 10,
            "CapExceeded": false
        },
        "ComplianceType": "NON_COMPLIANT"
    },
    "ConfigRuleName": "RequiredTagsForVolumes"
}
]
}

```

- For API details, see [DescribeComplianceByConfigRule](#) in *AWS CLI Command Reference*.

describe-compliance-by-resource

The following code example shows how to use `describe-compliance-by-resource`.

AWS CLI

To get compliance information for your AWS resources

The following command returns compliance information for each EC2 instance that is recorded by AWS Config and that violates one or more rules:

```
aws configservice describe-compliance-by-resource --resource-type AWS::EC2::Instance
--compliance-types NON_COMPLIANT
```

In the output, the value for each `CappedCount` attribute indicates how many rules the resource violates. For example, the following output indicates that instance `i-1a2b3c4d` violates 2 rules.

Output:

```
{
  "ComplianceByResources": [
```

```

    {
      "ResourceType": "AWS::EC2::Instance",
      "ResourceId": "i-1a2b3c4d",
      "Compliance": {
        "ComplianceContributorCount": {
          "CappedCount": 2,
          "CapExceeded": false
        },
        "ComplianceType": "NON_COMPLIANT"
      }
    },
    {
      "ResourceType": "AWS::EC2::Instance",
      "ResourceId": "i-2a2b3c4d ",
      "Compliance": {
        "ComplianceContributorCount": {
          "CappedCount": 3,
          "CapExceeded": false
        },
        "ComplianceType": "NON_COMPLIANT"
      }
    }
  ]
}

```

- For API details, see [DescribeComplianceByResource](#) in *AWS CLI Command Reference*.

describe-config-rule-evaluation-status

The following code example shows how to use `describe-config-rule-evaluation-status`.

AWS CLI

To get status information for an AWS Config rule

The following command returns the status information for an AWS Config rule named `MyConfigRule`:

```
aws configservice describe-config-rule-evaluation-status --config-rule-names
MyConfigRule
```

Output:

```
{
  "ConfigRulesEvaluationStatus": [
    {
      "ConfigRuleArn": "arn:aws:config:us-east-1:123456789012:config-rule/
config-rule-abcdef",
      "FirstActivatedTime": 1450311703.844,
      "ConfigRuleId": "config-rule-abcdef",
      "LastSuccessfulInvocationTime": 1450314643.156,
      "ConfigRuleName": "MyConfigRule"
    }
  ]
}
```

- For API details, see [DescribeConfigRuleEvaluationStatus](#) in *AWS CLI Command Reference*.

describe-config-rules

The following code example shows how to use describe-config-rules.

AWS CLI

To get details for an AWS Config rule

The following command returns details for an AWS Config rule named InstanceTypesAreT2micro:

```
aws configservice describe-config-rules --config-rule-names InstanceTypesAreT2micro
```

Output:

```
{
  "ConfigRules": [
    {
      "ConfigRuleState": "ACTIVE",
      "Description": "Evaluates whether EC2 instances are the t2.micro type.",
      "ConfigRuleName": "InstanceTypesAreT2micro",
      "ConfigRuleArn": "arn:aws:config:us-east-1:123456789012:config-rule/
config-rule-abcdef",
      "Source": {
        "Owner": "CUSTOM_LAMBDA",
        "SourceIdentifier": "arn:aws:lambda:us-
east-1:123456789012:function:InstanceTypeCheck",

```

```
        "SourceDetails": [
            {
                "EventSource": "aws.config",
                "MessageType": "ConfigurationItemChangeNotification"
            }
        ],
        "InputParameters": "{\"desiredInstanceType\":\"t2.micro\"}",
        "Scope": {
            "ComplianceResourceTypes": [
                "AWS::EC2::Instance"
            ]
        },
        "ConfigRuleId": "config-rule-abcdef"
    }
]
```

- For API details, see [DescribeConfigRules](#) in *AWS CLI Command Reference*.

describe-configuration-recorder-status

The following code example shows how to use `describe-configuration-recorder-status`.

AWS CLI

To get status information for the configuration recorder

The following command returns the status of the default configuration recorder:

```
aws configservice describe-configuration-recorder-status
```

Output:

```
{
  "ConfigurationRecordersStatus": [
    {
      "name": "default",
      "lastStatus": "SUCCESS",
      "recording": true,
      "lastStatusChangeTime": 1452193834.344,
      "lastStartTime": 1441039997.819,
    }
  ]
}
```

```
        "lastStopTime": 1441039992.835
      }
    ]
  }
```

- For API details, see [DescribeConfigurationRecorderStatus](#) in *AWS CLI Command Reference*.

describe-configuration-recorders

The following code example shows how to use `describe-configuration-recorders`.

AWS CLI

To get details about the configuration recorder

The following command returns details about the default configuration recorder:

```
aws configservice describe-configuration-recorders
```

Output:

```
{
  "ConfigurationRecorders": [
    {
      "recordingGroup": {
        "allSupported": true,
        "resourceTypes": [],
        "includeGlobalResourceTypes": true
      },
      "roleARN": "arn:aws:iam::123456789012:role/config-ConfigRole-
A1B2C3D4E5F6",
      "name": "default"
    }
  ]
}
```

- For API details, see [DescribeConfigurationRecorders](#) in *AWS CLI Command Reference*.

describe-delivery-channel-status

The following code example shows how to use `describe-delivery-channel-status`.

AWS CLI

To get status information for the delivery channel

The following command returns the status of the delivery channel:

```
aws configservice describe-delivery-channel-status
```

Output:

```
{
  "DeliveryChannelsStatus": [
    {
      "configStreamDeliveryInfo": {
        "lastStatusChangeTime": 1452193834.381,
        "lastStatus": "SUCCESS"
      },
      "configHistoryDeliveryInfo": {
        "lastSuccessfulTime": 1450317838.412,
        "lastStatus": "SUCCESS",
        "lastAttemptTime": 1450317838.412
      },
      "configSnapshotDeliveryInfo": {
        "lastSuccessfulTime": 1452185597.094,
        "lastStatus": "SUCCESS",
        "lastAttemptTime": 1452185597.094
      },
      "name": "default"
    }
  ]
}
```

- For API details, see [DescribeDeliveryChannelStatus](#) in *AWS CLI Command Reference*.

describe-delivery-channels

The following code example shows how to use describe-delivery-channels.

AWS CLI

To get details about the delivery channel

The following command returns details about the delivery channel:

```
aws configservice describe-delivery-channels
```

Output:

```
{
  "DeliveryChannels": [
    {
      "snsTopicARN": "arn:aws:sns:us-east-1:123456789012:config-topic",
      "name": "default",
      "s3BucketName": "config-bucket-123456789012"
    }
  ]
}
```

- For API details, see [DescribeDeliveryChannels](#) in *AWS CLI Command Reference*.

get-compliance-details-by-config-rule

The following code example shows how to use `get-compliance-details-by-config-rule`.

AWS CLI

To get the evaluation results for an AWS Config rule

The following command returns the evaluation results for all of the resources that don't comply with an AWS Config rule named `InstanceTypesAreT2micro`:

```
aws configservice get-compliance-details-by-config-rule --config-rule-name
InstanceTypesAreT2micro --compliance-types NON_COMPLIANT
```

Output:

```
{
  "EvaluationResults": [
    {
      "EvaluationResultIdentifier": {
        "OrderingTimestamp": 1450314635.065,
        "EvaluationResultQualifier": {
```

```

        "ResourceType": "AWS::EC2::Instance",
        "ResourceId": "i-1a2b3c4d",
        "ConfigRuleName": "InstanceTypesAreT2micro"
    }
},
"ResultRecordedTime": 1450314645.261,
"ConfigRuleInvokedTime": 1450314642.948,
"ComplianceType": "NON_COMPLIANT"
},
{
    "EvaluationResultIdentifier": {
        "OrderingTimestamp": 1450314635.065,
        "EvaluationResultQualifier": {
            "ResourceType": "AWS::EC2::Instance",
            "ResourceId": "i-2a2b3c4d",
            "ConfigRuleName": "InstanceTypesAreT2micro"
        }
    },
    "ResultRecordedTime": 1450314645.18,
    "ConfigRuleInvokedTime": 1450314642.902,
    "ComplianceType": "NON_COMPLIANT"
},
{
    "EvaluationResultIdentifier": {
        "OrderingTimestamp": 1450314635.065,
        "EvaluationResultQualifier": {
            "ResourceType": "AWS::EC2::Instance",
            "ResourceId": "i-3a2b3c4d",
            "ConfigRuleName": "InstanceTypesAreT2micro"
        }
    },
    "ResultRecordedTime": 1450314643.346,
    "ConfigRuleInvokedTime": 1450314643.124,
    "ComplianceType": "NON_COMPLIANT"
}
]
}

```

- For API details, see [GetComplianceDetailsByConfigRule](#) in *AWS CLI Command Reference*.

get-compliance-details-by-resource

The following code example shows how to use `get-compliance-details-by-resource`.

AWS CLI

To get the evaluation results for an AWS resource

The following command returns the evaluation results for each rule with which the EC2 instance `i-1a2b3c4d` does not comply:

```
aws configservice get-compliance-details-by-resource --resource-type
AWS::EC2::Instance --resource-id i-1a2b3c4d --compliance-types NON_COMPLIANT
```

Output:

```
{
  "EvaluationResults": [
    {
      "EvaluationResultIdentifier": {
        "OrderingTimestamp": 1450314635.065,
        "EvaluationResultQualifier": {
          "ResourceType": "AWS::EC2::Instance",
          "ResourceId": "i-1a2b3c4d",
          "ConfigRuleName": "InstanceTypesAreT2micro"
        }
      },
      "ResultRecordedTime": 1450314643.288,
      "ConfigRuleInvokedTime": 1450314643.034,
      "ComplianceType": "NON_COMPLIANT"
    },
    {
      "EvaluationResultIdentifier": {
        "OrderingTimestamp": 1450314635.065,
        "EvaluationResultQualifier": {
          "ResourceType": "AWS::EC2::Instance",
          "ResourceId": "i-1a2b3c4d",
          "ConfigRuleName": "RequiredTagForEC2Instances"
        }
      },
      "ResultRecordedTime": 1450314645.261,
      "ConfigRuleInvokedTime": 1450314642.948,
      "ComplianceType": "NON_COMPLIANT"
    }
  ]
}
```

- For API details, see [GetComplianceDetailsByResource](#) in *AWS CLI Command Reference*.

get-compliance-summary-by-config-rule

The following code example shows how to use `get-compliance-summary-by-config-rule`.

AWS CLI

To get the compliance summary for your AWS Config rules

The following command returns the number of rules that are compliant and the number that are noncompliant:

```
aws configservice get-compliance-summary-by-config-rule
```

In the output, the value for each `CappedCount` attribute indicates how many rules are compliant or noncompliant.

Output:

```
{
  "ComplianceSummary": {
    "NonCompliantResourceCount": {
      "CappedCount": 3,
      "CapExceeded": false
    },
    "ComplianceSummaryTimestamp": 1452204131.493,
    "CompliantResourceCount": {
      "CappedCount": 2,
      "CapExceeded": false
    }
  }
}
```

- For API details, see [GetComplianceSummaryByConfigRule](#) in *AWS CLI Command Reference*.

get-compliance-summary-by-resource-type

The following code example shows how to use `get-compliance-summary-by-resource-type`.

AWS CLI

To get the compliance summary for all resource types

The following command returns the number of AWS resources that are noncompliant and the number that are compliant:

```
aws configservice get-compliance-summary-by-resource-type
```

In the output, the value for each CappedCount attribute indicates how many resources are compliant or noncompliant.

Output:

```
{
  "ComplianceSummariesByResourceType": [
    {
      "ComplianceSummary": {
        "NonCompliantResourceCount": {
          "CappedCount": 16,
          "CapExceeded": false
        },
        "ComplianceSummaryTimestamp": 1453237464.543,
        "CompliantResourceCount": {
          "CappedCount": 10,
          "CapExceeded": false
        }
      }
    }
  ]
}
```

To get the compliance summary for a specific resource type

The following command returns the number of EC2 instances that are noncompliant and the number that are compliant:

```
aws configservice get-compliance-summary-by-resource-type --resource-types
AWS::EC2::Instance
```

In the output, the value for each CappedCount attribute indicates how many resources are compliant or noncompliant.

Output:

```
{
  "ComplianceSummariesByResourceType": [
    {
      "ResourceType": "AWS::EC2::Instance",
      "ComplianceSummary": {
        "NonCompliantResourceCount": {
          "CappedCount": 3,
          "CapExceeded": false
        },
        "ComplianceSummaryTimestamp": 1452204923.518,
        "CompliantResourceCount": {
          "CappedCount": 7,
          "CapExceeded": false
        }
      }
    }
  ]
}
```

- For API details, see [GetComplianceSummaryByResourceType](#) in *AWS CLI Command Reference*.

get-resource-config-history

The following code example shows how to use `get-resource-config-history`.

AWS CLI**To get the configuration history of an AWS resource**

The following command returns a list of configuration items for an EC2 instance with an ID of `i-1a2b3c4d`:

```
aws configservice get-resource-config-history --resource-type AWS::EC2::Instance --
resource-id i-1a2b3c4d
```

- For API details, see [GetResourceConfigHistory](#) in *AWS CLI Command Reference*.

get-status

The following code example shows how to use `get-status`.

AWS CLI

To get the status for AWS Config

The following command returns the status of the delivery channel and configuration recorder:

```
aws configservice get-status
```

Output:

```
Configuration Recorders:

name: default
recorder: ON
last status: SUCCESS

Delivery Channels:

name: default
last stream delivery status: SUCCESS
last history delivery status: SUCCESS
last snapshot delivery status: SUCCESS
```

- For API details, see [GetStatus](#) in *AWS CLI Command Reference*.

list-discovered-resources

The following code example shows how to use `list-discovered-resources`.

AWS CLI

To list resources that AWS Config has discovered

The following command lists the EC2 instances that AWS Config has discovered:

```
aws configservice list-discovered-resources --resource-type AWS::EC2::Instance
```

Output:

```
{
  "resourceIdentifiers": [
```



```
{
  "resourceType": "AWS::EC2::Instance",
  "resourceId": "i-1a2b3c4d"
},
{
  "resourceType": "AWS::EC2::Instance",
  "resourceId": "i-2a2b3c4d"
},
{
  "resourceType": "AWS::EC2::Instance",
  "resourceId": "i-3a2b3c4d"
}
]
```

- For API details, see [ListDiscoveredResources](#) in *AWS CLI Command Reference*.

put-config-rule

The following code example shows how to use `put-config-rule`.

AWS CLI

To add an AWS managed Config rule

The following command provides JSON code to add an AWS managed Config rule:

```
aws configservice put-config-rule --config-rule file://
RequiredTagsForEC2Instances.json
```

`RequiredTagsForEC2Instances.json` is a JSON file that contains the rule configuration:

```
{
  "ConfigRuleName": "RequiredTagsForEC2Instances",
  "Description": "Checks whether the CostCenter and Owner tags are applied to EC2
instances.",
  "Scope": {
    "ComplianceResourceTypes": [
      "AWS::EC2::Instance"
    ]
  },
  "Source": {
```

```

    "Owner": "AWS",
    "SourceIdentifier": "REQUIRED_TAGS"
  },
  "InputParameters": "{\"tag1Key\": \"CostCenter\", \"tag2Key\": \"Owner\"}"
}

```

For the `ComplianceResourceTypes` attribute, this JSON code limits the scope to resources of the `AWS::EC2::Instance` type, so AWS Config will evaluate only EC2 instances against the rule. Because the rule is a managed rule, the `Owner` attribute is set to `AWS`, and the `SourceIdentifier` attribute is set to the rule identifier, `REQUIRED_TAGS`. For the `InputParameters` attribute, the tag keys that the rule requires, `CostCenter` and `Owner`, are specified.

If the command succeeds, AWS Config returns no output. To verify the rule configuration, run the `describe-config-rules` command, and specify the rule name.

To add a customer managed Config rule

The following command provides JSON code to add a customer managed Config rule:

```
aws configservice put-config-rule --config-rule file://InstanceTypesAreT2micro.json
```

`InstanceTypesAreT2micro.json` is a JSON file that contains the rule configuration:

```

{
  "ConfigRuleName": "InstanceTypesAreT2micro",
  "Description": "Evaluates whether EC2 instances are the t2.micro type.",
  "Scope": {
    "ComplianceResourceTypes": [
      "AWS::EC2::Instance"
    ]
  },
  "Source": {
    "Owner": "CUSTOM_LAMBDA",
    "SourceIdentifier": "arn:aws:lambda:us-east-1:123456789012:function:InstanceTypeCheck",
    "SourceDetails": [
      {
        "EventSource": "aws.config",
        "MessageType": "ConfigurationItemChangeNotification"
      }
    ]
  }
}

```

```
},  
  "InputParameters": "{\"desiredInstanceType\": \"t2.micro\"}"  
}
```

For the `ComplianceResourceTypes` attribute, this JSON code limits the scope to resources of the `AWS::EC2::Instance` type, so AWS Config will evaluate only EC2 instances against the rule. Because this rule is a customer managed rule, the `Owner` attribute is set to `CUSTOM_LAMBDA`, and the `SourceIdentifier` attribute is set to the ARN of the AWS Lambda function. The `SourceDetails` object is required. The parameters that are specified for the `InputParameters` attribute are passed to the AWS Lambda function when AWS Config invokes it to evaluate resources against the rule.

If the command succeeds, AWS Config returns no output. To verify the rule configuration, run the `describe-config-rules` command, and specify the rule name.

- For API details, see [PutConfigRule](#) in *AWS CLI Command Reference*.

put-configuration-recorder

The following code example shows how to use `put-configuration-recorder`.

AWS CLI

Example 1: To record all supported resources

The following command creates a configuration recorder that tracks changes to all supported resource types, including global resource types:

```
aws configservice put-configuration-recorder \  
  --configuration-recorder name=default,roleARN=arn:aws:iam::123456789012:role/  
config-role \  
  --recording-group allSupported=true,includeGlobalResourceTypes=true
```

If the command succeeds, AWS Config returns no output. To verify the settings of your configuration recorder, run the `describe-configuration-recorders` command.

Example 2: To record specific types of resources

The following command creates a configuration recorder that tracks changes to only those types of resources that are specified in the JSON file for the `--recording-group` option:

```
aws configservice put-configuration-recorder \  
  --configuration-recorder name=default,roleARN=arn:aws:iam::123456789012:role/  
config-role \  
  --recording-group file://recordingGroup.json
```

recordingGroup.json is a JSON file that specifies the types of resources that AWS Config will record:

```
{  
  "allSupported": false,  
  "includeGlobalResourceTypes": false,  
  "resourceTypes": [  
    "AWS::EC2::EIP",  
    "AWS::EC2::Instance",  
    "AWS::EC2::NetworkAcl",  
    "AWS::EC2::SecurityGroup",  
    "AWS::CloudTrail::Trail",  
    "AWS::EC2::Volume",  
    "AWS::EC2::VPC",  
    "AWS::IAM::User",  
    "AWS::IAM::Policy"  
  ]  
}
```

Before you can specify resource types for the resourceTypes key, you must set the allSupported and includeGlobalResourceTypes options to false or omit them.

If the command succeeds, AWS Config returns no output. To verify the settings of your configuration recorder, run the describe-configuration-recorders command.

Example 3: To select all supported resources excluding specific types of resources

The following command creates a configuration recorder that tracks changes to all current and future supported resource types excluding those types of resources that are specified in the JSON file for the --recording-group option:

```
aws configservice put-configuration-recorder \  
  --configuration-recorder name=default,roleARN=arn:aws:iam::123456789012:role/  
config-role \  
  --recording-group file://recordingGroup.json
```

recordingGroup.json is a JSON file that specifies the types of resources that AWS Config will record:

```
{
  "allSupported": false,
  "exclusionByResourceTypes": {
    "resourceTypes": [
      "AWS::Redshift::ClusterSnapshot",
      "AWS::RDS::DBClusterSnapshot",
      "AWS::CloudFront::StreamingDistribution"
    ]
  },
  "includeGlobalResourceTypes": false,
  "recordingStrategy": {
    "useOnly": "EXCLUSION_BY_RESOURCE_TYPES"
  },
}
```

Before you can specify resource types to excluding from recording: 1) You must set the `allSupported` and `includeGlobalResourceTypes` options to `false` or omit them, and 2) You must set the `useOnly` field of `RecordingStrategy` to `EXCLUSION_BY_RESOURCE_TYPES`.

If the command succeeds, AWS Config returns no output. To verify the settings of your configuration recorder, run the `describe-configuration-recorders` command.

- For API details, see [PutConfigurationRecorder](#) in *AWS CLI Command Reference*.

put-delivery-channel

The following code example shows how to use `put-delivery-channel`.

AWS CLI

To create a delivery channel

The following command provides the settings for the delivery channel as JSON code:

```
aws configservice put-delivery-channel --delivery-channel file://
deliveryChannel.json
```

The `deliveryChannel.json` file specifies the delivery channel attributes:

```
{
  "name": "default",
  "s3BucketName": "config-bucket-123456789012",
  "snsTopicARN": "arn:aws:sns:us-east-1:123456789012:config-topic",
  "configSnapshotDeliveryProperties": {
    "deliveryFrequency": "Twelve_Hours"
  }
}
```

This example sets the following attributes:

`name` - The name of the delivery channel. By default, AWS Config assigns the name `default` to a new delivery channel. You cannot update the delivery channel name with the `put-delivery-channel` command. For the steps to change the name, see [Renaming the Delivery Channel](#). `s3BucketName` - The name of the Amazon S3 bucket to which AWS Config delivers configuration snapshots and configuration history files. If you specify a bucket that belongs to another AWS account, that bucket must have policies that grant access permissions to AWS Config. For more information, see [Permissions for the Amazon S3 Bucket](#).

`snsTopicARN` - The Amazon Resource Name (ARN) of the Amazon SNS topic to which AWS Config sends notifications about configuration changes. If you choose a topic from another account, the topic must have policies that grant access permissions to AWS Config. For more information, see [Permissions for the Amazon SNS Topic](#).

`configSnapshotDeliveryProperties` - Contains the `deliveryFrequency` attribute, which sets how often AWS Config delivers configuration snapshots and how often it invokes evaluations for periodic Config rules.

If the command succeeds, AWS Config returns no output. To verify the settings of your delivery channel, run the `describe-delivery-channels` command.

- For API details, see [PutDeliveryChannel](#) in *AWS CLI Command Reference*.

start-config-rules-evaluation

The following code example shows how to use `start-config-rules-evaluation`.

AWS CLI

To run an on-demand evaluation for AWS Config rules

The following command starts an evaluation for two AWS managed rules:

```
aws configservice start-config-rules-evaluation --config-rule-names s3-bucket-  
versioning-enabled cloudtrail-enabled
```

- For API details, see [StartConfigRulesEvaluation](#) in *AWS CLI Command Reference*.

start-configuration-recorder

The following code example shows how to use `start-configuration-recorder`.

AWS CLI

To start the configuration recorder

The following command starts the default configuration recorder:

```
aws configservice start-configuration-recorder --configuration-recorder-name default
```

If the command succeeds, AWS Config returns no output. To verify that AWS Config is recording your resources, run the `get-status` command.

- For API details, see [StartConfigurationRecorder](#) in *AWS CLI Command Reference*.

stop-configuration-recorder

The following code example shows how to use `stop-configuration-recorder`.

AWS CLI

To stop the configuration recorder

The following command stops the default configuration recorder:

```
aws configservice stop-configuration-recorder --configuration-recorder-name default
```

If the command succeeds, AWS Config returns no output. To verify that AWS Config is not recording your resources, run the `get-status` command.

- For API details, see [StopConfigurationRecorder](#) in *AWS CLI Command Reference*.

subscribe

The following code example shows how to use `subscribe`.

AWS CLI

To subscribe to AWS Config

The following command creates the default delivery channel and configuration recorder. The command also specifies the Amazon S3 bucket and Amazon SNS topic to which AWS Config will deliver configuration information:

```
aws configservice subscribe --s3-bucket config-bucket-123456789012 --
sns-topic arn:aws:sns:us-east-1:123456789012:config-topic --iam-role
arn:aws:iam::123456789012:role/ConfigRole-A1B2C3D4E5F6
```

Output:

```
Using existing S3 bucket: config-bucket-123456789012
Using existing SNS topic: arn:aws:sns:us-east-1:123456789012:config-topic
Subscribe succeeded:

Configuration Recorders: [
  {
    "recordingGroup": {
      "allSupported": true,
      "resourceTypes": [],
      "includeGlobalResourceTypes": false
    },
    "roleARN": "arn:aws:iam::123456789012:role/ConfigRole-A1B2C3D4E5F6",
    "name": "default"
  }
]

Delivery Channels: [
  {
    "snsTopicARN": "arn:aws:sns:us-east-1:123456789012:config-topic",
    "name": "default",
    "s3BucketName": "config-bucket-123456789012"
  }
]
```

- For API details, see [Subscribe](#) in *AWS CLI Command Reference*.

Amazon Connect examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon Connect.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-user

The following code example shows how to use `create-user`.

AWS CLI

To create a user

The following `create-user` example adds a user with the specified attributes to the specified Amazon Connect instance.

```
aws connect create-user \  
  --username Mary \  
  --password Pass@Word1 \  
  --identity-info FirstName=Mary,LastName=Major \  
  --phone-config  
  PhoneType=DESK_PHONE,AutoAccept=true,AfterContactWorkTimeLimit=60,DeskPhoneNumber=  
+15555551212 \  
  --security-profile-id 12345678-1111-2222-aaaa-a1b2c3d4f5g7 \  
  --routing-profile-id 87654321-9999-3434-abcd-x1y2z3a1b2c3 \  
  --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{
  "UserId": "87654321-2222-1234-1234-111234567891",
  "UserArn": "arn:aws:connect:us-west-2:123456789012:instance/a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111/agent/87654321-2222-1234-1234-111234567891"
}
```

For more information, see [Add Users](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [CreateUser](#) in *AWS CLI Command Reference*.

delete-user

The following code example shows how to use `delete-user`.

AWS CLI**To delete a user**

The following `delete-user` example deletes the specified user from the specified Amazon Connect instance.

```
aws connect delete-user \
  --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
  --user-id 87654321-2222-1234-1234-111234567891
```

This command produces no output.

For more information, see [Manage Users](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [DeleteUser](#) in *AWS CLI Command Reference*.

describe-user-hierarchy-group

The following code example shows how to use `describe-user-hierarchy-group`.

AWS CLI**To display the details for a hierarchy group**

The following `describe-user-hierarchy-group` example displays the details for the specified Amazon Connect hierarchy group.

```
aws connect describe-user-hierarchy-group \
  --hierarchy-group-id 12345678-1111-2222-800e-aaabbb555gg \
  --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{
  "HierarchyGroup": {
    "Id": "12345678-1111-2222-800e-a2b3c4d5f6g7",
    "Arn": "arn:aws:connect:us-west-2:123456789012:instance/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/agent-group/12345678-1111-2222-800e-a2b3c4d5f6g7",
    "Name": "Example Corporation",
    "LevelId": "1",
    "HierarchyPath": {
      "LevelOne": {
        "Id": "abcdefgh-3333-4444-8af3-201123456789",
        "Arn": "arn:aws:connect:us-west-2:123456789012:instance/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/agent-group/abcdefgh-3333-4444-8af3-201123456789",
        "Name": "Example Corporation"
      }
    }
  }
}
```

For more information, see [Set Up Agent Hierarchies](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [DescribeUserHierarchyGroup](#) in *AWS CLI Command Reference*.

describe-user-hierarchy-structure

The following code example shows how to use `describe-user-hierarchy-structure`.

AWS CLI

To display the details for a hierarchy structure

The following `describe-user-hierarchy-structure` example displays the details for the hierarchy structure for the specified Amazon Connect instance.

```
aws connect describe-user-hierarchy-group \
  --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{
  "HierarchyStructure": {
    "LevelOne": {
      "Id": "12345678-1111-2222-800e-aaabbb555gg",
      "Arn": "arn:aws:connect:us-west-2:123456789012:instance/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/agent-group-level/1",
      "Name": "Corporation"
    },
    "LevelTwo": {
      "Id": "87654321-2222-3333-ac99-123456789102",
      "Arn": "arn:aws:connect:us-west-2:123456789012:instance/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/agent-group-level/2",
      "Name": "Services Division"
    },
    "LevelThree": {
      "Id": "abcdefgh-3333-4444-8af3-201123456789",
      "Arn": "arn:aws:connect:us-west-2:123456789012:instance/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/agent-group-level/3",
      "Name": "EU Site"
    }
  }
}
```

For more information, see [Set Up Agent Hierarchies](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [DescribeUserHierarchyStructure](#) in *AWS CLI Command Reference*.

describe-user

The following code example shows how to use `describe-user`.

AWS CLI**To display the details for a user**

The following `describe-user` example displays the details for the specified Amazon Connect user.

```
aws connect describe-user \
  --user-id 0c245dc0-0cf5-4e37-800e-2a7481cc8a60
  --instance-id 40c83b68-ea62-414c-97bb-d018e39e158e
```

Output:

```
{
  "User": {
    "Id": "0c245dc0-0cf5-4e37-800e-2a7481cc8a60",
    "Arn": "arn:aws:connect:us-west-2:123456789012:instance/40c83b68-
ea62-414c-97bb-d018e39e158e/agent/0c245dc0-0cf5-4e37-800e-2a7481cc8a60",
    "Username": "Jane",
    "IdentityInfo": {
      "FirstName": "Jane",
      "LastName": "Doe",
      "Email": "example.com"
    },
    "PhoneConfig": {
      "PhoneType": "SOFT_PHONE",
      "AutoAccept": false,
      "AfterContactWorkTimeLimit": 0,
      "DeskPhoneNumber": ""
    },
    "DirectoryUserId": "8b444cf6-b368-4f29-ba18-07af27405658",
    "SecurityProfileIds": [
      "b6f85a42-1dc5-443b-b621-de0abf70c9cf"
    ],
    "RoutingProfileId": "0be36ee9-2b5f-4ef4-bcf7-87738e5be0e5",
    "Tags": {}
  }
}
```

For more information, see [Manage Users](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [DescribeUser](#) in *AWS CLI Command Reference*.

get-contact-attributes

The following code example shows how to use `get-contact-attributes`.

AWS CLI**To retrieve the attributes for a contact**

The following `get-contact-attributes` example retrieves the attributes that were set for the specified Amazon Connect contact.

```
aws connect get-contact-attributes \  
  --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \  
  --initial-contact-id 12345678-1111-2222-800e-a2b3c4d5f6g7
```

Output:

```
{  
  "Attributes": {  
    "greetingPlayed": "true"  
  }  
}
```

For more information, see [Use Amazon Connect Contact Attributes](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [GetContactAttributes](#) in *AWS CLI Command Reference*.

list-contact-flows

The following code example shows how to use `list-contact-flows`.

AWS CLI

To list the contact flows in an instance

The following `list-contact-flows` example lists the contact flows in the specified Amazon Connect instance.

```
aws connect list-contact-flows \  
  --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{  
  "ContactFlowSummaryList": [  
    {  
      "Id": "12345678-1111-2222-800e-a2b3c4d5f6g7",  
      "Arn": "arn:aws:connect:us-west-2:123456789012:instance/  
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/contact-flow/12345678-1111-2222-800e-  
a2b3c4d5f6g7",  
      "Name": "Default queue transfer",  
    }  
  ]  
}
```

```

        "ContactFlowType": "QUEUE_TRANSFER"
    },
    {
        "Id": "87654321-2222-3333-ac99-123456789102",
        "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/contact-flow/87654321-2222-3333-
ac99-123456789102",
        "Name": "Default agent hold",
        "ContactFlowType": "AGENT_HOLD"
    },
    {
        "Id": "abcdefgh-3333-4444-8af3-201123456789",
        "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/contact-flow/
abcdefgh-3333-4444-8af3-201123456789",
        "Name": "Default customer hold",
        "ContactFlowType": "CUSTOMER_HOLD"
    },
]
}

```

For more information, see [Create Amazon Connect Contact Flows](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [ListContactFlows](#) in *AWS CLI Command Reference*.

list-hours-of-operations

The following code example shows how to use `list-hours-of-operations`.

AWS CLI

To list the hours of operation in an instance

The following `list-hours-of-operations` example lists the hours of operations for the specified Amazon Connect instance.

```
aws connect list-hours-of-operations \
  --instance-id 40c83b68-ea62-414c-97bb-d018e39e158e
```

Output:

```
{
```

```
"HoursOfOperationSummaryList": [  
  {  
    "Id": "d69f1f84-7457-4924-8fbe-e64875546259",  
    "Arn": "arn:aws:connect:us-west-2:123456789012:instance/40c83b68-  
ea62-414c-97bb-d018e39e158e/operating-hours/d69f1f84-7457-4924-8fbe-e64875546259",  
    "Name": "Basic Hours"  
  }  
]  
}
```

For more information, see [Set the Hours of Operation for a Queue](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [ListHoursOfOperations](#) in *AWS CLI Command Reference*.

list-phone-numbers

The following code example shows how to use `list-phone-numbers`.

AWS CLI

To list the phone numbers in an instance

The following `list-phone-numbers` example lists the phone numbers in the specified Amazon Connect instance.

```
aws connect list-phone-numbers \  
  --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{  
  "PhoneNumberSummaryList": [  
    {  
      "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
      "Arn": "arn:aws:connect:us-west-2:123456789012:instance/  
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/phone-number/xyz80zxy-xyz1-80zx-  
zx80-11111EXAMPLE",  
      "PhoneNumber": "+17065551212",  
      "PhoneNumberType": "DID",  
      "PhoneNumberCountryCode": "US"  
    },  
  ],  
}
```



```
{
  "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE2222",
  "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/phone-number/ccc0ccc-xyz1-80zx-
zx80-22222EXAMPLE",
  "PhoneNumber": "+18555551212",
  "PhoneNumberType": "TOLL_FREE",
  "PhoneNumberCountryCode": "US"
}
]
```

For more information, see [Set Up Phone Numbers for Your Contact Center](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [ListPhoneNumbers](#) in *AWS CLI Command Reference*.

list-queues

The following code example shows how to use `list-queues`.

AWS CLI

To list the queues in an instance

The following `list-queues` example lists the queues in the specified Amazon Connect instance.

```
aws connect list-queues \
  --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{
  "QueueSummaryList": [
    {
      "Id": "12345678-1111-2222-800e-a2b3c4d5f6g7",
      "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/queue/agent/12345678-1111-2222-800e-
a2b3c4d5f6g7",
      "QueueType": "AGENT"
    },
  ],
}
```

```
{
  "Id": "87654321-2222-3333-ac99-123456789102",
  "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/queue/agent/87654321-2222-3333-
ac99-123456789102",
  "QueueType": "AGENT"
},
{
  "Id": "abcdefgh-3333-4444-8af3-201123456789",
  "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/queue/agent/
abcdefgh-3333-4444-8af3-201123456789",
  "QueueType": "AGENT"
},
{
  "Id": "hgfedcba-4444-5555-a31f-123456789102",
  "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/queue/hgfedcba-4444-5555-a31f-123456789102",
  "Name": "BasicQueue",
  "QueueType": "STANDARD"
},
]
}
```

For more information, see [Create a Queue](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [ListQueues](#) in *AWS CLI Command Reference*.

list-routing-profiles

The following code example shows how to use `list-routing-profiles`.

AWS CLI

To list the routing profiles in an instance

The following `list-routing-profiles` example lists the routing profiles in the specified Amazon Connect instance.

```
aws connect list-routing-profiles \
  --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{
  "RoutingProfileSummaryList": [
    {
      "Id": "12345678-1111-2222-800e-a2b3c4d5f6g7",
      "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/routing-profile/12345678-1111-2222-800e-
a2b3c4d5f6g7",
      "Name": "Basic Routing Profile"
    },
  ]
}
```

For more information, see [Create a Routing Profile](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [ListRoutingProfiles](#) in *AWS CLI Command Reference*.

list-security-profiles

The following code example shows how to use `list-security-profiles`.

AWS CLI

To list the security profiles in an instance

The following `list-security-profiles` example lists the security profiles in the specified Amazon Connect instance.

```
aws connect list-security-profiles \
  --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{
  "SecurityProfileSummaryList": [
    {
      "Id": "12345678-1111-2222-800e-a2b3c4d5f6g7",
      "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/security-profile/12345678-1111-2222-800e-
a2b3c4d5f6g7",
      "Name": "CallCenterManager"
    },
    {
```

```

        "Id": "87654321-2222-3333-ac99-123456789102",
        "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/security-profile/87654321-2222-3333-
ac99-123456789102",
        "Name": "QualityAnalyst"
    },
    {
        "Id": "abcdefgh-3333-4444-8af3-201123456789",
        "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/security-profile/
abcdefgh-3333-4444-8af3-201123456789",
        "Name": "Agent"
    },
    {
        "Id": "12345678-1111-2222-800e-x2y3c4d5fzzzz",
        "Arn": "arn:aws:connect:us-west-2:123456789012:instance/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/security-profile/12345678-1111-2222-800e-
x2y3c4d5fzzzz",
        "Name": "Admin"
    }
]
}

```

For more information, see [Assign Permissions: Security Profiles](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [ListSecurityProfiles](#) in *AWS CLI Command Reference*.

list-user-hierarchy-groups

The following code example shows how to use `list-user-hierarchy-groups`.

AWS CLI

To list the user hierarchy groups in an instance

The following `list-user-hierarchy-groups` example lists the user hierarchy groups in the specified Amazon Connect instance.

```
aws connect list-user-hierarchy-groups \
  --instance-id 40c83b68-ea62-414c-97bb-d018e39e158e
```

Output:

```
{
  "UserHierarchyGroupSummaryList": [
    {
      "Id": "0e2f6d1d-b3ca-494b-8dbc-ba81d9f8182a",
      "Arn": "arn:aws:connect:us-west-2:123456789012:instance/40c83b68-
ea62-414c-97bb-d018e39e158e/agent-group/0e2f6d1d-b3ca-494b-8dbc-ba81d9f8182a",
      "Name": "Example Corporation"
    },
  ]
}
```

For more information, see [Set Up Agent Hierarchies](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [ListUserHierarchyGroups](#) in *AWS CLI Command Reference*.

list-users

The following code example shows how to use `list-users`.

AWS CLI

To list the user hierarchy groups in an instance

The following `list-users` example lists the users in the specified Amazon Connect instance.

```
aws connect list-users \
  --instance-id 40c83b68-ea62-414c-97bb-d018e39e158e
```

Output:

```
{
  "UserSummaryList": [
    {
      "Id": "0c245dc0-0cf5-4e37-800e-2a7481cc8a60",
      "Arn": "arn:aws:connect:us-west-2:123456789012:instance/40c83b68-
ea62-414c-97bb-d018e39e158e/agent/0c245dc0-0cf5-4e37-800e-2a7481cc8a60",
      "Username": "Jane"
    },
    {
      "Id": "46f0c67c-3fc7-4806-ac99-403798788c14",
      "Arn": "arn:aws:connect:us-west-2:123456789012:instance/40c83b68-
ea62-414c-97bb-d018e39e158e/agent/46f0c67c-3fc7-4806-ac99-403798788c14",
    }
  ]
}
```

```

        "Username": "Paulo"
    },
    {
        "Id": "55a83578-95e1-4710-8af3-2b7afe310e48",
        "Arn": "arn:aws:connect:us-west-2:123456789012:instance/40c83b68-
ea62-414c-97bb-d018e39e158e/agent/55a83578-95e1-4710-8af3-2b7afe310e48",
        "Username": "JohnD"
    },
    {
        "Id": "703e27b5-c9f0-4f1f-a239-64ccbb160125",
        "Arn": "arn:aws:connect:us-west-2:123456789012:instance/40c83b68-
ea62-414c-97bb-d018e39e158e/agent/703e27b5-c9f0-4f1f-a239-64ccbb160125",
        "Username": "JohnS"
    }
]
}

```

For more information, see [Add Users](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [ListUsers](#) in *AWS CLI Command Reference*.

update-contact-attributes

The following code example shows how to use `update-contact-attributes`.

AWS CLI

To update a contact's attribute

The following `update-contact-attributes` example updates the `greetingPlayed` attribute for the specified Amazon Connect user.

```

aws connect update-contact-attributes \
  --initial-contact-id 11111111-2222-3333-4444-12345678910 \
  --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
  --attributes greetingPlayed=false

```

This command produces no output.

For more information, see [Use Amazon Connect Contact Attributes](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [UpdateContactAttributes](#) in *AWS CLI Command Reference*.

update-user-hierarchy

The following code example shows how to use `update-user-hierarchy`.

AWS CLI

To update a user's hierarchy

The following `update-user-hierarchy` example updates the agent hierarchy for the specified Amazon Connect user.

```
aws connect update-user-hierarchy \  
  --hierarchy-group-id 12345678-a1b2-c3d4-e5f6-123456789abc \  
  --user-id 87654321-2222-1234-1234-111234567891 \  
  --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

This command produces no output.

For more information, see [Configure Agent Settings](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [UpdateUserHierarchy](#) in *AWS CLI Command Reference*.

update-user-identity-info

The following code example shows how to use `update-user-identity-info`.

AWS CLI

To update a user's identity information

The following `update-user-identity-info` example updates the identity information for the specified Amazon Connect user.

```
aws connect update-user-identity-info \  
  --identity-info FirstName=Mary,LastName=Major,Email=marym@example.com \  
  --user-id 87654321-2222-1234-1234-111234567891 \  
  --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

This command produces no output.

For more information, see [Configure Agent Settings](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [UpdateUserIdentityInfo](#) in *AWS CLI Command Reference*.

update-user-phone-config

The following code example shows how to use `update-user-phone-config`.

AWS CLI

To update a user's phone configuration

The following `update-user-phone-config` example updates the phone configuration for the specified user.

```
aws connect update-user-phone-config \  
  --phone-config  
  PhoneType=SOFT_PHONE,AutoAccept=false,AfterContactWorkTimeLimit=60,DeskPhoneNumber=  
+18005551212 \  
  --user-id 12345678-4444-3333-2222-111122223333 \  
  --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

This command produces no output.

For more information, see [Configure Agent Settings](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [UpdateUserPhoneConfig](#) in *AWS CLI Command Reference*.

update-user-routing-profile

The following code example shows how to use `update-user-routing-profile`.

AWS CLI

To update a user's routing profile

The following `update-user-routing-profile` example updates the routing profile for the specified Amazon Connect user.

```
aws connect update-user-routing-profile \  
  --routing-profile-id 12345678-1111-3333-2222-4444EXAMPLE \  
  --user-id 87654321-2222-1234-1234-111234567891 \  
  --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

This command produces no output.

For more information, see [Configure Agent Settings](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [UpdateUserRoutingProfile](#) in *AWS CLI Command Reference*.

update-user-security-profiles

The following code example shows how to use `update-user-security-profiles`.

AWS CLI

To update a user's security profiles

The following `update-user-security-profiles` example updates the security profile for the specified Amazon Connect user.

```
aws connect update-user-security-profiles \  
  --security-profile-ids 12345678-1234-1234-1234-1234567892111 \  
  --user-id 87654321-2222-1234-1234-111234567891 \  
  --instance-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

This command produces no output.

For more information, see [Assign Permissions: Security Profiles](#) in the *Amazon Connect Administrator Guide*.

- For API details, see [UpdateUserSecurityProfiles](#) in *AWS CLI Command Reference*.

AWS Cost and Usage Report examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Cost and Usage Report.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

delete-report-definition

The following code example shows how to use `delete-report-definition`.

AWS CLI

To delete an AWS Cost and Usage Report

This example deletes an AWS Cost and Usage Report.

Command:

```
aws cur --region us-east-1 delete-report-definition --report-name "ExampleReport"
```

- For API details, see [DeleteReportDefinition](#) in *AWS CLI Command Reference*.

describe-report-definitions

The following code example shows how to use `describe-report-definitions`.

AWS CLI

To retrieve a list of AWS Cost and Usage Reports

This example describes a list of AWS Cost and Usage Reports owned by an account.

Command:

```
aws cur --region us-east-1 describe-report-definitions --max-items 5
```

Output:

```
{
  "ReportDefinitions": [
    {
      "ReportName": "ExampleReport",
```

```

    "Compression": "ZIP",
    "S3Region": "us-east-1",
    "Format": "textORcsv",
    "S3Prefix": "exampleprefix",
    "S3Bucket": "example-s3-bucket",
    "TimeUnit": "DAILY",
    "AdditionalArtifacts": [
        "REDSHIFT",
        "QUICKSIGHT"
    ],
    "AdditionalSchemaElements": [
        "RESOURCES"
    ]
  }
]
}
```

- For API details, see [DescribeReportDefinitions](#) in *AWS CLI Command Reference*.

put-report-definition

The following code example shows how to use `put-report-definition`.

AWS CLI

To create an AWS Cost and Usage Reports

The following `put-report-definition` example creates a daily AWS Cost and Usage Report that you can upload into Amazon Redshift or Amazon QuickSight.

```
aws cur put-report-definition --report-definition file://report-definition.json
```

Contents of `report-definition.json`:

```

{
  "ReportName": "ExampleReport",
  "TimeUnit": "DAILY",
  "Format": "textORcsv",
  "Compression": "ZIP",
  "AdditionalSchemaElements": [
    "RESOURCES"
  ],
}
```

```
"S3Bucket": "example-s3-bucket",
"S3Prefix": "exampleprefix",
"S3Region": "us-east-1",
"AdditionalArtifacts": [
  "REDSHIFT",
  "QUICKSIGHT"
]
```

- For API details, see [PutReportDefinition](#) in *AWS CLI Command Reference*.

Cost Explorer Service examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Cost Explorer Service.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

get-cost-and-usage

The following code example shows how to use `get-cost-and-usage`.

AWS CLI

To retrieve the S3 usage of an account for the month of September 2017

The following `get-cost-and-usage` example retrieves the S3 usage of an account for the month of September 2017.

```
aws ce get-cost-and-usage \  
  --time-period Start=2017-09-01,End=2017-10-01 \  
  --granularity MONTHLY \  
  --metrics "BlendedCost" "UnblendedCost" "UsageQuantity" \  
  --group-by Type=DIMENSION,Key=SERVICE Type=TAG,Key=Environment \  
  --filter file://filters.json
```

Contents of filters.json:

```
{  
  "Dimensions": {  
    "Key": "SERVICE",  
    "Values": [  
      "Amazon Simple Storage Service"  
    ]  
  }  
}
```

Output:

```
{  
  "GroupDefinitions": [  
    {  
      "Type": "DIMENSION",  
      "Key": "SERVICE"  
    },  
    {  
      "Type": "TAG",  
      "Key": "Environment"  
    }  
  ],  
  "ResultsByTime": [  
    {  
      "Estimated": false,  
      "TimePeriod": {  
        "Start": "2017-09-01",  
        "End": "2017-10-01"  
      },  
      "Total": {},  
      "Groups": [  
        {  
          "Keys": [  

```


get-dimension-values

The following code example shows how to use `get-dimension-values`.

AWS CLI

To retrieve the tags for the dimension SERVICE, with a value of "Elastic"

This example retrieves the tags for the dimension SERVICE, with a value of "Elastic" for January 01 2017 through May 18 2017.

Command:

```
aws ce get-dimension-values --search-string Elastic --time-period
Start=2017-01-01,End=2017-05-18 --dimension SERVICE
```

Output:

```
{
  "TotalSize": 6,
  "DimensionValues": [
    {
      "Attributes": {},
      "Value": "Amazon ElastiCache"
    },
    {
      "Attributes": {},
      "Value": "EC2 - Other"
    },
    {
      "Attributes": {},
      "Value": "Amazon Elastic Compute Cloud - Compute"
    },
    {
      "Attributes": {},
      "Value": "Amazon Elastic Load Balancing"
    },
    {
      "Attributes": {},
      "Value": "Amazon Elastic MapReduce"
    },
    {
      "Attributes": {},
```

```

        "Value": "Amazon Elasticsearch Service"
    }
],
"ReturnSize": 6
}

```

- For API details, see [GetDimensionValues](#) in *AWS CLI Command Reference*.

get-reservation-coverage

The following code example shows how to use `get-reservation-coverage`.

AWS CLI

To retrieve the reservation coverage for EC2 t2.nano instances in the us-east-1 region

This example retrieves the reservation coverage for EC2 t2.nano instances in the us-east-1 region for July-September of 2017.

Command:

```
aws ce get-reservation-coverage --time-period Start=2017-07-01,End=2017-10-01 --
group-by Type=Dimension,Key=REGION --filter file://filters.json
```

filters.json:

```

{
  "And": [
    {
      "Dimensions": {
        "Key": "INSTANCE_TYPE",
        "Values": [
          "t2.nano"
        ]
      },
      "Dimensions": {
        "Key": "REGION",
        "Values": [
          "us-east-1"
        ]
      }
    }
  ]
}

```



```

    }
  ]
}

```

Output:

```

{
  "TotalSize": 6,
  "DimensionValues": [
    {
      "Attributes": {},
      "Value": "Amazon ElastiCache"
    },
    {
      "Attributes": {},
      "Value": "EC2 - Other"
    },
    {
      "Attributes": {},
      "Value": "Amazon Elastic Compute Cloud - Compute"
    },
    {
      "Attributes": {},
      "Value": "Amazon Elastic Load Balancing"
    },
    {
      "Attributes": {},
      "Value": "Amazon Elastic MapReduce"
    },
    {
      "Attributes": {},
      "Value": "Amazon Elasticsearch Service"
    }
  ],
  "ReturnSize": 6
}

```

- For API details, see [GetReservationCoverage](#) in *AWS CLI Command Reference*.

get-reservation-purchase-recommendation

The following code example shows how to use `get-reservation-purchase-recommendation`.

AWS CLI

To retrieve the reservation recommendations for Partial Upfront EC2 RIs with a three year term

The following `get-reservation-purchase-recommendation` example retrieves recommendations for Partial Upfront EC2 instances with a three-year term, based on the last 60 days of EC2 usage.

```
aws ce get-reservation-purchase-recommendation \  
  --service "Amazon Redshift" \  
  --lookback-period-in-days SIXTY_DAYS \  
  --term-in-years THREE_YEARS \  
  --payment-option PARTIAL_UPFRONT
```

Output:

```
{  
  "Recommendations": [],  
  "Metadata": {  
    "GenerationTimestamp": "2018-08-08T15:20:57Z",  
    "RecommendationId": "00d59dde-a1ad-473f-8ff2-iexample3330b"  
  }  
}
```

- For API details, see [GetReservationPurchaseRecommendation](#) in *AWS CLI Command Reference*.

get-reservation-utilization

The following code example shows how to use `get-reservation-utilization`.

AWS CLI

To retrieve the reservation utilization for your account

The following `get-reservation-utilization` example retrieves the RI utilization for all `t2.nano` instance types from 2018-03-01 to 2018-08-01 for the account.

```
aws ce get-reservation-utilization \  
  --time-period Start=2018-03-01,End=2018-08-01 \  
  --filter file://filters.json
```

Contents of `filters.json`:

```
{
  "Dimensions": {
    "Key": "INSTANCE_TYPE",
    "Values": [
      "t2.nano"
    ]
  }
}
```

Output:

```
{
  "Total": {
    "TotalAmortizedFee": "0",
    "UtilizationPercentage": "0",
    "PurchasedHours": "0",
    "NetRISavings": "0",
    "TotalActualHours": "0",
    "AmortizedRecurringFee": "0",
    "UnusedHours": "0",
    "TotalPotentialRISavings": "0",
    "OnDemandCostOfRIHoursUsed": "0",
    "AmortizedUpfrontFee": "0"
  },
  "UtilizationsByTime": []
}
```

- For API details, see [GetReservationUtilization](#) in *AWS CLI Command Reference*.

get-tags

The following code example shows how to use `get-tags`.

AWS CLI

To retrieve keys and values for a cost allocation tag

This example retrieves all cost allocation tags with a key of "Project" and a value that contains "secretProject".

Command:

```
aws ce get-tags --search-string secretProject --time-period
Start=2017-01-01,End=2017-05-18 --tag-key Project
```

Output:

```
{
  "ReturnSize": 2,
  "Tags": [
    "secretProject1",
    "secretProject2"
  ],
  "TotalSize": 2
}
```

- For API details, see [GetTags](#) in *AWS CLI Command Reference*.

Firehose examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Firehose.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

list-delivery-streams

The following code example shows how to use `list-delivery-streams`.

AWS CLI

To list the available delivery streams

The following `list-delivery-streams` example lists the available delivery streams in your AWS account.

```
aws firehose list-delivery-streams
```

Output:

```
{
  "DeliveryStreamNames": [
    "my-stream"
  ],
  "HasMoreDeliveryStreams": false
}
```

For more information, see [Creating an Amazon Kinesis Data Firehose Delivery Stream](#) in the *Amazon Kinesis Data Firehose Developer Guide*.

- For API details, see [ListDeliveryStreams](#) in *AWS CLI Command Reference*.

put-record-batch

The following code example shows how to use `put-record-batch`.

AWS CLI

To write multiple records to a stream

The following `put-record-batch` example writes three records to a stream. The data is encoded in Base64 format.

```
aws firehose put-record-batch \
  --delivery-stream-name my-stream \
  --records file://records.json
```

Contents of `myfile.json`:

```
[
```

```

    {"Data": "Rm1yc3QgdGhpbmc="},
    {"Data": "U2Vjb25kIHRoaW5n"},
    {"Data": "VGhpcmQgdGhpbmc="}
  ]

```

Output:

```

{
  "FailedPutCount": 0,
  "Encrypted": false,
  "RequestResponses": [
    {
      "RecordId": "9D20J6t2EqCTZTXwGzeSv/EVHxRoRCw89xd+o3+sXg8DhY0aWKPSmZy/
CGlRVEys1u1xbeKh6VofEYKkoeiDrcjrxhQp9iF7sUW7pujiMEQ5LzlrzCkGosxQn
+3boDnURDEaD42V7Giixp0yLJkYZcae1i7HzlCEoy9LJhM1r8EjDSi40m/9Vc2uhwwuAtGE0XKpxJ2WD7ZRwtAnY1KAnv
    },
    {
      "RecordId": "jFirejqxCLlK5xjH/UNm1MVcjkTEN76I7916X9PaZ
+PVa0SXDFu1WG0qEZhxq2js7xcZ552eoeDxsuTU1MSq9nZTbVfb6cQTIXnm/GsuF37Uhg67GkmR5z9016XKJ
+/+pDl0Fv7Hh9a3oUS6wYm3DcNRLTHHAimANp1PhkQvWpvLRfzbuCUkBphR2QVzhP90iHLbzGwy8/
DfH8sqWEUYASNJKS8GXP5s"
    },
    {
      "RecordId":
"oy0amQ40o5Y2YV4vxzufdcM00w6n3EP13tpPJGoYVnKH4APPVqNcbUgefo1stEFRg4hTLrf2k6eliHu/9+YJ5R3iie
DTBt3qBlmTj7Xq8SKVb01S7YvMTpWkMKA86f8JfmT8BMKoMb4XZS/s0kQLe+qh0sYKXWl"
    }
  ]
}

```

For more information, see [Sending Data to an Amazon Kinesis Data Firehose Delivery Stream](#) in the *Amazon Kinesis Data Firehose Developer Guide*.

- For API details, see [PutRecordBatch](#) in *AWS CLI Command Reference*.

put-record

The following code example shows how to use `put-record`.

AWS CLI

To write a record to a stream

The following `put-record` example writes data to a stream. The data is encoded in Base64 format.

```
aws firehose put-record \  
  --delivery-stream-name my-stream \  
  --record '{"Data":"SGVsbG8gd29ybGQ="}'
```

Output:

```
{  
  "RecordId": "RjB5K/nnoGFHqwTsZ1Nd/  
TTqvjE8V5dsyXZTQn2JXrdpMT0wssyEb6nfC8fwf1whhwnItt4mvrn+gsqeK5jB7QjuLg283+Ps4Sz/  
j1Xujv31iDhnPdaLw4B0yM9Amv7PcCuB2079RuM0NhoakbyUymlwY8yt20G8X2420wu1j1Fafhci4erAt7QhDEvpwuK8  
  "Encrypted": false  
}
```

For more information, see [Sending Data to an Amazon Kinesis Data Firehose Delivery Stream](#) in the *Amazon Kinesis Data Firehose Developer Guide*.

- For API details, see [PutRecord](#) in *AWS CLI Command Reference*.

Amazon Data Lifecycle Manager examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon Data Lifecycle Manager.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-default-role

The following code example shows how to use `create-default-role`.

AWS CLI

To create the required IAM role for Amazon DLM

The following `dlm create-default-role` example creates the `AWSDataLifecycleManagerDefaultRole` default role for managing snapshots.

```
aws dlm create-default-role \  
  --resource-type snapshot
```

This command produces no output.

For more information, see [Default service roles for Amazon Data Lifecycle Manager](#) in the *Amazon Elastic Compute Cloud User Guide*.

- For API details, see [CreateDefaultRole](#) in *AWS CLI Command Reference*.

create-lifecycle-policy

The following code example shows how to use `create-lifecycle-policy`.

AWS CLI

To create a lifecycle policy

The following `create-lifecycle-policy` example creates a lifecycle policy that creates a daily snapshot of volumes at the specified time. The specified tags are added to the snapshots, and tags are also copied from the volume and added to the snapshots. If creating a new snapshot exceeds the specified maximum count, the oldest snapshot is deleted.

```
aws dlm create-lifecycle-policy \  
  --description "My first policy" \  
  --state ENABLED \  
  --execution-role-arn arn:aws:iam::12345678910:role/  
AWSDataLifecycleManagerDefaultRole \  
  --policy-details file://policyDetails.json
```


Contents of policyDetails.json:

```
{
  "ResourceTypes": [
    "VOLUME"
  ],
  "TargetTags": [
    {
      "Key": "costCenter",
      "Value": "115"
    }
  ],
  "Schedules": [
    {
      "Name": "DailySnapshots",
      "CopyTags": true,
      "TagsToAdd": [
        {
          "Key": "type",
          "Value": "myDailySnapshot"
        }
      ],
      "CreateRule": {
        "Interval": 24,
        "IntervalUnit": "HOURS",
        "Times": [
          "03:00"
        ]
      },
      "RetainRule": {
        "Count": 5
      }
    }
  ]
}
```

Output:

```
{
  "PolicyId": "policy-0123456789abcdef0"
}
```

- For API details, see [CreateLifecyclePolicy](#) in *AWS CLI Command Reference*.

delete-lifecycle-policy

The following code example shows how to use delete-lifecycle-policy.

AWS CLI

To delete a lifecycle policy

The following example deletes the specified lifecycle policy.:

```
aws dlm delete-lifecycle-policy --policy-id policy-0123456789abcdef0
```

- For API details, see [DeleteLifecyclePolicy](#) in *AWS CLI Command Reference*.

get-lifecycle-policies

The following code example shows how to use get-lifecycle-policies.

AWS CLI

To get a summary of your lifecycle policies

The following get-lifecycle-policies example lists all of your lifecycle policies.

```
aws dlm get-lifecycle-policies
```

Output:

```
{
  "Policies": [
    {
      "PolicyId": "policy-0123456789abcdef0",
      "Description": "My first policy",
      "State": "ENABLED"
    }
  ]
}
```

- For API details, see [GetLifecyclePolicies](#) in *AWS CLI Command Reference*.

get-lifecycle-policy

The following code example shows how to use `get-lifecycle-policy`.

AWS CLI

To describe a lifecycle policy

The following `get-lifecycle-policy` example displays details for the specified lifecycle policy.

```
aws dlm get-lifecycle-policy \  
  --policy-id policy-0123456789abcdef0
```

Output:

```
{  
  "Policy": {  
    "PolicyId": "policy-0123456789abcdef0",  
    "Description": "My policy",  
    "State": "ENABLED",  
    "ExecutionRoleArn": "arn:aws:iam::123456789012:role/  
AWSDataLifecycleManagerDefaultRole",  
    "DateCreated": "2019-08-08T17:45:42Z",  
    "DateModified": "2019-08-08T17:45:42Z",  
    "PolicyDetails": {  
      "PolicyType": "EBS_SNAPSHOT_MANAGEMENT",  
      "ResourceTypes": [  
        "VOLUME"  
      ],  
      "TargetTags": [  
        {  
          "Key": "costCenter",  
          "Value": "115"  
        }  
      ],  
      "Schedules": [  
        {  
          "Name": "DailySnapshots",  
          "CopyTags": true,  
          "TagsToAdd": [  
            {  
              "Key": "type",
```

```

        "Value": "myDailySnapshot"
      }
    ],
    "CreateRule": {
      "Interval": 24,
      "IntervalUnit": "HOURS",
      "Times": [
        "03:00"
      ]
    },
    "RetainRule": {
      "Count": 5
    }
  }
]
}

```

- For API details, see [GetLifecyclePolicy](#) in *AWS CLI Command Reference*.

update-lifecycle-policy

The following code example shows how to use update-lifecycle-policy.

AWS CLI

Example 1: To enable a lifecycle policy

The following update-lifecycle-policy example enables the specified lifecycle policy.

```

aws dlm update-lifecycle-policy \
  --policy-id policy-0123456789abcdef0 \
  --state ENABLED

```

Example 2: To disable a lifecycle policy

The following update-lifecycle-policy example disables the specified lifecycle policy.

```

aws dlm update-lifecycle-policy \
  --policy-id policy-0123456789abcdef0 \
  --state DISABLED

```

Example 3: To update the details for lifecycle policy

The following `update-lifecycle-policy` example updates the target tags for the specified lifecycle policy.

```
aws dlm update-lifecycle-policy \  
  --policy-id policy-0123456789abcdef0 \  
  --policy-details file://policyDetails.json
```

Contents of `policyDetails.json`. Other details not referenced in this file are not changed by the command.

```
{  
  "TargetTags": [  
    {  
      "Key": "costCenter",  
      "Value": "120"  
    },  
    {  
      "Key": "project",  
      "Value": "lima"  
    }  
  ]  
}
```

- For API details, see [UpdateLifecyclePolicy](#) in *AWS CLI Command Reference*.

AWS Data Pipeline examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Data Pipeline.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

activate-pipeline

The following code example shows how to use `activate-pipeline`.

AWS CLI

To activate a pipeline

This example activates the specified pipeline:

```
aws datapipeline activate-pipeline --pipeline-id df-00627471S0VYZEXAMPLE
```

To activate the pipeline at a specific date and time, use the following command:

```
aws datapipeline activate-pipeline --pipeline-id df-00627471S0VYZEXAMPLE --start-timestamp 2015-04-07T00:00:00Z
```

- For API details, see [ActivatePipeline](#) in *AWS CLI Command Reference*.

add-tags

The following code example shows how to use `add-tags`.

AWS CLI

To add a tag to a pipeline

This example adds the specified tag to the specified pipeline:

```
aws datapipeline add-tags --pipeline-id df-00627471S0VYZEXAMPLE --tags key=environment,value=production key=owner,value=sales
```

To view the tags, use the `describe-pipelines` command. For example, the tags added in the example command appear as follows in the output for `describe-pipelines`:

```
{
  ...
  "tags": [
    {
      "value": "production",
      "key": "environment"
    },
    {
      "value": "sales",
      "key": "owner"
    }
  ]
  ...
}
```

- For API details, see [AddTags](#) in *AWS CLI Command Reference*.

create-pipeline

The following code example shows how to use create-pipeline.

AWS CLI

To create a pipeline

This example creates a pipeline:

```
aws datapipeline create-pipeline --name my-pipeline --unique-id my-pipeline-token
```

The following is example output:

```
{
  "pipelineId": "df-00627471S0VYZEXAMPLE"
}
```

- For API details, see [CreatePipeline](#) in *AWS CLI Command Reference*.

deactivate-pipeline

The following code example shows how to use deactivate-pipeline.

AWS CLI

To deactivate a pipeline

This example deactivates the specified pipeline:

```
aws datapipeline deactivate-pipeline --pipeline-id df-00627471S0VYZEXAMPLE
```

To deactivate the pipeline only after all running activities finish, use the following command:

```
aws datapipeline deactivate-pipeline --pipeline-id df-00627471S0VYZEXAMPLE --no-cancel-active
```

- For API details, see [DeactivatePipeline](#) in *AWS CLI Command Reference*.

delete-pipeline

The following code example shows how to use delete-pipeline.

AWS CLI

To delete a pipeline

This example deletes the specified pipeline:

```
aws datapipeline delete-pipeline --pipeline-id df-00627471S0VYZEXAMPLE
```

- For API details, see [DeletePipeline](#) in *AWS CLI Command Reference*.

describe-pipelines

The following code example shows how to use describe-pipelines.

AWS CLI

To describe your pipelines

This example describes the specified pipeline:

```
aws datapipeline describe-pipelines --pipeline-ids df-00627471S0VYZEXAMPLE
```


The following is example output:

```
{
  "pipelineDescriptionList": [
    {
      "fields": [
        {
          "stringValue": "PENDING",
          "key": "@pipelineState"
        },
        {
          "stringValue": "my-pipeline",
          "key": "name"
        },
        {
          "stringValue": "2015-04-07T16:05:58",
          "key": "@creationTime"
        },
        {
          "stringValue": "df-00627471S0VYZEXAMPLE",
          "key": "@id"
        },
        {
          "stringValue": "123456789012",
          "key": "pipelineCreator"
        },
        {
          "stringValue": "PIPELINE",
          "key": "@sphere"
        },
        {
          "stringValue": "123456789012",
          "key": "@userId"
        },
        {
          "stringValue": "123456789012",
          "key": "@accountId"
        },
        {
          "stringValue": "my-pipeline-token",
          "key": "uniqueId"
        }
      ],
      "pipelineId": "df-00627471S0VYZEXAMPLE",
    }
  ]
}
```

```

        "name": "my-pipeline",
        "tags": []
    }
]
}

```

- For API details, see [DescribePipelines](#) in *AWS CLI Command Reference*.

get-pipeline-definition

The following code example shows how to use `get-pipeline-definition`.

AWS CLI

To get a pipeline definition

This example gets the pipeline definition for the specified pipeline:

```
aws datapipeline get-pipeline-definition --pipeline-id df-00627471S0VYZEXAMPLE
```

The following is example output:

```

{
  "parameters": [
    {
      "type": "AWS::S3::ObjectKey",
      "id": "myS3OutputLoc",
      "description": "S3 output folder"
    },
    {
      "default": "s3://us-east-1.elasticmapreduce.samples/pig-apache-logs/data",
      "type": "AWS::S3::ObjectKey",
      "id": "myS3InputLoc",
      "description": "S3 input folder"
    },
    {
      "default": "grep -rc \"GET\" ${INPUT1_STAGING_DIR}/* >
${OUTPUT1_STAGING_DIR}/output.txt",
      "type": "String",
      "id": "myShellCmd",
      "description": "Shell command to run"
    }
  ],
}

```

```

"objects": [
  {
    "type": "Ec2Resource",
    "terminateAfter": "20 Minutes",
    "instanceType": "t1.micro",
    "id": "EC2ResourceObj",
    "name": "EC2ResourceObj"
  },
  {
    "name": "Default",
    "failureAndRerunMode": "CASCADE",
    "resourceRole": "DataPipelineDefaultResourceRole",
    "schedule": {
      "ref": "DefaultSchedule"
    },
    "role": "DataPipelineDefaultRole",
    "scheduleType": "cron",
    "id": "Default"
  },
  {
    "directoryPath": "#{myS3OutputLoc}/#{format(@scheduledStartTime, 'YYYY-MM-dd-HH-mm-ss')}",
    "type": "S3DataNode",
    "id": "S3OutputLocation",
    "name": "S3OutputLocation"
  },
  {
    "directoryPath": "#{myS3InputLoc}",
    "type": "S3DataNode",
    "id": "S3InputLocation",
    "name": "S3InputLocation"
  },
  {
    "startAt": "FIRST_ACTIVATION_DATE_TIME",
    "name": "Every 15 minutes",
    "period": "15 minutes",
    "occurrences": "4",
    "type": "Schedule",
    "id": "DefaultSchedule"
  },
  {
    "name": "ShellCommandActivityObj",
    "command": "#{myShellCmd}",
    "output": {

```

```

        "ref": "S3OutputLocation"
    },
    "input": {
        "ref": "S3InputLocation"
    },
    "stage": "true",
    "type": "ShellCommandActivity",
    "id": "ShellCommandActivityObj",
    "runsOn": {
        "ref": "EC2ResourceObj"
    }
}
],
"values": {
    "myS3OutputLoc": "s3://my-s3-bucket/",
    "myS3InputLoc": "s3://us-east-1.elasticmapreduce.samples/pig-apache-logs/
data",
    "myShellCmd": "grep -rc \"GET\" ${INPUT1_STAGING_DIR}/* >
${OUTPUT1_STAGING_DIR}/output.txt"
}
}

```

- For API details, see [GetPipelineDefinition](#) in *AWS CLI Command Reference*.

list-pipelines

The following code example shows how to use `list-pipelines`.

AWS CLI

To list your pipelines

This example lists your pipelines:

```
aws datapipeline list-pipelines
```

The following is example output:

```
{
  "pipelineIdList": [
    {
      "id": "df-00627471S0VYZEXAMPLE",

```

```

    "name": "my-pipeline"
  },
  {
    "id": "df-09028963KNVMREXAMPLE",
    "name": "ImportDDB"
  },
  {
    "id": "df-0870198233ZYVEXAMPLE",
    "name": "CrossRegionDDB"
  },
  {
    "id": "df-00189603TB4MZEXAMPLE",
    "name": "CopyRedshift"
  }
]
}

```

- For API details, see [ListPipelines](#) in *AWS CLI Command Reference*.

list-runs

The following code example shows how to use `list-runs`.

AWS CLI

Example 1: To list your pipeline runs

The following `list-runs` example lists the runs for the specified pipeline.

```
aws datapipeline list-runs --pipeline-id df-00627471S0VYZEXAMPLE
```

Output:

Name	Scheduled Start	Status	ID
	Started		Ended

1. EC2ResourceObj	2015-04-12T17:33:02	CREATING	
@EC2ResourceObj_2015-04-12T17:33:02			2015-04-12T17:33:10
2. S3InputLocation	2015-04-12T17:33:02	FINISHED	
@S3InputLocation_2015-04-12T17:33:02			2015-04-12T17:33:09
2015-04-12T17:33:09			

```

3. S3OutputLocation          2015-04-12T17:33:02    WAITING_ON_DEPENDENCIES
   @S3OutputLocation_2015-04-12T17:33:02    2015-04-12T17:33:09
4. ShellCommandActivityObj   2015-04-12T17:33:02    WAITING_FOR_RUNNER
   @ShellCommandActivityObj_2015-04-12T17:33:02    2015-04-12T17:33:09

```

Example 2: To list the pipeline runs between the specified dates

The following `list-runs` example uses the `--start-interval` to specify the dates to include in the output.

```
aws datapipeline list-runs --pipeline-id df-01434553B58A2SHZUK05 --start-interval
2017-10-07T00:00:00,2017-10-08T00:00:00
```

- For API details, see [ListRuns](#) in *AWS CLI Command Reference*.

put-pipeline-definition

The following code example shows how to use `put-pipeline-definition`.

AWS CLI

To upload a pipeline definition

This example uploads the specified pipeline definition to the specified pipeline:

```
aws datapipeline put-pipeline-definition --pipeline-id df-00627471S0VYZEXAMPLE --
pipeline-definition file://my-pipeline-definition.json
```

The following is example output:

```
{
  "validationErrors": [],
  "errored": false,
  "validationWarnings": []
}
```

- For API details, see [PutPipelineDefinition](#) in *AWS CLI Command Reference*.

remove-tags

The following code example shows how to use `remove-tags`.

AWS CLI

To remove a tag from a pipeline

This example removes the specified tag from the specified pipeline:

```
aws datapipeline remove-tags --pipeline-id df-00627471S0VYZEXAMPLE --tag-keys
environment
```

- For API details, see [RemoveTags](#) in *AWS CLI Command Reference*.

DataSync examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with DataSync.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

update-location-azure-blob

The following code example shows how to use `update-location-azure-blob`.

AWS CLI

To update your transfer location with a new agent

The following `update-location-object-storage` example updates your DataSync location for Microsoft Azure Blob Storage with a new agent.

```
aws datasync update-location-azure-blob \  
  --location-arn arn:aws:datasync:us-west-2:123456789012:location/loc-  
abcdef01234567890 \  
  --agent-arns arn:aws:datasync:us-west-2:123456789012:agent/  
agent-1234567890abcdef0 \  
  --sas-configuration '{ \  
    "Token": "sas-token-for-azure-blob-storage-access" \  
  }'
```

This command produces no output.

For more information, see [Replacing your agent](#) in the *AWS DataSync User Guide*.

- For API details, see [UpdateLocationAzureBlob](#) in *AWS CLI Command Reference*.

update-location-hdfs

The following code example shows how to use `update-location-hdfs`.

AWS CLI

To update your transfer location with a new agent

The following `update-location-hdfs` example updates your DataSync HDFS location with a new agent. You only need the `--kerberos-keytab` and `--kerberos-krb5-conf` options if your HDFS cluster uses Kerberos authentication.

```
aws datasync update-location-hdfs \  
  --location-arn arn:aws:datasync:us-west-2:123456789012:location/loc-  
abcdef01234567890 \  
  --agent-arns arn:aws:datasync:us-west-2:123456789012:agent/  
agent-1234567890abcdef0 \  
  --kerberos-keytab file://hdfs.keytab \  
  --kerberos-krb5-conf file://krb5.conf
```

Contents of `hdfs.keytab`:

N/A. The content of this file is encrypted and not human readable.

Contents of `krb5.conf`:

```
[libdefaults]
```



```
default_realm = EXAMPLE.COM
dns_lookup_realm = false
dns_lookup_kdc = false
rdns = true
ticket_lifetime = 24h
forwardable = true
udp_preference_limit = 1000000
default_tkt_etypes = aes256-cts-hmac-sha1-96 aes128-cts-hmac-sha1-96 des3-cbc-sha1
default_tgs_etypes = aes256-cts-hmac-sha1-96 aes128-cts-hmac-sha1-96 des3-cbc-sha1
permitted_etypes = aes256-cts-hmac-sha1-96 aes128-cts-hmac-sha1-96 des3-cbc-sha1

[realms]
EXAMPLE.COM = {
    kdc = kdc1.example.com
    admin_server = krbadmin.example.com
    default_domain = example.com
}

[domain_realm]
.example.com = EXAMPLE.COM
example.com = EXAMPLE.COM

[logging]
kdc = FILE:/var/log/krb5kdc.log
admin_server = FILE:/var/log/kerberos/kadmin.log
default = FILE:/var/log/krb5libs.log
```

This command produces no output.

For more information, see [Replacing your agent](#) in the *AWS DataSync User Guide*.

- For API details, see [UpdateLocationHdfs](#) in *AWS CLI Command Reference*.

update-location-nfs

The following code example shows how to use `update-location-nfs`.

AWS CLI

To update your transfer location with a new agent

The following `update-location-nfs` example updates your DataSync NFS location with a new agent.

```
aws datasync update-location-nfs \  
  --location-arn arn:aws:datasync:us-west-2:123456789012:location/loc-  
  abcdef01234567890 \  
  --on-prem-config AgentArns=arn:aws:datasync:us-west-2:123456789012:agent/  
  agent-1234567890abcdef0
```

This command produces no output.

For more information, see [Replacing your agent](#) in the *AWS DataSync User Guide*.

- For API details, see [UpdateLocationNfs](#) in *AWS CLI Command Reference*.

update-location-object-storage

The following code example shows how to use `update-location-object-storage`.

AWS CLI

To update your transfer location with a new agent

The following `update-location-object-storage` example updates your DataSync object storage location with a new agent.

```
aws datasync update-location-object-storage \  
  --location-arn arn:aws:datasync:us-west-2:123456789012:location/loc-  
  abcdef01234567890 \  
  --agent-arns arn:aws:datasync:us-west-2:123456789012:agent/  
  agent-1234567890abcdef0 \  
  --secret-key secret-key-for-object-storage
```

This command produces no output.

For more information, see [Replacing your agent](#) in the *AWS DataSync User Guide*.

- For API details, see [UpdateLocationObjectStorage](#) in *AWS CLI Command Reference*.

update-location-smb

The following code example shows how to use `update-location-smb`.

AWS CLI

To update your transfer location with a new agent

The following `update-location-smb` example updates your DataSync SMB location with a new agent.

```
aws datasync update-location-smb \  
  --location-arn arn:aws:datasync:us-west-2:123456789012:location/loc-  
abcdef01234567890 \  
  --agent-arns arn:aws:datasync:us-west-2:123456789012:agent/  
agent-1234567890abcdef0 \  
  --password smb-file-server-password
```

This command produces no output.

For more information, see [Replacing your agent](#) in the *AWS DataSync User Guide*.

- For API details, see [UpdateLocationSmb](#) in *AWS CLI Command Reference*.

DAX examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with DAX.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-cluster

The following code example shows how to use `create-cluster`.

AWS CLI

To create a DAX cluster

The following `create-cluster` example creates a DAX cluster with the specified settings.

```
aws dax create-cluster \  
  --cluster-name daxcluster \  
  --node-type dax.r4.large \  
  --replication-factor 3 \  
  --iam-role-arn roleARN \  
  --sse-specification Enabled=true
```

Output:

```
{  
  "Cluster": {  
    "ClusterName": "daxcluster",  
    "ClusterArn": "arn:aws:dax:us-west-2:123456789012:cache/daxcluster",  
    "TotalNodes": 3,  
    "ActiveNodes": 0,  
    "NodeType": "dax.r4.large",  
    "Status": "creating",  
    "ClusterDiscoveryEndpoint": {  
      "Port": 8111  
    },  
    "PreferredMaintenanceWindow": "thu:13:00-thu:14:00",  
    "SubnetGroup": "default",  
    "SecurityGroups": [  
      {  
        "SecurityGroupIdentifier": "sg-1af6e36e",  
        "Status": "active"  
      }  
    ],  
    "IamRoleArn": "arn:aws:iam::123456789012:role/  
DAXServiceRoleForDynamoDBAccess",  
    "ParameterGroup": {
```

```

        "ParameterGroupName": "default.dax1.0",
        "ParameterApplyStatus": "in-sync",
        "NodeIdsToReboot": []
    },
    "SSEDescription": {
        "Status": "ENABLED"
    }
}
}

```

For more information, see [Step 3: Create a DAX Cluster](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [CreateCluster](#) in *AWS CLI Command Reference*.

create-parameter-group

The following code example shows how to use `create-parameter-group`.

AWS CLI

To create a parameter group

The following `create-parameter-group` example creates a parameter group with the specified settings.

```

aws dax create-parameter-group \
  --parameter-group-name daxparametergroup \
  --description "A new parameter group"

```

Output:

```

{
  "ParameterGroup": {
    "ParameterGroupName": "daxparametergroup",
    "Description": "A new parameter group"
  }
}

```

For more information, see [Managing DAX Clusters](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [CreateParameterGroup](#) in *AWS CLI Command Reference*.

create-subnet-group

The following code example shows how to use `create-subnet-group`.

AWS CLI

To create a DAX subnet group

The following `create-subnet-group` example creates a subnet group with the specified settings.

```
aws dax create-subnet-group \  
  --subnet-group-name daxSubnetGroup \  
  --subnet-ids subnet-11111111 subnet-22222222
```

Output:

```
{  
  "SubnetGroup": {  
    "SubnetGroupName": "daxSubnetGroup",  
    "VpcId": "vpc-05a1fa8e00c325226",  
    "Subnets": [  
      {  
        "SubnetIdentifier": "subnet-11111111",  
        "SubnetAvailabilityZone": "us-west-2b"  
      },  
      {  
        "SubnetIdentifier": "subnet-22222222",  
        "SubnetAvailabilityZone": "us-west-2c"  
      }  
    ]  
  }  
}
```

For more information, see [Step 2: Create a Subnet Group](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [CreateSubnetGroup](#) in *AWS CLI Command Reference*.

decrease-replication-factor

The following code example shows how to use `decrease-replication-factor`.

AWS CLI

To remove one or more nodes from the cluster

The following `decrease-replication-factor` example decreases the number of nodes in the specified DAX cluster to one.

```
aws dax decrease-replication-factor \  
  --cluster-name daxcluster \  
  --new-replication-factor 1
```

Output:

```
{  
  "Cluster": {  
    "ClusterName": "daxcluster",  
    "ClusterArn": "arn:aws:dax:us-west-2:123456789012:cache/daxcluster",  
    "TotalNodes": 3,  
    "ActiveNodes": 3,  
    "NodeType": "dax.r4.large",  
    "Status": "modifying",  
    "ClusterDiscoveryEndpoint": {  
      "Address": "daxcluster.ey3o9d.clustercfg.dax.usw2.cache.amazonaws.com",  
      "Port": 8111  
    },  
    "Nodes": [  
      {  
        "NodeId": "daxcluster-a",  
        "Endpoint": {  
          "Address": "daxcluster-  
a.ey3o9d.0001.dax.usw2.cache.amazonaws.com",  
          "Port": 8111  
        },  
        "NodeCreateTime": 1576625059.509,  
        "AvailabilityZone": "us-west-2c",  
        "NodeStatus": "available",  
        "ParameterGroupStatus": "in-sync"  
      },  
      {  
        "NodeId": "daxcluster-b",  
        "Endpoint": {  
          "Address": "daxcluster-  
b.ey3o9d.0001.dax.usw2.cache.amazonaws.com",
```

```

        "Port": 8111
      },
      "NodeCreateTime": 1576625059.509,
      "AvailabilityZone": "us-west-2a",
      "NodeStatus": "available",
      "ParameterGroupStatus": "in-sync"
    },
    {
      "NodeId": "daxcluster-c",
      "Endpoint": {
        "Address": "daxcluster-
c.ey3o9d.0001.dax.usw2.cache.amazonaws.com",
        "Port": 8111
      },
      "NodeCreateTime": 1576625059.509,
      "AvailabilityZone": "us-west-2b",
      "NodeStatus": "available",
      "ParameterGroupStatus": "in-sync"
    }
  ],
  "PreferredMaintenanceWindow": "thu:13:00-thu:14:00",
  "SubnetGroup": "default",
  "SecurityGroups": [
    {
      "SecurityGroupIdentifier": "sg-1af6e36e",
      "Status": "active"
    }
  ],
  "IamRoleArn": "arn:aws:iam::123456789012:role/
DAXServiceRoleForDynamoDBAccess",
  "ParameterGroup": {
    "ParameterGroupName": "default.dax1.0",
    "ParameterApplyStatus": "in-sync",
    "NodeIdsToReboot": []
  },
  "SSEDescription": {
    "Status": "ENABLED"
  }
}
}

```

For more information, see [Managing DAX Clusters](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DecreaseReplicationFactor](#) in *AWS CLI Command Reference*.

delete-cluster

The following code example shows how to use `delete-cluster`.

AWS CLI

To delete a DAX cluster

The following `delete-cluster` example deletes the specified DAX cluster.

```
aws dax delete-cluster \  
  --cluster-name daxcluster
```

Output:

```
{  
  "Cluster": {  
    "ClusterName": "daxcluster",  
    "ClusterArn": "arn:aws:dax:us-west-2:123456789012:cache/daxcluster",  
    "TotalNodes": 3,  
    "ActiveNodes": 0,  
    "NodeType": "dax.r4.large",  
    "Status": "deleting",  
    "ClusterDiscoveryEndpoint": {  
      "Address": "dd.eyJ3o9d.clustercfg.dax.usw2.cache.amazonaws.com",  
      "Port": 8111  
    },  
    "PreferredMaintenanceWindow": "fri:06:00-fri:07:00",  
    "SubnetGroup": "default",  
    "SecurityGroups": [  
      {  
        "SecurityGroupIdentifier": "sg-1af6e36e",  
        "Status": "active"  
      }  
    ],  
    "IamRoleArn": "arn:aws:iam::123456789012:role/  
DAXServiceRoleForDynamoDBAccess",  
    "ParameterGroup": {  
      "ParameterGroupName": "default.dax1.0",  
      "ParameterApplyStatus": "in-sync",  
      "NodeIdsToReboot": []  
    },  
    "SSEDescription": {
```

```
        "Status": "ENABLED"
      }
    }
  }
```

For more information, see [Managing DAX Clusters](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DeleteCluster](#) in *AWS CLI Command Reference*.

delete-parameter-group

The following code example shows how to use `delete-parameter-group`.

AWS CLI

To delete a parameter group

The following `delete-parameter-group` example deletes the specified DAX parameter group.

```
aws dax delete-parameter-group \
  --parameter-group-name daxparametergroup
```

Output:

```
{
  "DeletionMessage": "Parameter group daxparametergroup has been deleted."
}
```

For more information, see [Managing DAX Clusters](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DeleteParameterGroup](#) in *AWS CLI Command Reference*.

delete-subnet-group

The following code example shows how to use `delete-subnet-group`.

AWS CLI

To delete a subnet group

The following `delete-subnet-group` example deletes the specified DAX subnet group.

```
aws dax delete-subnet-group \  
  --subnet-group-name daxSubnetGroup
```

Output:

```
{  
  "DeletionMessage": "Subnet group daxSubnetGroup has been deleted."  
}
```

For more information, see [Managing DAX Clusters](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DeleteSubnetGroup](#) in *AWS CLI Command Reference*.

describe-clusters

The following code example shows how to use `describe-clusters`.

AWS CLI**To return information about all provisioned DAX clusters**

The following `describe-clusters` example displays details about all provisioned DAX clusters.

```
aws dax describe-clusters
```

Output:

```
{  
  "Clusters": [  
    {  
      "ClusterName": "daxcluster",  
      "ClusterArn": "arn:aws:dax:us-west-2:123456789012:cache/daxcluster",  
      "TotalNodes": 1,  
      "ActiveNodes": 1,  
      "NodeType": "dax.r4.large",  
      "Status": "available",  
      "ClusterDiscoveryEndpoint": {  
        "Address":  
"daxcluster.ey3o9d.clustercfg.dax.usw2.cache.amazonaws.com",  
        "Port": 8111  
      }  
    },  
  ],  
}
```

```

    "Nodes": [
      {
        "NodeId": "daxcluster-a",
        "Endpoint": {
          "Address": "daxcluster-
a.ey3o9d.0001.dax.usw2.cache.amazonaws.com",
          "Port": 8111
        },
        "NodeCreateTime": 1576625059.509,
        "AvailabilityZone": "us-west-2c",
        "NodeStatus": "available",
        "ParameterGroupStatus": "in-sync"
      }
    ],
    "PreferredMaintenanceWindow": "thu:13:00-thu:14:00",
    "SubnetGroup": "default",
    "SecurityGroups": [
      {
        "SecurityGroupIdentifier": "sg-1af6e36e",
        "Status": "active"
      }
    ],
    "IamRoleArn": "arn:aws:iam::123456789012:role/
DAXServiceRoleForDynamoDBAccess",
    "ParameterGroup": {
      "ParameterGroupName": "default.dax1.0",
      "ParameterApplyStatus": "in-sync",
      "NodeIdsToReboot": []
    },
    "SSEDescription": {
      "Status": "ENABLED"
    }
  }
]
}

```

For more information, see [Managing DAX Clusters](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DescribeClusters](#) in *AWS CLI Command Reference*.

describe-default-parameters

The following code example shows how to use describe-default-parameters.

AWS CLI

To return the default system parameter information for DAX

The following `describe-default-parameters` example displays the default system parameter information for DAX.

```
aws dax describe-default-parameters
```

Output:

```
{
  "Parameters": [
    {
      "ParameterName": "query-ttl-millis",
      "ParameterType": "DEFAULT",
      "ParameterValue": "300000",
      "NodeTypeSpecificValues": [],
      "Description": "Duration in milliseconds for queries to remain cached",
      "Source": "user",
      "DataType": "integer",
      "AllowedValues": "0-",
      "IsModifiable": "TRUE",
      "ChangeType": "IMMEDIATE"
    },
    {
      "ParameterName": "record-ttl-millis",
      "ParameterType": "DEFAULT",
      "ParameterValue": "300000",
      "NodeTypeSpecificValues": [],
      "Description": "Duration in milliseconds for records to remain valid in
cache (Default: 0 = infinite)",
      "Source": "user",
      "DataType": "integer",
      "AllowedValues": "0-",
      "IsModifiable": "TRUE",
      "ChangeType": "IMMEDIATE"
    }
  ]
}
```

For more information, see [Managing DAX Clusters](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DescribeDefaultParameters](#) in *AWS CLI Command Reference*.

describe-events

The following code example shows how to use describe-events.

AWS CLI

To return all events related to DAX clusters and parameter groups

The following describe-events example displays details of events that are related to DAX clusters and parameter groups.

```
aws dax describe-events
```

Output:

```
{
  "Events": [
    {
      "SourceName": "daxcluster",
      "SourceType": "CLUSTER",
      "Message": "Cluster deleted.",
      "Date": 1576702736.706
    },
    {
      "SourceName": "daxcluster",
      "SourceType": "CLUSTER",
      "Message": "Removed node daxcluster-b.",
      "Date": 1576702691.738
    },
    {
      "SourceName": "daxcluster",
      "SourceType": "CLUSTER",
      "Message": "Removed node daxcluster-a.",
      "Date": 1576702633.498
    },
    {
      "SourceName": "daxcluster",
      "SourceType": "CLUSTER",
      "Message": "Removed node daxcluster-c.",
      "Date": 1576702631.329
    }
  ]
}
```

```
    },
    {
      "SourceName": "daxcluster",
      "SourceType": "CLUSTER",
      "Message": "Cluster created.",
      "Date": 1576626560.057
    }
  ]
}
```

For more information, see [Managing DAX Clusters](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DescribeEvents](#) in *AWS CLI Command Reference*.

describe-parameter-groups

The following code example shows how to use `describe-parameter-groups`.

AWS CLI

To describe the parameter groups defined in DAX

The following `describe-parameter-groups` example retrieves details about the parameter groups that are defined in DAX.

```
aws dax describe-parameter-groups
```

Output:

```
{
  "ParameterGroups": [
    {
      "ParameterGroupName": "default.dax1.0",
      "Description": "Default parameter group for dax1.0"
    }
  ]
}
```

For more information, see [Managing DAX Clusters](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DescribeParameterGroups](#) in *AWS CLI Command Reference*.

describe-parameters

The following code example shows how to use describe-parameters.

AWS CLI

To describe the parameters defined in a DAX parameter group

The following describe-parameters example retrieves details about the parameters that are defined in the specified DAX parameter group.

```
aws dax describe-parameters \  
  --parameter-group-name default.dax1.0
```

Output:

```
{  
  "Parameters": [  
    {  
      "ParameterName": "query-ttl-millis",  
      "ParameterType": "DEFAULT",  
      "ParameterValue": "300000",  
      "NodeTypeSpecificValues": [],  
      "Description": "Duration in milliseconds for queries to remain cached",  
      "Source": "user",  
      "DataType": "integer",  
      "AllowedValues": "0-",  
      "IsModifiable": "TRUE",  
      "ChangeType": "IMMEDIATE"  
    },  
    {  
      "ParameterName": "record-ttl-millis",  
      "ParameterType": "DEFAULT",  
      "ParameterValue": "300000",  
      "NodeTypeSpecificValues": [],  
      "Description": "Duration in milliseconds for records to remain valid in  
cache (Default: 0 = infinite)",  
      "Source": "user",  
      "DataType": "integer",  
      "AllowedValues": "0-",  
      "IsModifiable": "TRUE",  
      "ChangeType": "IMMEDIATE"  
    }  
  ]  
}
```



```
]
}
```

For more information, see [Managing DAX Clusters](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DescribeParameters](#) in *AWS CLI Command Reference*.

describe-subnet-groups

The following code example shows how to use describe-subnet-groups.

AWS CLI

To describe subnet groups defined in DAX

The following describe-subnet-groups example retrieves details for the subnet groups defined in DAX.

```
aws dax describe-subnet-groups
```

Output:

```
{
  "SubnetGroups": [
    {
      "SubnetGroupName": "default",
      "Description": "Default CacheSubnetGroup",
      "VpcId": "vpc-ee70a196",
      "Subnets": [
        {
          "SubnetIdentifier": "subnet-874953af",
          "SubnetAvailabilityZone": "us-west-2d"
        },
        {
          "SubnetIdentifier": "subnet-bd3d1fc4",
          "SubnetAvailabilityZone": "us-west-2a"
        },
        {
          "SubnetIdentifier": "subnet-72c2ff28",
          "SubnetAvailabilityZone": "us-west-2c"
        },
        {
          "SubnetIdentifier": "subnet-09e6aa42",
```

```

        "SubnetAvailabilityZone": "us-west-2b"
      }
    ]
  }
}

```

For more information, see [Managing DAX Clusters](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DescribeSubnetGroups](#) in *AWS CLI Command Reference*.

increase-replication-factor

The following code example shows how to use `increase-replication-factor`.

AWS CLI

To increase the replication factor for a DAX cluster

The following `increase-replication-factor` example increases the specified DAX cluster's replication factor to 3.

```

aws dax increase-replication-factor \
  --cluster-name daxcluster \
  --new-replication-factor 3

```

Output:

```

{
  "Cluster": {
    "ClusterName": "daxcluster",
    "ClusterArn": "arn:aws:dax:us-west-2:123456789012:cache/daxcluster",
    "TotalNodes": 3,
    "ActiveNodes": 1,
    "NodeType": "dax.r4.large",
    "Status": "modifying",
    "ClusterDiscoveryEndpoint": {
      "Address": "daxcluster.ey3o9d.clustercfg.dax.usw2.cache.amazonaws.com",
      "Port": 8111
    },
    "Nodes": [
      {

```

```

        "NodeId": "daxcluster-a",
        "Endpoint": {
            "Address": "daxcluster-
a.ey3o9d.0001.dax.usw2.cache.amazonaws.com",
            "Port": 8111
        },
        "NodeCreateTime": 1576625059.509,
        "AvailabilityZone": "us-west-2c",
        "NodeStatus": "available",
        "ParameterGroupStatus": "in-sync"
    },
    {
        "NodeId": "daxcluster-b",
        "NodeStatus": "creating"
    },
    {
        "NodeId": "daxcluster-c",
        "NodeStatus": "creating"
    }
],
"PreferredMaintenanceWindow": "thu:13:00-thu:14:00",
"SubnetGroup": "default",
"SecurityGroups": [
    {
        "SecurityGroupIdentifier": "sg-1af6e36e",
        "Status": "active"
    }
],
"IamRoleArn": "arn:aws:iam::123456789012:role/
DAXServiceRoleForDynamoDBAccess",
"ParameterGroup": {
    "ParameterGroupName": "default.dax1.0",
    "ParameterApplyStatus": "in-sync",
    "NodeIdsToReboot": []
},
"SSEDescription": {
    "Status": "ENABLED"
}
}
}

```

For more information, see [Managing DAX Clusters](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [IncreaseReplicationFactor](#) in *AWS CLI Command Reference*.

list-tags

The following code example shows how to use `list-tags`.

AWS CLI

To list tags on a DAX resource

The following `list-tags` example lists the tag keys and values attached to the specified DAX cluster.

```
aws dax list-tags \  
  --resource-name arn:aws:dax:us-west-2:123456789012:cache/daxcluster
```

Output:

```
{  
  "Tags": [  
    {  
      "Key": "ClusterUsage",  
      "Value": "prod"  
    }  
  ]  
}
```

For more information, see [Managing DAX Clusters](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [ListTags](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To tag a DAX resource

The following `tag-resource` example attaches the specified tag key name and associated value to the specified DAX cluster to describe the cluster usage.

```
aws dax tag-resource \  
  --resource-name arn:aws:dax:us-west-2:123456789012:cache/daxcluster
```

```
--resource-name arn:aws:dax:us-west-2:123456789012:cache/daxcluster \  
--tags="Key=ClusterUsage,Value=prod"
```

Output:

```
{  
  "Tags": [  
    {  
      "Key": "ClusterUsage",  
      "Value": "prod"  
    }  
  ]  
}
```

For more information, see [Managing DAX Clusters](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from a DAX resource

The following `untag-resource` example removes the tag with the specified key name from a DAX cluster.

```
aws dax untag-resource \  
  --resource-name arn:aws:dax:us-west-2:123456789012:cache/daxcluster \  
  --tag-keys="ClusterUsage"
```

Output:

```
{  
  "Tags": []  
}
```

For more information, see [Managing DAX Clusters](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

Detective examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Detective.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

accept-invitation

The following code example shows how to use `accept-invitation`.

AWS CLI

To accept an invitation to become a member account in a behavior graph

The following `accept-invitation` example accepts an invitation to become a member account in behavior graph `arn:aws:detective:us-east-1:111122223333:graph:123412341234`.

```
aws detective accept-invitation \  
  --graph-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234
```

This command produces no output.

For more information, see [Responding to a behavior graph invitation](#) in the *Amazon Detective Administration Guide*.

- For API details, see [AcceptInvitation](#) in *AWS CLI Command Reference*.

create-graph

The following code example shows how to use `create-graph`.

AWS CLI

To enable Amazon Detective and create a new behavior graph

The following `create-graph` example enables Detective for the AWS account that runs the command in the Region where the command is run. A new behavior graph is created that has that account as its administrator account. The command also assigns the value `Finance` to the `Department` tag.

```
aws detective create-graph \  
  --tags '{"Department": "Finance"}'
```

Output:

```
{  
  "GraphArn": "arn:aws:detective:us-  
east-1:111122223333:graph:027c7c4610ea4aacaf0b883093cab899"  
}
```

For more information, see [Enabling Amazon Detective](#) in the *Amazon Detective Administration Guide*.

- For API details, see [CreateGraph](#) in *AWS CLI Command Reference*.

create-members

The following code example shows how to use `create-members`.

AWS CLI

To invite member accounts to a behavior graph

The following `create-members` example invites two AWS accounts to become member accounts in the behavior graph `arn:aws:detective:us-east-1:111122223333:graph:123412341234`. For each account, the request provides the AWS

account ID and the account root user email address. The request includes a custom message to insert into the invitation email.

```
aws detective create-members \  
  --accounts AccountId=444455556666,EmailAddress=mmajor@example.com  
  AccountId=123456789012,EmailAddress=jstiles@example.com \  
  --graph-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234 \  
  --message "This is Paul Santos. I need to add your account to the data we use  
for security investigation in Amazon Detective. If you have any questions, contact  
me at psantos@example.com."
```

Output:

```
{  
  "Members": [  
    {  
      "AccountId": "444455556666",  
      "AdministratorId": "111122223333",  
      "EmailAddress": "mmajor@example.com",  
      "GraphArn": "arn:aws:detective:us-east-1:111122223333:graph:123412341234",  
      "InvitedTime": 1579826107000,  
      "MasterId": "111122223333",  
      "Status": "INVITED",  
      "UpdatedTime": 1579826107000  
    },  
    {  
      "AccountId": "123456789012",  
      "AdministratorId": "111122223333",  
      "EmailAddress": "jstiles@example.com",  
      "GraphArn": "arn:aws:detective:us-east-1:111122223333:graph:123412341234",  
      "InvitedTime": 1579826107000,  
      "MasterId": "111122223333",  
      "Status": "VERIFICATION_IN_PROGRESS",  
      "UpdatedTime": 1579826107000  
    }  
  ],  
  "UnprocessedAccounts": [ ]  
}
```

For more information, see [Inviting member accounts to a behavior graph](https://docs.aws.amazon.com/detective/latest/adminguide/graph-admin-add-member-accounts.html) in the *Amazon Detective Administration Guide*.

To invite member accounts without sending invitation emails

The following `create-members` example invites two AWS accounts to become member accounts in the behavior graph `arn:aws:detective:us-east-1:111122223333:graph:123412341234`. For each account, the request provides the AWS account ID and the account root user email address. The member accounts do not receive invitation emails.

```
aws detective create-members \  
  --accounts AccountId=444455556666,EmailAddress=mmajor@example.com \  
  AccountId=123456789012,EmailAddress=jstiles@example.com \  
  --graph-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234 \  
  --disable-email-notification
```

Output:

```
{  
  "Members": [  
    {  
      "AccountId": "444455556666",  
      "AdministratorId": "111122223333",  
      "EmailAddress": "mmajor@example.com",  
      "GraphArn": "arn:aws:detective:us-east-1:111122223333:graph:123412341234",  
      "InvitedTime": 1579826107000,  
      "MasterId": "111122223333",  
      "Status": "INVITED",  
      "UpdatedTime": 1579826107000  
    },  
    {  
      "AccountId": "123456789012",  
      "AdministratorId": "111122223333",  
      "EmailAddress": "jstiles@example.com",  
      "GraphArn": "arn:aws:detective:us-east-1:111122223333:graph:123412341234",  
      "InvitedTime": 1579826107000,  
      "MasterId": "111122223333",  
      "Status": "VERIFICATION_IN_PROGRESS",  
      "UpdatedTime": 1579826107000  
    }  
  ],  
  "UnprocessedAccounts": [ ]  
}
```

For more information, see [Inviting member accounts to a behavior graph](https://docs.aws.amazon.com/detective/latest/adminguide/graph-admin-add-member-accounts.html) in the *Amazon Detective Administration Guide*.

- For API details, see [CreateMembers](#) in *AWS CLI Command Reference*.

delete-graph

The following code example shows how to use `delete-graph`.

AWS CLI

To disable Detective and delete the behavior graph

The following `delete-graph` example disables Detective and deletes the specified behavior graph.

```
aws detective delete-graph \  
  --graph-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234
```

This command produces no output.

For more information, see [Disabling Amazon Detective](#) in the *Amazon Detective Administration Guide*.

- For API details, see [DeleteGraph](#) in *AWS CLI Command Reference*.

delete-members

The following code example shows how to use `delete-members`.

AWS CLI

To remove member accounts from a behavior graph

The following `delete-members` example removes two member accounts from the behavior graph `arn:aws:detective:us-east-1:111122223333:graph:123412341234`. To identify the accounts, the request provides the AWS account IDs.

```
aws detective delete-members \  
  --account-ids 444455556666 123456789012 \  
  --graph-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234
```

Output:

```
{
  "AccountIds": [ "444455556666", "123456789012" ],
  "UnprocessedAccounts": [ ]
}
```

For more information, see Removing member accounts from a behavior graph<<https://docs.aws.amazon.com/detective/latest/adminguide/graph-admin-remove-member-accounts.html>> in the *Amazon Detective Administration Guide*.

- For API details, see [DeleteMembers](#) in *AWS CLI Command Reference*.

disassociate-membership

The following code example shows how to use `disassociate-membership`.

AWS CLI**To resign membership from a behavior graph**

The following `disassociate-membership` example removes the AWS account that runs the command from the behavior graph `arn:aws:detective:us-east-1:111122223333:graph:123412341234`.

```
aws detective disassociate-membership \
  --graph-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234
```

For more information, see Removing your account from a behavior graph<<https://docs.aws.amazon.com/detective/latest/adminguide/member-remove-self-from-graph.html>> in the *Amazon Detective Administration Guide*.

- For API details, see [DisassociateMembership](#) in *AWS CLI Command Reference*.

get-members

The following code example shows how to use `get-members`.

AWS CLI**To retrieve information about selected behavior graph member accounts**

The following `get-members` example retrieves information about two member accounts in the behavior graph `arn:aws:detective:us-east-1:111122223333:graph:123412341234`. For the two accounts, the request provides the AWS account IDs.

```
aws detective get-members \  
  --account-ids 444455556666 123456789012 \  
  --graph-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234
```

Output:

```
{  
  "MemberDetails": [  
    {  
      "AccountId": "444455556666",  
      "AdministratorId": "111122223333",  
      "EmailAddress": "mmajor@example.com",  
      "GraphArn": "arn:aws:detective:us-east-1:111122223333:graph:123412341234",  
      "InvitedTime": 1579826107000,  
      "MasterId": "111122223333",  
      "Status": "INVITED",  
      "UpdatedTime": 1579826107000  
    }  
    {  
      "AccountId": "123456789012",  
      "AdministratorId": "111122223333",  
      "EmailAddress": "jstiles@example.com",  
      "GraphArn": "arn:aws:detective:us-east-1:111122223333:graph:123412341234",  
      "InvitedTime": 1579826107000,  
      "MasterId": "111122223333",  
      "Status": "INVITED",  
      "UpdatedTime": 1579826107000  
    }  
  ],  
  "UnprocessedAccounts": [ ]  
}
```

For more information, see [Viewing the list of accounts in a behavior graph](https://docs.aws.amazon.com/detective/latest/adminguide/graph-admin-view-accounts.html) in the *Amazon Detective Administration Guide*.

- For API details, see [GetMembers](#) in *AWS CLI Command Reference*.

list-graphs

The following code example shows how to use `list-graphs`.

AWS CLI

To view a list of behavior graphs that your account is the administrator for

The following `list-graphs` example retrieves the behavior graphs that the calling account is the administrator for within the current Region.

```
aws detective list-graphs
```

Output:

```
{
  "GraphList": [
    {
      "Arn": "arn:aws:detective:us-east-1:111122223333:graph:123412341234",
      "CreatedTime": 1579736111000
    }
  ]
}
```

- For API details, see [ListGraphs](#) in *AWS CLI Command Reference*.

list-invitations

The following code example shows how to use `list-invitations`.

AWS CLI

To view a list of behavior graphs that an account is a member of or is invited to

The following `list-invitations` example retrieves the behavior graphs that the calling account has been invited to. The results include only open and accepted invitations. They do not include rejected invitations or removed memberships.

```
aws detective list-invitations
```

Output:

```
{
  "Invitations": [
    {
      "AccountId": "444455556666",
      "AdministratorId": "111122223333",
      "EmailAddress": "mmajor@example.com",
      "GraphArn": "arn:aws:detective:us-east-1:111122223333:graph:123412341234",
      "InvitedTime": 1579826107000,
      "MasterId": "111122223333",
      "Status": "INVITED",
      "UpdateTime": 1579826107000
    }
  ]
}
```

For more information, see [Viewing your list of behavior graph invitations](https://docs.aws.amazon.com/detective/latest/adminguide/member-view-graph-invitations.html) in the *Amazon Detective Administration Guide*.

- For API details, see [ListInvitations](#) in *AWS CLI Command Reference*.

list-members

The following code example shows how to use `list-members`.

AWS CLI

To list the member accounts in a behavior graph

The following `list-members` example retrieves the invited and enabled member accounts for the behavior graph `arn:aws:detective:us-east-1:111122223333:graph:123412341234`. The results do not include member accounts that were removed.

```
aws detective list-members \
  --graph-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234
```

Output:

```
{
  "MemberDetails": [
    {
```

```

    "AccountId": "444455556666",
    "AdministratorId": "111122223333",
    "EmailAddress": "mmajor@example.com",
    "GraphArn": "arn:aws:detective:us-
east-1:111122223333:graph:123412341234",
    "InvitedTime": 1579826107000,
    "MasterId": "111122223333",
    "Status": "INVITED",
    "UpdatedTime": 1579826107000
  },
  {
    "AccountId": "123456789012",
    "AdministratorId": "111122223333",
    "EmailAddress": "jstiles@example.com",
    "GraphArn": "arn:aws:detective:us-
east-1:111122223333:graph:123412341234",
    "InvitedTime": 1579826107000,
    "MasterId": "111122223333",
    "PercentOfGraphUtilization": 2,
    "PercentOfGraphUtilizationUpdatedTime": 1586287843,
    "Status": "ENABLED",
    "UpdatedTime": 1579973711000,
    "VolumeUsageInBytes": 200,
    "VolumeUsageUpdatedTime": 1586287843
  }
]
}

```

For more information, see [Viewing the list of accounts in a behavior graph](#) in the *Amazon Detective Administration Guide*.

- For API details, see [ListMembers](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To retrieve the tags assigned to a behavior graph

The following `list-tags-for-resource` example returns the tags assigned to the specified behavior graph.

```
aws detective list-tags-for-resource \  
  --resource-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234
```

Output:

```
{  
  "Tags": {  
    "Department" : "Finance"  
  }  
}
```

For more information, see [Managing tags for a behavior graph](#) in the *Amazon Detective Administration Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

reject-invitation

The following code example shows how to use `reject-invitation`.

AWS CLI

To reject an invitation to become a member account in a behavior graph

The following `reject-invitation` example rejects an invitation to become a member account in the behavior graph `arn:aws:detective:us-east-1:111122223333:graph:123412341234`.

```
aws detective reject-invitation \  
  --graph-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234
```

This command produces no output.

For more information, see [Responding to a behavior graph invitation](https://docs.aws.amazon.com/detective/latest/adminguide/member-invitation-response.html) in the *Amazon Detective Administration Guide*.

- For API details, see [RejectInvitation](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To assign a tag to a resource

The following `tag-resource` example assigns a value for the Department tag to the specified behavior graph.

```
aws detective tag-resource \  
  --resource-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234 \  
  --tags '{"Department":"Finance"}'
```

This command produces no output.

For more information, see [Managing tags for a behavior graph](#) in the *Amazon Detective Administration Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag value from a resource

The following `untag-resource` example removes the Department tag from the specified behavior graph.

```
aws detective untag-resource \  
  --resource-arn arn:aws:detective:us-east-1:111122223333:graph:123412341234 \  
  --tag-keys "Department"
```

This command produces no output.

For more information, see [Managing tags for a behavior graph](#) in the *Amazon Detective Administration Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

Device Farm examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Device Farm.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-device-pool

The following code example shows how to use `create-device-pool`.

AWS CLI

To create a device pool

The following command creates an Android device pool for a project:

```
aws devicefarm create-device-pool --name pool1 --rules file://  
device-pool-rules.json --project-arn "arn:aws:devicefarm:us-  
west-2:123456789012:project:070fc3ca-7ec1-4741-9c1f-d3e044efc506"
```

You can get the project ARN from the output of `create-project` or `list-projects`. The file `device-pool-rules.json` is a JSON document in the current folder that specifies the device platform:

```
[  
  {  
    "attribute": "PLATFORM",
```

```
    "operator": "EQUALS",
    "value": "\"ANDROID\""
  }
]
```

Output:

```
{
  "devicePool": {
    "rules": [
      {
        "operator": "EQUALS",
        "attribute": "PLATFORM",
        "value": "\"ANDROID\""
      }
    ],
    "type": "PRIVATE",
    "name": "pool1",
    "arn": "arn:aws:devicefarm:us-
west-2:123456789012:devicepool:070fc3ca-7ec1-4741-9c1f-
d3e044efc506/2aa8d2a9-5e73-47ca-b929-659cb34b7dcd"
  }
}
```

- For API details, see [CreateDevicePool](#) in *AWS CLI Command Reference*.

create-project

The following code example shows how to use create-project.

AWS CLI

To create a project

The following command creates a new project named my-project:

```
aws devicefarm create-project --name my-project
```

Output:

```
{
```

```
"project": {
  "name": "myproject",
  "arn": "arn:aws:devicefarm:us-
west-2:123456789012:project:070fc3ca-7ec1-4741-9c1f-d3e044efc506",
  "created": 1503612890.057
}
}
```

- For API details, see [CreateProject](#) in *AWS CLI Command Reference*.

create-upload

The following code example shows how to use create-upload.

AWS CLI

To create an upload

The following command creates an upload for an Android app:

```
aws devicefarm create-upload --project-arn "arn:aws:devicefarm:us-
west-2:123456789012:project:070fc3ca-7ec1-4741-9c1f-d3e044efc506" --name app.apk --
type ANDROID_APP
```

You can get the project ARN from the output of create-project or list-projects.

Output:

```
{
  "upload": {
    "status": "INITIALIZED",
    "name": "app.apk",
    "created": 1503614408.769,
    "url": "https://prod-us-west-2-uploads.s3-us-west-2.amazonaws.com/
arn%3Aaws%3Adevicefarm%3Aus-west-2%3A123456789012%3Aproject%3A070fc3ca-
c7e1-4471-91cf-d3e4efc50604/uploads/arn%3Aaws%3Adevicefarm%3Aus-
west-2%3A123456789012%3Aupload%3A070fc3ca-7ec1-4741-9c1f-d3e044efc506/dd72723a-
ae9e-4087-09e6-f4cea3599514/app.apk?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Date=20170824T224008Z&X-Amz-SignedHeaders=host&X-Amz-Expires=86400&X-Amz-
Credential=AKIAEXAMPLEPBUMBC3GA%2F20170824%2Fus-west-2%2Fs%2Faws4_request&X-Amz-
Signature=05050370c38894ef5bd09f5d009f36fc8f96fa4bb04e1bba9aca71b8dbe49a0f",
    "type": "ANDROID_APP",
```

```

    "arn": "arn:aws:devicefarm:us-
west-2:123456789012:upload:070fc3ca-7ec1-4741-9c1f-d3e044efc506/dd72723a-
ae9e-4087-09e6-f4cea3599514"
  }
}

```

Use the signed URL in the output to upload a file to Device Farm:

```

curl -T app.apk "https://prod-us-west-2-uploads.s3-us-west-2.amazonaws.com/
arn%3Aaws%3Adevicefarm%3Aus-west-2%3A123456789012%3Aproject%3A070fc3ca-
c7e1-4471-91cf-d3e4efc50604/uploads/arn%3Aaws%3Adevicefarm%3Aus-
west-2%3A123456789012%3Aupload%3A070fc3ca-7ec1-4741-9c1f-d3e044efc506/dd72723a-
ae9e-4087-09e6-f4cea3599514/app.apk?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Date=20170824T224008Z&X-Amz-SignedHeaders=host&X-Amz-Expires=86400&X-Amz-
Credential=AKIAEXAMPLEBUMBC3GA%2F20170824%2Fus-west-2%2Fs%2Faws4_request&X-Amz-
Signature=05050370c38894ef5bd09f5d009f36fc8f96fa4bb04e1bba9aca71b8dbe49a0f"

```

- For API details, see [CreateUpload](#) in *AWS CLI Command Reference*.

get-upload

The following code example shows how to use `get-upload`.

AWS CLI

To view an upload

The following command retrieves information about an upload:

```

aws devicefarm get-upload --arn "arn:aws:devicefarm:us-
west-2:123456789012:upload:070fc3ca-7ec1-4741-9c1f-d3e044efc506/dd72723a-
ae9e-4087-09e6-f4cea3599514"

```

You can get the upload ARN from the output of `create-upload`.

Output:

```

{
  "upload": {
    "status": "SUCCEEDED",
    "name": "app.apk",
    "created": 1505262773.186,

```

```

    "type": "ANDROID_APP",
    "arn": "arn:aws:devicefarm:us-
west-2:123456789012:upload:070fc3ca-7ec1-4741-9c1f-d3e044efc506/dd72723a-
ae9e-4087-09e6-f4cea3599514",
    "metadata": "{\\"device_admin\\":false,\\"activity_name\\":
\\"ccom.example.client.LauncherActivity\\",\\"version_name\\":\\"1.0.2.94\\",\\"screens
\\":[\\"small\\",\\"normal\\",\\"large\\",\\"xlarge\\"],\\"error_type\\":null,\\"sdk_version
\\":\\"16\\",\\"package_name\\":\\"com.example.client\\",\\"version_code\\":\\"20994\\",
\\"native_code\\":[\\"armeabi-v7a\\"],\\"target_sdk_version\\":\\"25\\"}"
  }
}

```

- For API details, see [GetUpload](#) in *AWS CLI Command Reference*.

list-projects

The following code example shows how to use `list-projects`.

AWS CLI

To list projects

The following retrieves a list of projects:

```
aws devicefarm list-projects
```

Output:

```

{
  "projects": [
    {
      "name": "myproject",
      "arn": "arn:aws:devicefarm:us-
west-2:123456789012:project:070fc3ca-7ec1-4741-9c1f-d3e044efc506",
      "created": 1503612890.057
    },
    {
      "name": "otherproject",
      "arn": "arn:aws:devicefarm:us-
west-2:123456789012:project:a5f5b752-8098-49d1-86bf-5f7682c1c77e",
      "created": 1505257519.337
    }
  ]
}

```

```
}
```

- For API details, see [ListProjects](#) in *AWS CLI Command Reference*.

AWS Direct Connect examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Direct Connect.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

accept-direct-connect-gateway-association-proposal

The following code example shows how to use `accept-direct-connect-gateway-association-proposal`.

AWS CLI

To accept a gateway association proposal

The following `accept-direct-connect-gateway-association-proposal` accepts the specified proposal.

```
aws directconnect accept-direct-connect-gateway-association-proposal \  
  --direct-connect-gateway-id 11460968-4ac1-4fd3-bdb2-00599EXAMPLE \  
  --proposal-id cb7f41cb-8128-43a5-93b1-dcaedEXAMPLE \  
  --associated-gateway-owner-account 111122223333
```

```
{
  "directConnectGatewayAssociation": {
    "directConnectGatewayId": "11460968-4ac1-4fd3-bdb2-00599EXAMPLE",
    "directConnectGatewayOwnerAccount": "111122223333",
    "associationState": "associating",
    "associatedGateway": {
      "id": "tgw-02f776b1a7EXAMPLE",
      "type": "transitGateway",
      "ownerAccount": "111122223333",
      "region": "us-east-1"
    },
    "associationId": "6441f8bf-5917-4279-ade1-9708bEXAMPLE",
    "allowedPrefixesToDirectConnectGateway": [
      {
        "cidr": "192.168.1.0/30"
      }
    ]
  }
}
```

For more information, see [Accepting or Rejecting a Transit Gateway Association Proposal](#) in the *AWS Direct Connect User Guide*.

- For API details, see [AcceptDirectConnectGatewayAssociationProposal](#) in *AWS CLI Command Reference*.

allocate-connection-on-interconnect

The following code example shows how to use `allocate-connection-on-interconnect`.

AWS CLI

To create a hosted connection on an interconnect

The following `allocate-connection-on-interconnect` command creates a hosted connection on an interconnect:

```
aws directconnect allocate-connection-on-interconnect --bandwidth 500Mbps --
connection-name mydcinterconnect --owner-account 123456789012 --interconnect-id
dxcon-fgktov66 --vlan 101
```

Output:


```
{
  "partnerName": "TIVIT",
  "vlan": 101,
  "ownerAccount": "123456789012",
  "connectionId": "dxcon-ffzc51m1",
  "connectionState": "ordering",
  "bandwidth": "500Mbps",
  "location": "TIVIT",
  "connectionName": "mydcinterconnect",
  "region": "sa-east-1"
}
```

- For API details, see [AllocateConnectionOnInterconnect](#) in *AWS CLI Command Reference*.

allocate-hosted-connection

The following code example shows how to use `allocate-hosted-connection`.

AWS CLI

To create a hosted connection on an interconnect

The following `allocate-hosted-connection` example creates a hosted connection on the specified interconnect.

```
aws directconnect allocate-hosted-connection \
  --bandwidth 500Mbps \
  --connection-name mydcinterconnect \
  --owner-account 123456789012
-connection-id dxcon-fgktov66
-vlan 101
```

Output:

```
{
  "partnerName": "TIVIT",
  "vlan": 101,
  "ownerAccount": "123456789012",
  "connectionId": "dxcon-ffzc51m1",
  "connectionState": "ordering",
  "bandwidth": "500Mbps",
```

```

    "location": "TIVIT",
    "connectionName": "mydcinterconnect",
    "region": "sa-east-1"
  }

```

- For API details, see [AllocateHostedConnection](#) in *AWS CLI Command Reference*.

allocate-private-virtual-interface

The following code example shows how to use `allocate-private-virtual-interface`.

AWS CLI

To provision a private virtual interface

The following `allocate-private-virtual-interface` command provisions a private virtual interface to be owned by a different customer:

```

aws directconnect allocate-private-virtual-interface --connection-id dxcon-
ffjrnx17 --owner-account 123456789012 --new-private-virtual-interface-allocation
virtualInterfaceName=PrivateVirtualInterface,vlan=1000,asn=65000,authKey=asdf34example,amaz

```

Output:

```

{
  "virtualInterfaceState": "confirming",
  "asn": 65000,
  "vlan": 1000,
  "customerAddress": "192.168.1.2/30",
  "ownerAccount": "123456789012",
  "connectionId": "dxcon-ffjrnx17",
  "virtualInterfaceId": "dxvif-fgy8orxu",
  "authKey": "asdf34example",
  "routeFilterPrefixes": [],
  "location": "TIVIT",
  "customerRouterConfig": "<?xml version=\"1.0\" encoding=\"UTF-8\"?
>\n <logical_connection id=\"dxvif-fgy8orxu\">\n <vlan>1000</
vlan>\n <customer_address>192.168.1.2/30</customer_address>\n
<amazon_address>192.168.1.1/30</amazon_address>\n <bgp_asn>65000</bgp_asn>\n
<bgp_auth_key>asdf34example</bgp_auth_key>\n <amazon_bgp_asn>7224</amazon_bgp_asn>
\n <connection_type>private</connection_type>\n</logical_connection>\n",
  "amazonAddress": "192.168.1.1/30",

```

```

    "virtualInterfaceType": "private",
    "virtualInterfaceName": "PrivateVirtualInterface"
  }

```

- For API details, see [AllocatePrivateVirtualInterface](#) in *AWS CLI Command Reference*.

allocate-public-virtual-interface

The following code example shows how to use `allocate-public-virtual-interface`.

AWS CLI

To provision a public virtual interface

The following `allocate-public-virtual-interface` command provisions a public virtual interface to be owned by a different customer:

```

aws directconnect allocate-public-virtual-interface --connection-id dxcon-
ffjrkx17 --owner-account 123456789012 --new-public-virtual-interface-allocation
  virtualInterfaceName=PublicVirtualInterface,vlan=2000,asn=65000,authKey=asdf34example,amaz
  {cidr=203.0.113.4/30}]

```

Output:

```

{
  "virtualInterfaceState": "confirming",
  "asn": 65000,
  "vlan": 2000,
  "customerAddress": "203.0.113.2/30",
  "ownerAccount": "123456789012",
  "connectionId": "dxcon-ffjrkx17",
  "virtualInterfaceId": "dxvif-fg9xo9vp",
  "authKey": "asdf34example",
  "routeFilterPrefixes": [
    {
      "cidr": "203.0.113.0/30"
    },
    {
      "cidr": "203.0.113.4/30"
    }
  ],
  "location": "TIVIT",

```

```

    "customerRouterConfig": "<?xml version=\"1.0\" encoding=\"UTF-8\"?
>\n<logical_connection id=\"dxvif-fg9xo9vp\">\n  <vlan>2000</
vlan>\n  <customer_address>203.0.113.2/30</customer_address>\n
  <amazon_address>203.0.113.1/30</amazon_address>\n  <bgp_asn>65000</bgp_asn>\n
  <bgp_auth_key>asdf34example</bgp_auth_key>\n  <amazon_bgp_asn>7224</amazon_bgp_asn>
\n  <connection_type>public</connection_type>\n</logical_connection>\n",
    "amazonAddress": "203.0.113.1/30",
    "virtualInterfaceType": "public",
    "virtualInterfaceName": "PublicVirtualInterface"
  }

```

- For API details, see [AllocatePublicVirtualInterface](#) in *AWS CLI Command Reference*.

allocate-transit-virtual-interface

The following code example shows how to use `allocate-transit-virtual-interface`.

AWS CLI

To provision a transit virtual interface to be owned by the specified AWS account

The following `allocate-transit-virtual-interface` example provisions a transit virtual interface for the specified account.

```

aws directconnect allocate-transit-virtual-interface \
  --connection-id dxlag-fEXAMPLE \
  --owner-account 123456789012 \
  --new-transit-virtual-interface-allocation "virtualInterfaceName=Example Transit
Virtual
Interface,vlan=126,asn=65110,mtu=1500,authKey=0xzxcgA9YoW9h58u8SEXAMPLE,amazonAddress=192.16

```

Output:

```

{
  "virtualInterface": {
    "ownerAccount": "123456789012",
    "virtualInterfaceId": "dxvif-fEXAMPLE",
    "location": "loc1",
    "connectionId": "dxlag-fEXAMPLE",
    "virtualInterfaceType": "transit",
    "virtualInterfaceName": "Example Transit Virtual Interface",
    "vlan": 126,

```

```

    "asn": 65110,
    "amazonSideAsn": 7224,
    "authKey": "0xzxgA9YoW9h58u8SEXAMPLE",
    "amazonAddress": "192.168.1.1/30",
    "customerAddress": "192.168.1.2/30",
    "addressFamily": "ipv4",
    "virtualInterfaceState": "confirming",
    "customerRouterConfig": "<?xml version=\"1.0\" encoding=
\\\"UTF-8\\\"?>\\n<logical_connection id=\\\"dxvif-fEXAMPLE\\\">\\n  <vlan>126</
vlan>\\n  <customer_address>192.168.1.2/30</customer_address>\\n
  <amazon_address>192.168.1.1/30</amazon_address>\\n  <bgp_asn>65110</bgp_asn>\\n
  <bgp_auth_key>0xzxgA9YoW9h58u8SEXAMPLE</bgp_auth_key>\\n  <amazon_bgp_asn>7224</
amazon_bgp_asn>\\n  <connection_type>transit</connection_type>\\n</logical_connection>
\\n\",
    "mtu": 1500,
    "jumboFrameCapable": true,
    "virtualGatewayId": "",
    "directConnectGatewayId": "",
    "routeFilterPrefixes": [],
    "bgpPeers": [
      {
        "bgpPeerId": "dxpeer-fEXAMPLE",
        "asn": 65110,
        "authKey": "0xzxgA9YoW9h58u8EXAMPLE",
        "addressFamily": "ipv4",
        "amazonAddress": "192.168.1.1/30",
        "customerAddress": "192.168.1.2/30",
        "bgpPeerState": "pending",
        "bgpStatus": "down",
        "awsDeviceV2": "loc1-26wz6vEXAMPLE"
      }
    ],
    "region": "sa-east-1",
    "awsDeviceV2": "loc1-26wz6vEXAMPLE",
    "tags": [
      {
        "key": "Tag",
        "value": "Example"
      }
    ]
  }
}

```

For more information, see [Creating a Hosted Transit Virtual Interface](#) in the *AWS Direct Connect User Guide*.

- For API details, see [AllocateTransitVirtualInterface](#) in *AWS CLI Command Reference*.

associate-connection-with-lag

The following code example shows how to use `associate-connection-with-lag`.

AWS CLI

To associate a connection with a LAG

The following example associates the specified connection with the specified LAG.

Command:

```
aws directconnect associate-connection-with-lag --lag-id dxlag-fhccu14t --
connection-id dxcon-fg9607vm
```

Output:

```
{
  "ownerAccount": "123456789012",
  "connectionId": "dxcon-fg9607vm",
  "lagId": "dxlag-fhccu14t",
  "connectionState": "requested",
  "bandwidth": "1Gbps",
  "location": "EqDC2",
  "connectionName": "Con2ForLag",
  "region": "us-east-1"
}
```

- For API details, see [AssociateConnectionWithLag](#) in *AWS CLI Command Reference*.

associate-hosted-connection

The following code example shows how to use `associate-hosted-connection`.

AWS CLI

To associate a hosted connection with a LAG

The following example associates the specified hosted connection with the specified LAG.

Command:

```
aws directconnect associate-hosted-connection --parent-connection-id dxlag-fhccu14t
--connection-id dxcon-fg9607vm
```

Output:

```
{
  "partnerName": "TIVIT",
  "vlan": 101,
  "ownerAccount": "123456789012",
  "connectionId": "dxcon-fg9607vm",
  "lagId": "dxlag-fhccu14t",
  "connectionState": "ordering",
  "bandwidth": "500Mbps",
  "location": "TIVIT",
  "connectionName": "mydcinterconnect",
  "region": "sa-east-1"
}
```

- For API details, see [AssociateHostedConnection](#) in *AWS CLI Command Reference*.

associate-virtual-interface

The following code example shows how to use `associate-virtual-interface`.

AWS CLI

To associate a virtual interface with a connection

The following example associates the specified virtual interface with the specified LAG. Alternatively, to associate the virtual interface with a connection, specify the ID of an AWS Direct Connect connection for `--connection-id`; for example, `dxcon-ffnikghc`.

Command:

```
aws directconnect associate-virtual-interface --connection-id dxlag-ffjhj9lx --
virtual-interface-id dxvif-fgputw0j
```

Output:

```
{
  "virtualInterfaceState": "pending",
  "asn": 65000,
  "vlan": 123,
  "customerAddress": "169.254.255.2/30",
  "ownerAccount": "123456789012",
  "connectionId": "dxdlag-ffjhj9lx",
  "addressFamily": "ipv4",
  "virtualGatewayId": "vgw-38e90b51",
  "virtualInterfaceId": "dxvif-fgputw0j",
  "authKey": "0x123pK5_VBqv.UQ3kJ4123_",
  "routeFilterPrefixes": [],
  "location": "CSVA1",
  "bgpPeers": [
    {
      "bgpStatus": "down",
      "customerAddress": "169.254.255.2/30",
      "addressFamily": "ipv4",
      "authKey": "0x123pK5_VBqv.UQ3kJ4123_",
      "bgpPeerState": "deleting",
      "amazonAddress": "169.254.255.1/30",
      "asn": 65000
    },
    {
      "bgpStatus": "down",
      "customerAddress": "169.254.255.2/30",
      "addressFamily": "ipv4",
      "authKey": "0x123pK5_VBqv.UQ3kJ4123_",
      "bgpPeerState": "pending",
      "amazonAddress": "169.254.255.1/30",
      "asn": 65000
    }
  ],
  "customerRouterConfig": "<?xml version='1.0' encoding='UTF-8'?>
  <logical_connection id='dxvif-fgputw0j'>
    <vlan>123</vlan>
    <customer_address>169.254.255.2/30</customer_address>
    <amazon_address>169.254.255.1/30</amazon_address>
    <bgp_asn>65000</bgp_asn>
    <bgp_auth_key>0x123pK5_VBqv.UQ3kJ4123_</bgp_auth_key>
    <amazon_bgp_asn>7224</amazon_bgp_asn>
    <connection_type>private</connection_type>
  </logical_connection>
  ",
  "amazonAddress": "169.254.255.1/30",
  "virtualInterfaceType": "private",

```



```
"virtualInterfaceName": "VIF1A"
}
```

- For API details, see [AssociateVirtualInterface](#) in *AWS CLI Command Reference*.

confirm-connection

The following code example shows how to use `confirm-connection`.

AWS CLI

To confirm the creation of a hosted connection on an interconnect

The following `confirm-connection` command confirms the creation of a hosted connection on an interconnect:

```
aws directconnect confirm-connection --connection-id dxcon-fg2wi7hy
```

Output:

```
{
  "connectionState": "pending"
}
```

- For API details, see [ConfirmConnection](#) in *AWS CLI Command Reference*.

confirm-private-virtual-interface

The following code example shows how to use `confirm-private-virtual-interface`.

AWS CLI

To accept ownership of a private virtual interface

The following `confirm-private-virtual-interface` command accepts ownership of a private virtual interface created by another customer:

```
aws directconnect confirm-private-virtual-interface --virtual-interface-id dxvif-
fgy8orxu --virtual-gateway-id vgw-e4a47df9
```

Output:

```
{
  "virtualInterfaceState": "pending"
}
```

- For API details, see [ConfirmPrivateVirtualInterface](#) in *AWS CLI Command Reference*.

confirm-public-virtual-interface

The following code example shows how to use `confirm-public-virtual-interface`.

AWS CLI**To accept ownership of a public virtual interface**

The following `confirm-public-virtual-interface` command accepts ownership of a public virtual interface created by another customer:

```
aws directconnect confirm-public-virtual-interface --virtual-interface-id dxvif-
fg9xo9vp
```

Output:

```
{
  "virtualInterfaceState": "verifying"
}
```

- For API details, see [ConfirmPublicVirtualInterface](#) in *AWS CLI Command Reference*.

confirm-transit-virtual-interface

The following code example shows how to use `confirm-transit-virtual-interface`.

AWS CLI**To accept ownership of a transit virtual interface**

The following `confirm-transit-virtual-interface` accepts ownership of a transit virtual interface created by another customer.

```
aws directconnect confirm-transit-virtual-interface \  
  --virtual-interface-id dxvif-fEXAMPLE \  
  --direct-connect-gateway-id 4112ccf9-25e9-4111-8237-b6c5dEXAMPLE
```

Output:

```
{  
  "virtualInterfaceState": "pending"  
}
```

For more information, see [Accepting a Hosted Virtual Interface](#) in the *AWS Direct Connect User Guide*.

- For API details, see [ConfirmTransitVirtualInterface](#) in *AWS CLI Command Reference*.

create-bgp-peer

The following code example shows how to use `create-bgp-peer`.

AWS CLI

To create an IPv6 BGP peering session

The following example creates an IPv6 BGP peering session on private virtual interface `dxvif-fg1vuj3d`. The peer IPv6 addresses are automatically allocated by Amazon.

Command:

```
aws directconnect create-bgp-peer --virtual-interface-id dxvif-fg1vuj3d --new-bgp-peer asn=64600,addressFamily=ipv6
```

Output:

```
{  
  "virtualInterface": {  
    "virtualInterfaceState": "available",  
    "asn": 65000,  
    "vlan": 125,  
    "customerAddress": "169.254.255.2/30",  
    "ownerAccount": "123456789012",
```

```

"connectionId": "dxcon-fguhmq1c",
"addressFamily": "ipv4",
"virtualGatewayId": "vgw-f9eb0c90",
"virtualInterfaceId": "dxvif-fg1vuj3d",
"authKey": "0xC_ukbCer16EYA0example",
"routeFilterPrefixes": [],
"location": "EqDC2",
"bgpPeers": [
  {
    "bgpStatus": "down",
    "customerAddress": "169.254.255.2/30",
    "addressFamily": "ipv4",
    "authKey": "0xC_ukbCer16EYA0uexample",
    "bgpPeerState": "available",
    "amazonAddress": "169.254.255.1/30",
    "asn": 65000
  },
  {
    "bgpStatus": "down",
    "customerAddress": "2001:db8:1100:2f0:0:1:9cb4:4216/125",
    "addressFamily": "ipv6",
    "authKey": "0xS27kAIU_VHPjjAexample",
    "bgpPeerState": "pending",
    "amazonAddress": "2001:db8:1100:2f0:0:1:9cb4:4211/125",
    "asn": 64600
  }
],
"customerRouterConfig": "<?xml version=\"1.0\" encoding=
\"UTF-8\"?>\n<logical_connection id=\"dxvif-fg1vuj3d\">\n  <vlan>125</
vlan>\n  <customer_address>169.254.255.2/30</customer_address>\n
  <amazon_address>169.254.255.1/30</amazon_address>\n  <bgp_asn>65000</
bgp_asn>\n  <bgp_auth_key>0xC_ukbCer16EYA0uexample</bgp_auth_key>\n
  <ipv6_customer_address>2001:db8:1100:2f0:0:1:9cb4:4216/125</ipv6_customer_address>
\n  <ipv6_amazon_address>2001:db8:1100:2f0:0:1:9cb4:4211/125</ipv6_amazon_address>\n
  <ipv6_bgp_asn>64600</ipv6_bgp_asn>\n  <ipv6_bgp_auth_key>0xS27kAIU_VHPjjAexample</
ipv6_bgp_auth_key>\n  <amazon_bgp_asn>7224</amazon_bgp_asn>\n
  <connection_type>private</connection_type>\n</logical_connection>\n",
  "amazonAddress": "169.254.255.1/30",
  "virtualInterfaceType": "private",
  "virtualInterfaceName": "Test"
}
}

```

- For API details, see [CreateBgpPeer](#) in *AWS CLI Command Reference*.

create-connection

The following code example shows how to use `create-connection`.

AWS CLI

To create a connection from your network to an AWS Direct Connect location

The following `create-connection` command creates a connection from your network to an AWS Direct Connect location:

```
aws directconnect create-connection --location TIVIT --bandwidth 1Gbps --connection-name "Connection to AWS"
```

Output:

```
{
  "ownerAccount": "123456789012",
  "connectionId": "dxcon-fg31dyv6",
  "connectionState": "requested",
  "bandwidth": "1Gbps",
  "location": "TIVIT",
  "connectionName": "Connection to AWS",
  "region": "sa-east-1"
}
```

- For API details, see [CreateConnection](#) in *AWS CLI Command Reference*.

create-direct-connect-gateway-association-proposal

The following code example shows how to use `create-direct-connect-gateway-association-proposal`.

AWS CLI

To create a proposal to associate the specified transit gateway with the specified Direct Connect gateway

The following `create-direct-connect-gateway-association-proposal` example creates a proposal that associates the specified transit gateway with the specified Direct Connect gateway.

```
aws directconnect create-direct-connect-gateway-association-proposal \  
  --direct-connect-gateway-id 11460968-4ac1-4fd3-bdb2-00599EXAMPLE \  
  --direct-connect-gateway-owner-account 111122223333 \  
  --gateway-id tgw-02f776b1a7EXAMPLE \  
  --add-allowed-prefixes-to-direct-connect-gateway cidr=192.168.1.0/30
```

Output:

```
{  
  "directConnectGatewayAssociationProposal": {  
    "proposalId": "cb7f41cb-8128-43a5-93b1-dcaedEXAMPLE",  
    "directConnectGatewayId": "11460968-4ac1-4fd3-bdb2-00599EXAMPLE",  
    "directConnectGatewayOwnerAccount": "111122223333",  
    "proposalState": "requested",  
    "associatedGateway": {  
      "id": "tgw-02f776b1a7EXAMPLE",  
      "type": "transitGateway",  
      "ownerAccount": "111122223333",  
      "region": "us-east-1"  
    },  
    "requestedAllowedPrefixesToDirectConnectGateway": [  
      {  
        "cidr": "192.168.1.0/30"  
      }  
    ]  
  }  
}
```

For more information, see [Creating a Transit Gateway Association Proposal](#) in the *AWS Direct Connect User Guide*.

- For API details, see [CreateDirectConnectGatewayAssociationProposal](#) in *AWS CLI Command Reference*.

create-direct-connect-gateway-association

The following code example shows how to use `create-direct-connect-gateway-association`.

AWS CLI

To associate a virtual private gateway with a Direct Connect gateway

The following example associates virtual private gateway `vgw-6efe725e` with Direct Connect gateway `5f294f92-bafb-4011-916d-9b0bexample`. You must run the command in the region in which the virtual private gateway is located.

Command:

```
aws directconnect create-direct-connect-gateway-association --direct-connect-gateway-id 5f294f92-bafb-4011-916d-9b0bexample --virtual-gateway-id vgw-6efe725e
```

Output:

```
{
  "directConnectGatewayAssociation": {
    "associationState": "associating",
    "virtualGatewayOwnerAccount": "123456789012",
    "directConnectGatewayId": "5f294f92-bafb-4011-916d-9b0bexample",
    "virtualGatewayId": "vgw-6efe725e",
    "virtualGatewayRegion": "us-east-2"
  }
}
```

- For API details, see [CreateDirectConnectGatewayAssociation](#) in *AWS CLI Command Reference*.

create-direct-connect-gateway

The following code example shows how to use `create-direct-connect-gateway`.

AWS CLI

To create a Direct Connect gateway

The following example creates a Direct Connect gateway with the name `DxGateway1`.

Command:

```
aws directconnect create-direct-connect-gateway --direct-connect-gateway-name "DxGateway1"
```

Output:

```
{
```

```
"directConnectGateway": {
  "amazonSideAsn": 64512,
  "directConnectGatewayId": "5f294f92-bafb-4011-916d-9b0bdexample",
  "ownerAccount": "123456789012",
  "directConnectGatewayName": "DxGateway1",
  "directConnectGatewayState": "available"
}
```

- For API details, see [CreateDirectConnectGateway](#) in *AWS CLI Command Reference*.

create-interconnect

The following code example shows how to use `create-interconnect`.

AWS CLI

To create an interconnect between a partner's network and AWS

The following `create-interconnect` command creates an interconnect between an AWS Direct Connect partner's network and a specific AWS Direct Connect location:

```
aws directconnect create-interconnect --interconnect-name "1G Interconnect to AWS"
--bandwidth 1Gbps --location TIVIT
```

Output:

```
{
  "region": "sa-east-1",
  "bandwidth": "1Gbps",
  "location": "TIVIT",
  "interconnectName": "1G Interconnect to AWS",
  "interconnectId": "dxcon-fgktov66",
  "interconnectState": "requested"
}
```

- For API details, see [CreateInterconnect](#) in *AWS CLI Command Reference*.

create-lag

The following code example shows how to use `create-lag`.

AWS CLI

To create a LAG with new connections

The following example creates a LAG and requests two new AWS Direct Connect connections for the LAG with a bandwidth of 1 Gbps.

Command:

```
aws directconnect create-lag --location CSVA1 --number-of-connections 2 --
connections-bandwidth 1Gbps --lag-name 1GBLag
```

Output:

```
{
  "awsDevice": "CSVA1-23u8t1paz8iks",
  "numberOfConnections": 2,
  "lagState": "pending",
  "ownerAccount": "123456789012",
  "lagName": "1GBLag",
  "connections": [
    {
      "ownerAccount": "123456789012",
      "connectionId": "dxcon-ffqr6x5q",
      "lagId": "dxlag-ffjhj91x",
      "connectionState": "requested",
      "bandwidth": "1Gbps",
      "location": "CSVA1",
      "connectionName": "Requested Connection 1 for Lag dxlag-ffjhj91x",
      "region": "us-east-1"
    },
    {
      "ownerAccount": "123456789012",
      "connectionId": "dxcon-fflqyj95",
      "lagId": "dxlag-ffjhj91x",
      "connectionState": "requested",
      "bandwidth": "1Gbps",
      "location": "CSVA1",
      "connectionName": "Requested Connection 2 for Lag dxlag-ffjhj91x",
      "region": "us-east-1"
    }
  ],
  "lagId": "dxlag-ffjhj91x",
```

```
"minimumLinks": 0,  
"connectionsBandwidth": "1Gbps",  
"region": "us-east-1",  
"location": "CSVA1"  
}
```

To create a LAG using an existing connection

The following example creates a LAG from an existing connection in your account, and requests a second new connection for the LAG with the same bandwidth and location as the existing connection.

Command:

```
aws directconnect create-lag --location EqDC2 --number-of-connections 2 --  
connections-bandwidth 1Gbps --lag-name 2ConnLAG --connection-id dxcon-fgk145dr
```

Output:

```
{  
  "awsDevice": "EqDC2-4h6ce2r1bes6",  
  "numberOfConnections": 2,  
  "lagState": "pending",  
  "ownerAccount": "123456789012",  
  "lagName": "2ConnLAG",  
  "connections": [  
    {  
      "ownerAccount": "123456789012",  
      "connectionId": "dxcon-fh6ljcv0",  
      "lagId": "dxlag-fhccu14t",  
      "connectionState": "requested",  
      "bandwidth": "1Gbps",  
      "location": "EqDC2",  
      "connectionName": "Requested Connection 1 for Lag dxlag-fhccu14t",  
      "region": "us-east-1"  
    },  
    {  
      "ownerAccount": "123456789012",  
      "connectionId": "dxcon-fgk145dr",  
      "lagId": "dxlag-fhccu14t",  
      "connectionState": "down",  
      "bandwidth": "1Gbps",  
      "location": "EqDC2",
```

```

        "connectionName": "VAConn1",
        "region": "us-east-1"
    }
],
"lagId": "dxlag-fhccu14t",
"minimumLinks": 0,
"connectionsBandwidth": "1Gbps",
"region": "us-east-1",
"location": "EqDC2"
}

```

- For API details, see [CreateLag](#) in *AWS CLI Command Reference*.

create-private-virtual-interface

The following code example shows how to use `create-private-virtual-interface`.

AWS CLI

To create a private virtual interface

The following `create-private-virtual-interface` command creates a private virtual interface:

```

aws directconnect create-private-virtual-interface --
connection-id dxcon-ffjrnx17 --new-private-virtual-interface
virtualInterfaceName=PrivateVirtualInterface,vlan=101,asn=65000,authKey=asdf34example,amaz
aba37db6

```

Output:

```

{
  "virtualInterfaceState": "pending",
  "asn": 65000,
  "vlan": 101,
  "customerAddress": "192.168.1.2/30",
  "ownerAccount": "123456789012",
  "connectionId": "dxcon-ffjrnx17",
  "virtualGatewayId": "vgw-aba37db6",
  "virtualInterfaceId": "dxvif-ffhkh74f",
  "authKey": "asdf34example",
  "routeFilterPrefixes": [],

```

```

    "location": "TIVIT",
    "customerRouterConfig": "<?xml version=\"1.0\" encoding=
\\\"UTF-8\\\"?>\\n<logical_connection id=\\\"dxvif-ffhkh74f\\\">\\n  <vlan>101</
vlan>\\n  <customer_address>192.168.1.2/30</customer_address>\\n
  <amazon_address>192.168.1.1/30</amazon_address>\\n  <bgp_asn>65000</bgp_asn>\\n
  <bgp_auth_key>asdf34example</bgp_auth_key>\\n  <amazon_bgp_asn>7224</amazon_bgp_asn>
\\n  <connection_type>private</connection_type>\\n</logical_connection>\\n",
    "amazonAddress": "192.168.1.1/30",
    "virtualInterfaceType": "private",
    "virtualInterfaceName": "PrivateVirtualInterface"
  }

```

- For API details, see [CreatePrivateVirtualInterface](#) in *AWS CLI Command Reference*.

create-public-virtual-interface

The following code example shows how to use create-public-virtual-interface.

AWS CLI

To create a public virtual interface

The following create-public-virtual-interface command creates a public virtual interface:

```

aws directconnect create-public-virtual-interface --
connection-id dxcon-ffjrkx17 --new-public-virtual-interface
virtualInterfaceName=PublicVirtualInterface,vlan=2000,asn=65000,authKey=asdf34example,amazonAddress=192.168.1.1/30,{cidr=203.0.113.4/30}

```

Output:

```

{
  "virtualInterfaceState": "verifying",
  "asn": 65000,
  "vlan": 2000,
  "customerAddress": "203.0.113.2/30",
  "ownerAccount": "123456789012",
  "connectionId": "dxcon-ffjrkx17",
  "virtualInterfaceId": "dxvif-fgh0hcrk",
  "authKey": "asdf34example",
  "routeFilterPrefixes": [

```

```

    {
      "cidr": "203.0.113.0/30"
    },
    {
      "cidr": "203.0.113.4/30"
    }
  ],
  "location": "TIVIT",
  "customerRouterConfig": "<?xml version=\"1.0\" encoding=\"UTF-8\"?
>\n<logical_connection id=\"dxvif-fgh0hcrk\">\n  <vlan>2000</
vlan>\n  <customer_address>203.0.113.2/30</customer_address>\n
  <amazon_address>203.0.113.1/30</amazon_address>\n  <bgp_asn>65000</bgp_asn>\n
  <bgp_auth_key>asdf34example</bgp_auth_key>\n  <amazon_bgp_asn>7224</amazon_bgp_asn>
\n  <connection_type>public</connection_type>\n</logical_connection>\n",
  "amazonAddress": "203.0.113.1/30",
  "virtualInterfaceType": "public",
  "virtualInterfaceName": "PublicVirtualInterface"
}

```

- For API details, see [CreatePublicVirtualInterface](#) in *AWS CLI Command Reference*.

create-transit-virtual-interface

The following code example shows how to use `create-transit-virtual-interface`.

AWS CLI

To create a transit virtual interface

The following `create-transit-virtual-interface` example creates a transit virtual interface for the specified connection.

```

aws directconnect create-transit-virtual-interface \
  --connection-id dxlag-fEXAMPLE \
  --new-transit-virtual-interface "virtualInterfaceName=Example Transit Virtual
Interface,vlan=126,asn=65110,mtu=1500,authKey=0xxzgA9YoW9h58u8SvEXAMPLE,amazonAddress=192.1
aada-5a1baEXAMPLE,tags=[{key=Tag,value=Example}]"

```

Output:

```

{
  "virtualInterface": {

```

```

"ownerAccount": "1111222233333",
"virtualInterfaceId": "dxvif-fEXAMPLE",
"location": "loc1",
"connectionId": "dxlag-fEXAMPLE",
"virtualInterfaceType": "transit",
"virtualInterfaceName": "Example Transit Virtual Interface",
"vlan": 126,
"asn": 65110,
"amazonSideAsn": 4200000000,
"authKey": "0xzxgA9YoW9h58u8SEXAMPLE",
"amazonAddress": "192.168.1.1/30",
"customerAddress": "192.168.1.2/30",
"addressFamily": "ipv4",
"virtualInterfaceState": "pending",
"customerRouterConfig": "<?xml version=\"1.0\" encoding=
\\\"UTF-8\\\"?>\\n<logical_connection id=\\\"dxvif-fEXAMPLE\\\">\\n  <vlan>126</
vlan>\\n  <customer_address>192.168.1.2/30</customer_address>\\n
  <amazon_address>192.168.1.1/30</amazon_address>\\n  <bgp_asn>65110</
bgp_asn>\\n  <bgp_auth_key>0xzxgA9YoW9h58u8Sv0mXRTw</bgp_auth_key>\\n
  <amazon_bgp_asn>4200000000</amazon_bgp_asn>\\n  <connection_type>transit</
connection_type>\\n</logical_connection>\\n\",
"mtu": 1500,
"jumboFrameCapable": true,
"virtualGatewayId": "",
"directConnectGatewayId": "8384da05-13ce-4a91-aada-5a1baEXAMPLE",
"routeFilterPrefixes": [],
"bgpPeers": [
  {
    "bgpPeerId": "dxpeer-EXAMPLE",
    "asn": 65110,
    "authKey": "0xzxgA9YoW9h58u8SEXAMPLE",
    "addressFamily": "ipv4",
    "amazonAddress": "192.168.1.1/30",
    "customerAddress": "192.168.1.2/30",
    "bgpPeerState": "pending",
    "bgpStatus": "down",
    "awsDeviceV2": "loc1-26wz6vEXAMPLE"
  }
],
"region": "sa-east-1",
"awsDeviceV2": "loc1-26wz6vEXAMPLE",
"tags": [
  {
    "key": "Tag",

```

```

    "value": "Example"
  }
]
}
}

```

For more information, see [Creating a Transit Virtual Interface to the Direct Connect Gateway](#) in the *AWS Direct Connect User Guide*.

- For API details, see [CreateTransitVirtualInterface](#) in *AWS CLI Command Reference*.

delete-bgp-peer

The following code example shows how to use `delete-bgp-peer`.

AWS CLI

To delete a BGP peer from a virtual interface

The following example deletes the IPv6 BGP peer from virtual interface `dxvif-fg1vuj3d`.

Command:

```
aws directconnect delete-bgp-peer --virtual-interface-id dxvif-fg1vuj3d --asn 64600
--customer-address 2001:db8:1100:2f0:0:1:9cb4:4216/125
```

Output:

```
{
  "virtualInterface": {
    "virtualInterfaceState": "available",
    "asn": 65000,
    "vlan": 125,
    "customerAddress": "169.254.255.2/30",
    "ownerAccount": "123456789012",
    "connectionId": "dxcon-fguhmq1c",
    "addressFamily": "ipv4",
    "virtualGatewayId": "vgw-f9eb0c90",
    "virtualInterfaceId": "dxvif-fg1vuj3d",
    "authKey": "0xC_ukbCerl6EYA0example",
    "routeFilterPrefixes": [],
    "location": "EqDC2",
    "bgpPeers": [

```

```

    {
      "bgpStatus": "down",
      "customerAddress": "169.254.255.2/30",
      "addressFamily": "ipv4",
      "authKey": "0xC_ukbCerl6EYA0uexample",
      "bgpPeerState": "available",
      "amazonAddress": "169.254.255.1/30",
      "asn": 65000
    },
    {
      "bgpStatus": "down",
      "customerAddress": "2001:db8:1100:2f0:0:1:9cb4:4216/125",
      "addressFamily": "ipv6",
      "authKey": "0xS27kAIU_VHPjjAexample",
      "bgpPeerState": "deleting",
      "amazonAddress": "2001:db8:1100:2f0:0:1:9cb4:4211/125",
      "asn": 64600
    }
  ],
  "customerRouterConfig": "<?xml version=\"1.0\" encoding=
  \"UTF-8\"?>\n<logical_connection id=\"dxvif-fg1vuj3d\">\n  <vlan>125</
  vlan>\n  <customer_address>169.254.255.2/30</customer_address>\n
  <amazon_address>169.254.255.1/30</amazon_address>\n  <bgp_asn>65000</bgp_asn>\n
  <bgp_auth_key>0xC_ukbCerl6EYA0example</bgp_auth_key>\n  <amazon_bgp_asn>7224</
  amazon_bgp_asn>\n  <connection_type>private</connection_type>\n</logical_connection>
  \n",
  "amazonAddress": "169.254.255.1/30",
  "virtualInterfaceType": "private",
  "virtualInterfaceName": "Test"
}
}

```

- For API details, see [DeleteBgpPeer](#) in *AWS CLI Command Reference*.

delete-connection

The following code example shows how to use delete-connection.

AWS CLI

To delete a connection

The following delete-connection command deletes the specified connection:


```
aws directconnect delete-connection --connection-id dxcon-fg31dyv6
```

Output:

```
{
  "ownerAccount": "123456789012",
  "connectionId": "dxcon-fg31dyv6",
  "connectionState": "deleted",
  "bandwidth": "1Gbps",
  "location": "TIVIT",
  "connectionName": "Connection to AWS",
  "region": "sa-east-1"
}
```

- For API details, see [DeleteConnection](#) in *AWS CLI Command Reference*.

delete-direct-connect-gateway-association

The following code example shows how to use `delete-direct-connect-gateway-association`.

AWS CLI**To delete a Direct Connect gateway association**

The following `delete-direct-connect-gateway-association` example deletes the Direct Connect gateway association with a transit gateway that has the specified association ID.

```
aws directconnect delete-direct-connect-gateway-association --association-id
be85116d-46eb-4b43-a27a-da0c2ad648de
```

Output:

```
{
  "directConnectGatewayAssociation": {
    "directConnectGatewayId": "11460968-4ac1-4fd3-bdb2-00599EXAMPLE",
    "directConnectGatewayOwnerAccount": "123456789012",
    "associationState": "disassociating",
    "associatedGateway": {
      "id": "tgw-095b3b0b54EXAMPLE",
      "type": "transitGateway",

```

```

        "ownerAccount": "123456789012",
        "region": "us-east-1"
    },
    "associationId": " be85116d-46eb-4b43-a27a-da0c2ad648deEXAMPLE ",
    "allowedPrefixesToDirectConnectGateway": [
        {
            "cidr": "192.0.1.0/28"
        }
    ]
}
}

```

For more information, see [Associating and Disassociating Transit Gateways](#) in the *AWS Direct Connect User Guide*.

- For API details, see [DeleteDirectConnectGatewayAssociation](#) in *AWS CLI Command Reference*.

delete-direct-connect-gateway

The following code example shows how to use `delete-direct-connect-gateway`.

AWS CLI

To delete a Direct Connect gateway

The following example deletes Direct Connect gateway `5f294f92-bafb-4011-916d-9b0bexample`.

Command:

```
aws directconnect delete-direct-connect-gateway --direct-connect-gateway-id
5f294f92-bafb-4011-916d-9b0bexample
```

Output:

```

{
  "directConnectGateway": {
    "amazonSideAsn": 64512,
    "directConnectGatewayId": "5f294f92-bafb-4011-916d-9b0bexample",
    "ownerAccount": "123456789012",
    "directConnectGatewayName": "DxGateway1",
    "directConnectGatewayState": "deleting"
  }
}

```

```
}
```

- For API details, see [DeleteDirectConnectGateway](#) in *AWS CLI Command Reference*.

delete-interconnect

The following code example shows how to use `delete-interconnect`.

AWS CLI

To delete an interconnect

The following `delete-interconnect` command deletes the specified interconnect:

```
aws directconnect delete-interconnect --interconnect-id dxcon-fgktov66
```

Output:

```
{
  "interconnectState": "deleted"
}
```

- For API details, see [DeleteInterconnect](#) in *AWS CLI Command Reference*.

delete-lag

The following code example shows how to use `delete-lag`.

AWS CLI

To delete a LAG

The following example deletes the specified LAG.

Command:

```
aws directconnect delete-lag --lag-id dxlag-ffrhowd9
```

Output:

```
{
```

```
"awsDevice": "EqDC2-4h6ce2r1bes6",
"numberOfConnections": 0,
"lagState": "deleted",
"ownerAccount": "123456789012",
"lagName": "TestLAG",
"connections": [],
"lagId": "dxlag-ffrhowd9",
"minimumLinks": 0,
"connectionsBandwidth": "1Gbps",
"region": "us-east-1",
"location": "EqDC2"
}
```

- For API details, see [DeleteLag](#) in *AWS CLI Command Reference*.

delete-virtual-interface

The following code example shows how to use `delete-virtual-interface`.

AWS CLI

To delete a virtual interface

The following `delete-virtual-interface` command deletes the specified virtual interface:

```
aws directconnect delete-virtual-interface --virtual-interface-id dxvif-ffhkh74f
```

Output:

```
{
  "virtualInterfaceState": "deleting"
}
```

- For API details, see [DeleteVirtualInterface](#) in *AWS CLI Command Reference*.

describe-connection-loa

The following code example shows how to use `describe-connection-loa`.

AWS CLI

To describe your LOA-CFA for a connection using Linux or Mac OS X

The following example describes your LOA-CFA for connection `dxcon-fh6ayh1d`. The contents of the LOA-CFA are base64-encoded. This command uses the `--output` and `--query` parameters to control the output and extract the contents of the `loaContent` structure. The final part of the command decodes the content using the `base64` utility, and sends the output to a PDF file.

```
aws directconnect describe-connection-loa --connection-id dxcon-fh6ayh1d --output
text --query loa.loaContent|base64 --decode > myLoaCfa.pdf
```

To describe your LOA-CFA for a connection using Windows

The previous example requires the use of the `base64` utility to decode the output. On a Windows computer, you can use `certutil` instead. In the following example, the first command describes your LOA-CFA for connection `dxcon-fh6ayh1d` and uses the `--output` and `--query` parameters to control the output and extract the contents of the `loaContent` structure to a file called `myLoaCfa.base64`. The second command uses the `certutil` utility to decode the file and send the output to a PDF file.

```
aws directconnect describe-connection-loa --connection-id dxcon-fh6ayh1d --output
text --query loa.loaContent > myLoaCfa.base64
```

```
certutil -decode myLoaCfa.base64 myLoaCfa.pdf
```

For more information about controlling AWS CLI output, see [Controlling Command Output from the AWS Command Line Interface](#) in the *AWS Command Line Interface User Guide*.

- For API details, see [DescribeConnectionLoa](#) in *AWS CLI Command Reference*.

describe-connections-on-interconnect

The following code example shows how to use `describe-connections-on-interconnect`.

AWS CLI

To list connections on an interconnect

The following `describe-connections-on-interconnect` command lists connections that have been provisioned on the given interconnect:

```
aws directconnect describe-connections-on-interconnect --interconnect-id dxcon-
fgktov66
```

Output:

```
{
  "connections": [
    {
      "partnerName": "TIVIT",
      "vlan": 101,
      "ownerAccount": "123456789012",
      "connectionId": "dxcon-ffzc51m1",
      "connectionState": "ordering",
      "bandwidth": "500Mbps",
      "location": "TIVIT",
      "connectionName": "mydcinterconnect",
      "region": "sa-east-1"
    }
  ]
}
```

- For API details, see [DescribeConnectionsOnInterconnect](#) in *AWS CLI Command Reference*.

describe-connections

The following code example shows how to use describe-connections.

AWS CLI

To list all connections in the current region

The following describe-connections command lists all connections in the current region:

```
aws directconnect describe-connections
```

Output:

```
{
  "connections": [
    {
      "awsDevice": "EqDC2-123h49s71dabc",
      "ownerAccount": "123456789012",

```

```
    "connectionId": "dxcon-fguhmq1c",
    "lagId": "dxlag-ffrz71kw",
    "connectionState": "down",
    "bandwidth": "1Gbps",
    "location": "EqDC2",
    "connectionName": "My_Connection",
    "loaIssueTime": 1491568964.0,
    "region": "us-east-1"
  }
]
```

- For API details, see [DescribeConnections](#) in *AWS CLI Command Reference*.

describe-direct-connect-gateway-association-proposals

The following code example shows how to use `describe-direct-connect-gateway-association-proposals`.

AWS CLI

To describe your Direct Connect gateway association proposals

The following `describe-direct-connect-gateway-association-proposals` example displays details about your Direct Connect gateway association proposals.

```
aws directconnect describe-direct-connect-gateway-association-proposals
```

Output:

```
{
  "directConnectGatewayAssociationProposals": [
    {
      "proposalId": "c2ede9b4-bbc6-4d33-923c-bc4feEXAMPLE",
      "directConnectGatewayId": "11460968-4ac1-4fd3-bdb2-00599EXAMPLE",
      "directConnectGatewayOwnerAccount": "111122223333",
      "proposalState": "requested",
      "associatedGateway": {
        "id": "tgw-02f776b1a7EXAMPLE",
        "type": "transitGateway",
        "ownerAccount": "111122223333",
        "region": "us-east-1"
      }
    }
  ]
}
```

```
    },
    "existingAllowedPrefixesToDirectConnectGateway": [
      {
        "cidr": "192.168.2.0/30"
      },
      {
        "cidr": "192.168.1.0/30"
      }
    ],
    "requestedAllowedPrefixesToDirectConnectGateway": [
      {
        "cidr": "192.168.1.0/30"
      }
    ]
  },
  {
    "proposalId": "cb7f41cb-8128-43a5-93b1-dcaedEXAMPLE",
    "directConnectGatewayId": "11560968-4ac1-4fd3-bcb2-00599EXAMPLE",
    "directConnectGatewayOwnerAccount": "111122223333",
    "proposalState": "accepted",
    "associatedGateway": {
      "id": "tgw-045776b1a7EXAMPLE",
      "type": "transitGateway",
      "ownerAccount": "111122223333",
      "region": "us-east-1"
    },
    "existingAllowedPrefixesToDirectConnectGateway": [
      {
        "cidr": "192.168.4.0/30"
      },
      {
        "cidr": "192.168.5.0/30"
      }
    ],
    "requestedAllowedPrefixesToDirectConnectGateway": [
      {
        "cidr": "192.168.5.0/30"
      }
    ]
  }
]
```



```

    }
  ]
}

```

- For API details, see [DescribeDirectConnectGatewayAssociations](#) in *AWS CLI Command Reference*.

describe-direct-connect-gateway-attachments

The following code example shows how to use `describe-direct-connect-gateway-attachments`.

AWS CLI

To describe Direct Connect gateway attachments

The following example describes the virtual interfaces that are attached to Direct Connect gateway `5f294f92-bafb-4011-916d-9b0bexample`.

Command:

```
aws directconnect describe-direct-connect-gateway-attachments --direct-connect-gateway-id 5f294f92-bafb-4011-916d-9b0bexample
```

Output:

```
{
  "directConnectGatewayAttachments": [
    {
      "virtualInterfaceOwnerAccount": "123456789012",
      "directConnectGatewayId": "5f294f92-bafb-4011-916d-9b0bexample",
      "virtualInterfaceRegion": "us-east-2",
      "attachmentState": "attaching",
      "virtualInterfaceId": "dxvif-fg9zyabc"
    }
  ],
  "nextToken":
  "eyJ2IjoxLCJzIjoxLCJpIjoibEhXd1NpUXF5RzhoL1JyUW52S1V2QT09IiwieYyI6Im5wQjFHQ0RyQUdRS3puNnNXcU"
}
```

- For API details, see [DescribeDirectConnectGatewayAttachments](#) in *AWS CLI Command Reference*.

describe-direct-connect-gateways

The following code example shows how to use `describe-direct-connect-gateways`.

AWS CLI

To describe your Direct Connect gateways

The following example describe all of your Direct Connect gateways.

Command:

```
aws directconnect describe-direct-connect-gateways
```

Output:

```
{
  "directConnectGateways": [
    {
      "amazonSideAsn": 64512,
      "directConnectGatewayId": "cf68415c-f4ae-48f2-87a7-3b52cexample",
      "ownerAccount": "123456789012",
      "directConnectGatewayName": "DxGateway2",
      "directConnectGatewayState": "available"
    },
    {
      "amazonSideAsn": 64512,
      "directConnectGatewayId": "5f294f92-bafb-4011-916d-9b0bdexample",
      "ownerAccount": "123456789012",
      "directConnectGatewayName": "DxGateway1",
      "directConnectGatewayState": "available"
    }
  ]
}
```

- For API details, see [DescribeDirectConnectGateways](#) in *AWS CLI Command Reference*.

describe-hosted-connections

The following code example shows how to use `describe-hosted-connections`.

AWS CLI

To list connections on an interconnect

The following example lists connections that have been provisioned on the given interconnect.

Command:

```
aws directconnect describe-hosted-connections --connection-id dxcon-fgktov66
```

Output:

```
{
  "connections": [
    {
      "partnerName": "TIVIT",
      "vlan": 101,
      "ownerAccount": "123456789012",
      "connectionId": "dxcon-ffzc51m1",
      "connectionState": "ordering",
      "bandwidth": "500Mbps",
      "location": "TIVIT",
      "connectionName": "mydcinterconnect",
      "region": "sa-east-1"
    }
  ]
}
```

- For API details, see [DescribeHostedConnections](#) in *AWS CLI Command Reference*.

describe-interconnect-loa

The following code example shows how to use `describe-interconnect-loa`.

AWS CLI

To describe your LOA-CFA for an interconnect using Linux or Mac OS X

The following example describes your LOA-CFA for interconnect `dxcon-fh6ayh1d`. The contents of the LOA-CFA are base64-encoded. This command uses the `--output` and `--query` parameters to control the output and extract the contents of the `loaContent` structure. The

final part of the command decodes the content using the base64 utility, and sends the output to a PDF file.

```
aws directconnect describe-interconnect-loa --interconnect-id dxcon-fh6ayh1d --output text --query loa.loaContent|base64 --decode > myLoaCfa.pdf
```

To describe your LOA-CFA for an interconnect using Windows

The previous example requires the use of the base64 utility to decode the output. On a Windows computer, you can use `certutil` instead. In the following example, the first command describes your LOA-CFA for interconnect `dxcon-fh6ayh1d` and uses the `--output` and `--query` parameters to control the output and extract the contents of the `loaContent` structure to a file called `myLoaCfa.base64`. The second command uses the `certutil` utility to decode the file and send the output to a PDF file.

```
aws directconnect describe-interconnect-loa --interconnect-id dxcon-fh6ayh1d --output text --query loa.loaContent > myLoaCfa.base64
```

```
certutil -decode myLoaCfa.base64 myLoaCfa.pdf
```

For more information about controlling AWS CLI output, see [Controlling Command Output from the AWS Command Line Interface](#) in the *AWS Command Line Interface User Guide*.

- For API details, see [DescribeInterconnectLoa](#) in *AWS CLI Command Reference*.

describe-interconnects

The following code example shows how to use `describe-interconnects`.

AWS CLI

To list interconnects

The following `describe-interconnects` command lists the interconnects owned by your AWS account:

```
aws directconnect describe-interconnects
```

Output:

```
{
  "interconnects": [
    {
      "region": "sa-east-1",
      "bandwidth": "1Gbps",
      "location": "TIVIT",
      "interconnectName": "1G Interconnect to AWS",
      "interconnectId": "dxcon-fgktov66",
      "interconnectState": "down"
    }
  ]
}
```

- For API details, see [DescribeInterconnects](#) in *AWS CLI Command Reference*.

describe-lags

The following code example shows how to use `describe-lags`.

AWS CLI

To describe your LAGs

The following command describes all of your LAGs for the current region.

Command:

```
aws directconnect describe-lags
```

Output:

```
{
  "lags": [
    {
      "awsDevice": "EqDC2-19y7z3m17xpuz",
      "numberOfConnections": 2,
      "lagState": "down",
      "ownerAccount": "123456789012",
      "lagName": "DA-LAG",
      "connections": [
        {
```

```

        "ownerAccount": "123456789012",
        "connectionId": "dxcon-ffnikghc",
        "lagId": "dxlag-fgsu9erb",
        "connectionState": "requested",
        "bandwidth": "10Gbps",
        "location": "EqDC2",
        "connectionName": "Requested Connection 1 for Lag dxlag-fgsu9erb",
        "region": "us-east-1"
    },
    {
        "ownerAccount": "123456789012",
        "connectionId": "dxcon-fglgbdea",
        "lagId": "dxlag-fgsu9erb",
        "connectionState": "requested",
        "bandwidth": "10Gbps",
        "location": "EqDC2",
        "connectionName": "Requested Connection 2 for Lag dxlag-fgsu9erb",
        "region": "us-east-1"
    }
],
"lagId": "dxlag-fgsu9erb",
"minimumLinks": 0,
"connectionsBandwidth": "10Gbps",
"region": "us-east-1",
"location": "EqDC2"
}
]
}

```

- For API details, see [DescribeLags](#) in *AWS CLI Command Reference*.

describe-loa

The following code example shows how to use `describe-loa`.

AWS CLI

To describe your LOA-CFA for a connection using Linux or Mac OS X

The following example describes your LOA-CFA for connection `dxcon-fh6ayh1d`. The contents of the LOA-CFA are base64-encoded. This command uses the `--output` and `--query` parameters to control the output and extract the contents of the `loaContent` structure. The

final part of the command decodes the content using the base64 utility, and sends the output to a PDF file.

```
aws directconnect describe-loa --connection-id dxcon-fh6ayh1d --output text --query
loa.loaContent|base64 --decode > myLoaCfa.pdf
```

To describe your LOA-CFA for a connection using Windows

The previous example requires the use of the base64 utility to decode the output. On a Windows computer, you can use `certutil` instead. In the following example, the first command describes your LOA-CFA for connection `dxcon-fh6ayh1d` and uses the `--output` and `--query` parameters to control the output and extract the contents of the `loaContent` structure to a file called `myLoaCfa.base64`. The second command uses the `certutil` utility to decode the file and send the output to a PDF file.

```
aws directconnect describe-loa --connection-id dxcon-fh6ayh1d --output text --query
loa.loaContent > myLoaCfa.base64
```

```
certutil -decode myLoaCfa.base64 myLoaCfa.pdf
```

For more information about controlling AWS CLI output, see [Controlling Command Output from the AWS Command Line Interface](#) in the *AWS Command Line Interface User Guide*.

- For API details, see [DescribeLoa](#) in *AWS CLI Command Reference*.

describe-locations

The following code example shows how to use `describe-locations`.

AWS CLI

To list AWS Direct Connect partners and locations

The following `describe-locations` command lists AWS Direct Connect partners and locations in the current region:

```
aws directconnect describe-locations
```

Output:


```
{
  "locations": [
    {
      "locationName": "NAP do Brasil, Barueri, Sao Paulo",
      "locationCode": "TNDB"
    },
    {
      "locationName": "Tivit - Site Transamerica (Sao Paulo)",
      "locationCode": "TIVIT"
    }
  ]
}
```

- For API details, see [DescribeLocations](#) in *AWS CLI Command Reference*.

describe-tags

The following code example shows how to use `describe-tags`.

AWS CLI

To describe tags for your AWS Direct Connect resources

The following command describes the tags for the connection `dxcon-abcabc12`.

Command:

```
aws directconnect describe-tags --resource-arns arn:aws:directconnect:us-east-1:123456789012:dxcon/dxcon-abcabc12
```

Output:

```
{
  "resourceTags": [
    {
      "resourceArn": "arn:aws:directconnect:us-east-1:123456789012:dxcon/dxcon-abcabc12",
      "tags": [
        {
          "value": "VAConnection",
          "key": "Name"
        }
      ]
    }
  ]
}
```

```
    ]
  }
]
}
```

- For API details, see [DescribeTags](#) in *AWS CLI Command Reference*.

describe-virtual-gateways

The following code example shows how to use `describe-virtual-gateways`.

AWS CLI

To list virtual private gateways

The following `describe-virtual-gateways` command lists virtual private gateways owned by your AWS account:

```
aws directconnect describe-virtual-gateways
```

Output:

```
{
  "virtualGateways": [
    {
      "virtualGatewayId": "vgw-aba37db6",
      "virtualGatewayState": "available"
    }
  ]
}
```

- For API details, see [DescribeVirtualGateways](#) in *AWS CLI Command Reference*.

describe-virtual-interfaces

The following code example shows how to use `describe-virtual-interfaces`.

AWS CLI

To list all virtual interfaces

The following `describe-virtual-interfaces` command lists the information about all virtual interfaces associated with your AWS account:

```
aws directconnect describe-virtual-interfaces --connection-id dxcon-ffjrkx17
```

Output:

```
{
  "virtualInterfaces": [
    {
      "virtualInterfaceState": "down",
      "asn": 65000,
      "vlan": 101,
      "customerAddress": "192.168.1.2/30",
      "ownerAccount": "123456789012",
      "connectionId": "dxcon-ffjrkx17",
      "virtualGatewayId": "vgw-aba37db6",
      "virtualInterfaceId": "dxvif-ffhkh74f",
      "authKey": "asdf34example",
      "routeFilterPrefixes": [],
      "location": "TIVIT",
      "customerRouterConfig": "<?xml version='1.0' encoding='UTF-8'?'>\n<logical_connection id='dxvif-ffhkh74f'>\n  <vlan>101</vlan>\n  <customer_address>192.168.1.2/30</customer_address>\n  <amazon_address>192.168.1.1/30</amazon_address>\n  <bgp_asn>65000</bgp_asn>\n  <bgp_auth_key>asdf34example</bgp_auth_key>\n  <amazon_bgp_asn>7224</amazon_bgp_asn>\n  <connection_type>private</connection_type>\n</logical_connection>\n",
      "amazonAddress": "192.168.1.1/30",
      "virtualInterfaceType": "private",
      "virtualInterfaceName": "PrivateVirtualInterface"
    },
    {
      "virtualInterfaceState": "verifying",
      "asn": 65000,
      "vlan": 2000,
      "customerAddress": "203.0.113.2/30",
      "ownerAccount": "123456789012",
      "connectionId": "dxcon-ffjrkx17",
      "virtualGatewayId": "",
      "virtualInterfaceId": "dxvif-fgh0hcrk",
      "authKey": "asdf34example",
      "routeFilterPrefixes": [
        {

```

```

        "cidr": "203.0.113.4/30"
      },
      {
        "cidr": "203.0.113.0/30"
      }
    ],
    "location": "TIVIT",
    "customerRouterConfig": "<?xml version=\"1.0\" encoding=
\"UTF-8\"?>\n<logical_connection id=\"dxvif-fgh0hcrk\">\n  <vlan>2000</
vlan>\n  <customer_address>203.0.113.2/30</customer_address>\n
  <amazon_address>203.0.113.1/30</amazon_address>\n  <bgp_asn>65000</bgp_asn>\n
  <bgp_auth_key>asdf34example</bgp_auth_key>\n  <amazon_bgp_asn>7224</amazon_bgp_asn>
\n  <connection_type>public</connection_type>\n</logical_connection>\n",
    "amazonAddress": "203.0.113.1/30",
    "virtualInterfaceType": "public",
    "virtualInterfaceName": "PublicVirtualInterface"
  }
]
}

```

- For API details, see [DescribeVirtualInterfaces](#) in *AWS CLI Command Reference*.

disassociate-connection-from-lag

The following code example shows how to use `disassociate-connection-from-lag`.

AWS CLI

To disassociate a connection from a LAG

The following example disassociates the specified connection from the specified LAG.

Command:

```
aws directconnect disassociate-connection-from-lag --lag-id dxlag-fhccu14t --
connection-id dxcon-fg9607vm
```

Output:

```
{
  "ownerAccount": "123456789012",
```

```
"connectionId": "dxcon-fg9607vm",
"connectionState": "requested",
"bandwidth": "1Gbps",
"location": "EqDC2",
"connectionName": "Con2ForLag",
"region": "us-east-1"
}
```

- For API details, see [DisassociateConnectionFromLag](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add a tag to an AWS Direct Connect resource

The following command adds a tag with a key of `Name` and a value of `VACconnection` to the connection `dxcon-abcabc12`. If the command succeeds, no output is returned.

Command:

```
aws directconnect tag-resource --resource-arn arn:aws:directconnect:us-east-1:123456789012:dxcon/dxcon-abcabc12 --tags "key=Name,value=VACconnection"
```

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag from an AWS Direct Connect resource

The following command removes the tag with the key `Name` from connection `dxcon-abcabc12`. If the command succeeds, no output is returned.

Command:

```
aws directconnect untag-resource --resource-arn arn:aws:directconnect:us-east-1:123456789012:dxcon/dxcon-abcabc12 --tag-keys Name
```

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-direct-connect-gateway-association

The following code example shows how to use `update-direct-connect-gateway-association`.

AWS CLI

To update the specified attributes of the Direct Connect gateway association

The following `update-direct-connect-gateway-association` example adds the specified CIDR block to a Direct Connect gateway association.

```
aws directconnect update-direct-connect-gateway-association \
  --association-id 820a6e4f-5374-4004-8317-3f64bEXAMPLE \
  --add-allowed-prefixes-to-direct-connect-gateway cidr=192.168.2.0/30
```

Output:

```
{
  "directConnectGatewayAssociation": {
    "directConnectGatewayId": "11460968-4ac1-4fd3-bdb2-00599EXAMPLE",
    "directConnectGatewayOwnerAccount": "111122223333",
    "associationState": "updating",
    "associatedGateway": {
      "id": "tgw-02f776b1a7EXAMPLE",
      "type": "transitGateway",
      "ownerAccount": "111122223333",
      "region": "us-east-1"
    },
    "associationId": "820a6e4f-5374-4004-8317-3f64bEXAMPLE",
    "allowedPrefixesToDirectConnectGateway": [
      {
        "cidr": "192.168.2.0/30"
      },
      {
        "cidr": "192.168.1.0/30"
      }
    ]
  }
}
```

```

    }
  ]
}

```

For more information, see [Working with Direct Connect Gateways](#) in the *AWS Direct Connect User Guide*.

- For API details, see [UpdateDirectConnectGatewayAssociation](#) in *AWS CLI Command Reference*.

update-lag

The following code example shows how to use `update-lag`.

AWS CLI

To update a LAG

The following example changes the name of the specified LAG.

Command:

```
aws directconnect update-lag --lag-id dxlag-ffjhj9lx --lag-name 2ConnLag
```

Output:

```
{
  "awsDevice": "CSVA1-23u8t1paz8iks",
  "numberOfConnections": 2,
  "lagState": "down",
  "ownerAccount": "123456789012",
  "lagName": "2ConnLag",
  "connections": [
    {
      "ownerAccount": "123456789012",
      "connectionId": "dxcon-fflqyj95",
      "lagId": "dxlag-ffjhj9lx",
      "connectionState": "requested",
      "bandwidth": "1Gbps",
      "location": "CSVA1",
      "connectionName": "Requested Connection 2 for Lag dxlag-ffjhj9lx",
      "region": "us-east-1"
    }
  ],
}
```

```
{
  "ownerAccount": "123456789012",
  "connectionId": "dxcon-ffqr6x5q",
  "lagId": "dxlag-ffjhj9lx",
  "connectionState": "requested",
  "bandwidth": "1Gbps",
  "location": "CSVA1",
  "connectionName": "Requested Connection 1 for Lag dxlag-ffjhj9lx",
  "region": "us-east-1"
},
"lagId": "dxlag-ffjhj9lx",
"minimumLinks": 0,
"connectionsBandwidth": "1Gbps",
"region": "us-east-1",
"location": "CSVA1"
}
```

- For API details, see [UpdateLag](#) in *AWS CLI Command Reference*.

update-virtual-interface-attributes

The following code example shows how to use `update-virtual-interface-attributes`.

AWS CLI

To update the MTU of a virtual interface

The following `update-virtual-interface-attributes` example updates the MTU of the specified virtual interface.

```
aws directconnect update-virtual-interface-attributes \
  --virtual-interface-id dxvif-fEXAMPLE \
  --mtu 1500
```

Output:

```
{
  "ownerAccount": "1111222233333",
  "virtualInterfaceId": "dxvif-fEXAMPLE",
  "location": "loc1",
  "connectionId": "dxlag-fEXAMPLE",
```



```

"virtualInterfaceType": "transit",
"virtualInterfaceName": "example transit virtual interface",
"vlan": 125,
"asn": 650001,
"amazonSideAsn": 64512,
"authKey": "0xzxgA9YoW9h58u8SEXAMPLE",
"amazonAddress": "169.254.248.1/30",
"customerAddress": "169.254.248.2/30",
"addressFamily": "ipv4",
"virtualInterfaceState": "down",
"customerRouterConfig": "<?xml version=\"1.0\" encoding=\"UTF-8\"?
>\n<logical_connection id=\"dxvif-fEXAMPLE\">\n  <vlan>125</vlan>
\n  <customer_address>169.254.248.2/30</customer_address>\n
  <amazon_address>169.254.248.1/30</amazon_address>\n  <bgp_asn>650001</bgp_asn>\n
  <bgp_auth_key>0xzxgA9YoW9h58u8SEXAMPLE</bgp_auth_key>\n  <amazon_bgp_asn>64512</
amazon_bgp_asn>\n  <connection_type>transit</connection_type>\n</logical_connection>
\n",
  "mtu": 1500,
  "jumboFrameCapable": true,
  "virtualGatewayId": "",
  "directConnectGatewayId": "879b76a1-403d-4700-8b53-4a56ed85436e",
  "routeFilterPrefixes": [],
  "bgpPeers": [
    {
      "bgpPeerId": "dxpeer-fEXAMPLE",
      "asn": 650001,
      "authKey": "0xzxgA9YoW9h58u8SEXAMPLE",
      "addressFamily": "ipv4",
      "amazonAddress": "169.254.248.1/30",
      "customerAddress": "169.254.248.2/30",
      "bgpPeerState": "available",
      "bgpStatus": "down",
      "awsDeviceV2": "loc1-26wz6vEXAMPLE"
    }
  ],
  "region": "sa-east-1",
  "awsDeviceV2": "loc1-26wz6vEXAMPLE",
  "tags": []
}

```

For more information, see [Setting Network MTU for Private Virtual Interfaces or Transit Virtual Interfaces](#) in the *AWS Direct Connect User Guide*.

- For API details, see [UpdateVirtualInterfaceAttributes](#) in *AWS CLI Command Reference*.

AWS Directory Service examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Directory Service.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

describe-directories

The following code example shows how to use `describe-directories`.

AWS CLI

To get details about your directories

The following `describe-directories` example displays details about the specified directory.

```
aws ds describe-directories \  
  --directory-id d-a1b2c3d4e5
```

Output:

```
{  
  "DirectoryDescriptions": [  
    {  
      "DirectoryId": "d-a1b2c3d4e5",
```

```

    "Name": "mydirectory.example.com",
    "ShortName": "mydirectory",
    "Size": "Small",
    "Edition": "Standard",
    "Alias": "d-a1b2c3d4e5",
    "AccessUrl": "d-a1b2c3d4e5.awsapps.com",
    "Stage": "Active",
    "ShareStatus": "Shared",
    "ShareMethod": "HANDSHAKE",
    "ShareNotes": "These are my share notes",
    "LaunchTime": "2019-07-08T15:33:46.327000-07:00",
    "StageLastUpdatedDateTime": "2019-07-08T15:59:12.307000-07:00",
    "Type": "SharedMicrosoftAD",
    "SsoEnabled": false,
    "DesiredNumberOfDomainControllers": 0,
    "OwnerDirectoryDescription": {
      "DirectoryId": "d-b2c3d4e5f6",
      "AccountId": "123456789111",
      "DnsIpAddrs": [
        "203.113.0.248",
        "203.113.0.253"
      ],
      "VpcSettings": {
        "VpcId": "vpc-a1b2c3d4",
        "SubnetIds": [
          "subnet-a1b2c3d4",
          "subnet-d4c3b2a1"
        ],
        "AvailabilityZones": [
          "us-west-2a",
          "us-west-2c"
        ]
      }
    }
  }
}
]
}

```

- For API details, see [DescribeDirectories](#) in *AWS CLI Command Reference*.

describe-trusts

The following code example shows how to use describe-trusts.

AWS CLI

To get details about your trust relationships

The following `describe-trusts` example displays details about the trust relationships for the specified directory.

```
aws ds describe-trusts \  
  --directory-id d-a1b2c3d4e5
```

Output:

```
{  
  "Trusts": [  
    {  
      "DirectoryId": "d-a1b2c3d4e5",  
      "TrustId": "t-9a8b7c6d5e",  
      "RemoteDomainName": "other.example.com",  
      "TrustType": "Forest",  
      "TrustDirection": "Two-Way",  
      "TrustState": "Verified",  
      "CreatedDateTime": "2017-06-20T18:08:45.614000-07:00",  
      "LastUpdatedDateTime": "2019-06-04T10:52:12.410000-07:00",  
      "StateLastUpdatedDateTime": "2019-06-04T10:52:12.410000-07:00",  
      "SelectiveAuth": "Disabled"  
    }  
  ]  
}
```

- For API details, see [DescribeTrusts](#) in *AWS CLI Command Reference*.

AWS DMS examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS DMS.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.


```
--endpoint-identifier src-endpoint \
--s3-settings file://s3-settings.json
```

Contents of s3-settings.json:

```
{
  "BucketName": "my-corp-data",
  "BucketFolder": "sourcedata",
  "ServiceAccessRoleArn": "arn:aws:iam::123456789012:role/my-s3-access-role"
}
```

Output:

```
{
  "Endpoint": {
    "EndpointIdentifier": "src-endpoint",
    "EndpointType": "SOURCE",
    "EngineName": "s3",
    "EngineDisplayName": "Amazon S3",
    "ExtraConnectionAttributes": "bucketFolder=sourcedata;bucketName=my-corp-
data;compressionType=NONE;csvDelimiter=,;csvRowDelimiter=\n;",
    "Status": "active",
    "EndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:GUVAFG34EECU0J6QVZ56DAHT3U",
    "SslMode": "none",
    "ServiceAccessRoleArn": "arn:aws:iam::123456789012:role/my-s3-access-role",
    "S3Settings": {
      "ServiceAccessRoleArn": "arn:aws:iam::123456789012:role/my-s3-access-
role",
      "CsvRowDelimiter": "\n",
      "CsvDelimiter": ",",
      "BucketFolder": "sourcedata",
      "BucketName": "my-corp-data",
      "CompressionType": "NONE",
      "EnableStatistics": true
    }
  }
}
```

For more information, see [Working with AWS DMS Endpoints](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [CreateEndpoint](#) in *AWS CLI Command Reference*.

create-event-subscription

The following code example shows how to use `create-event-subscription`.

AWS CLI

To list event subscriptions

The following `create-event-subscription` example creates an event subscription to an Amazon SNS topic (`my-sns-topic`).

```
aws dms create-event-subscription \  
  --subscription-name my-dms-events \  
  --sns-topic-arn arn:aws:sns:us-east-1:123456789012:my-sns-topic
```

Output:

```
{  
  "EventSubscription": {  
    "CustomerAwsId": "123456789012",  
    "CustSubscriptionId": "my-dms-events",  
    "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:my-sns-topic",  
    "Status": "creating",  
    "SubscriptionCreationTime": "2020-05-21 21:58:38.598",  
    "Enabled": true  
  }  
}
```

For more information, see [Working with Events and Notifications](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [CreateEventSubscription](#) in *AWS CLI Command Reference*.

create-replication-instance

The following code example shows how to use `create-replication-instance`.

AWS CLI

To create a replication instance

The following `create-replication-instance` example creates a replication instance.

```
aws dms create-replication-instance \  
  --replication-instance-identifier my-repl-instance \  
  --replication-instance-class dms.t2.micro \  
  --allocated-storage 5
```

Output:

```
{  
  "ReplicationInstance": {  
    "ReplicationInstanceIdentifier": "my-repl-instance",  
    "ReplicationInstanceClass": "dms.t2.micro",  
    "ReplicationInstanceStatus": "creating",  
    "AllocatedStorage": 5,  
    "VpcSecurityGroups": [  
      {  
        "VpcSecurityGroupId": "sg-f839b688",  
        "Status": "active"  
      }  
    ],  
    "ReplicationSubnetGroup": {  
      "ReplicationSubnetGroupIdentifier": "default",  
      "ReplicationSubnetGroupDescription": "default",  
      "VpcId": "vpc-136a4c6a",  
      "SubnetGroupStatus": "Complete",  
      "Subnets": [  
        {  
          "SubnetIdentifier": "subnet-da327bf6",  
          "SubnetAvailabilityZone": {  
            "Name": "us-east-1a"  
          },  
          "SubnetStatus": "Active"  
        },  
        {  
          "SubnetIdentifier": "subnet-42599426",  
          "SubnetAvailabilityZone": {  
            "Name": "us-east-1d"  
          },  
          "SubnetStatus": "Active"  
        },  
        {  
          "SubnetIdentifier": "subnet-bac383e0",  
          "SubnetAvailabilityZone": {  
            "Name": "us-east-1c"  
          }  
        }  
      ]  
    }  
  }  
}
```



```

        },
        "SubnetStatus": "Active"
    },
    {
        "SubnetIdentifier": "subnet-6746046b",
        "SubnetAvailabilityZone": {
            "Name": "us-east-1f"
        },
        "SubnetStatus": "Active"
    },
    {
        "SubnetIdentifier": "subnet-d7c825e8",
        "SubnetAvailabilityZone": {
            "Name": "us-east-1e"
        },
        "SubnetStatus": "Active"
    },
    {
        "SubnetIdentifier": "subnet-cbfff283",
        "SubnetAvailabilityZone": {
            "Name": "us-east-1b"
        },
        "SubnetStatus": "Active"
    }
]
},
"PreferredMaintenanceWindow": "sat:12:35-sat:13:05",
"PendingModifiedValues": {},
"MultiAZ": false,
"EngineVersion": "3.3.2",
"AutoMinorVersionUpgrade": true,
"KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/f7bc0f8e-1a3a-4ace-9faa-
e8494fa3921a",
"ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:ZK2VQBUWFDBAWHIXHAYG5G2PKY",
"PubliclyAccessible": true
}
}

```

For more information, see [Working with an AWS DMS Replication Instance](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [CreateReplicationInstance](#) in *AWS CLI Command Reference*.

create-replication-subnet-group

The following code example shows how to use create-replication-subnet-group.

AWS CLI

To create a subnet group

The following create-replication-subnet-group example creates a group consisting of 3 subnets.

```
aws dms create-replication-subnet-group \  
  --replication-subnet-group-identifier my-subnet-group \  
  --replication-subnet-group-description "my subnet group" \  
  --subnet-ids subnet-da327bf6 subnet-bac383e0 subnet-d7c825e8
```

Output:

```
{  
  "ReplicationSubnetGroup": {  
    "ReplicationSubnetGroupIdentifier": "my-subnet-group",  
    "ReplicationSubnetGroupDescription": "my subnet group",  
    "VpcId": "vpc-136a4c6a",  
    "SubnetGroupStatus": "Complete",  
    "Subnets": [  
      {  
        "SubnetIdentifier": "subnet-da327bf6",  
        "SubnetAvailabilityZone": {  
          "Name": "us-east-1a"  
        },  
        "SubnetStatus": "Active"  
      },  
      {  
        "SubnetIdentifier": "subnet-bac383e0",  
        "SubnetAvailabilityZone": {  
          "Name": "us-east-1c"  
        },  
        "SubnetStatus": "Active"  
      },  
      {  
        "SubnetIdentifier": "subnet-d7c825e8",  
        "SubnetAvailabilityZone": {  
          "Name": "us-east-1e"  
        }  
      }  
    ]  
  }  
}
```

```

        },
        "SubnetStatus": "Active"
    }
]
}
}

```

For more information, see [Setting Up a Network for a Replication Instance](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [CreateReplicationSubnetGroup](#) in *AWS CLI Command Reference*.

create-replication-task

The following code example shows how to use `create-replication-task`.

AWS CLI

To create a replication task

The following `create-replication-task` example creates a replication task.

```

aws dms create-replication-task \
  --replication-task-identifier movedata \
  --source-endpoint-arn arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWGWAYUVLKIB732KEVWA \
  --target-endpoint-arn arn:aws:dms:us-
east-1:123456789012:endpoint:E0M4SFKCZEYHZBFGAGZT3QEC5U \
  --replication-instance-arn $RI_ARN \
  --migration-type full-load \
  --table-mappings file://table-mappings.json

```

Contents of `table-mappings.json`:

```

{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "prodrep",
        "table-name": "%"
      }
    }
  ]
}

```

```

        },
        "rule-action": "include",
        "filters": []
    }
]
}

```

Output:

```

{
  "ReplicationTask": {
    "ReplicationTaskIdentifier": "moveit2",
    "SourceEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWGWAYUVLKIB732KEVWA",
    "TargetEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:E0M4SFKCZEYHZBFGAGZT3QEC5U",
    "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE",
    "MigrationType": "full-load",
    "TableMappings": ...output omitted... ,
    "ReplicationTaskSettings": ...output omitted... ,
    "Status": "creating",
    "ReplicationTaskCreationDate": 1590524772.505,
    "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII"
  }
}

```

For more information, see [Working with AWS DMS Tasks](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [CreateReplicationTask](#) in *AWS CLI Command Reference*.

delete-connection

The following code example shows how to use `delete-connection`.

AWS CLI**To delete a connection**

The following `delete-connection` example disassociates an endpoint from a replication instance.

```
aws dms delete-connection \  
  --endpoint-arn arn:aws:dms:us-  
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA \  
  --replication-instance-arn arn:aws:dms:us-  
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE
```

Output:

```
{  
  "Connection": {  
    "ReplicationInstanceArn": "arn:aws:dms:us-  
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE",  
    "EndpointArn": "arn:aws:dms:us-  
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA",  
    "Status": "deleting",  
    "EndpointIdentifier": "src-database-1",  
    "ReplicationInstanceIdentifier": "my-repl-instance"  
  }  
}
```

For more information, see https://docs.aws.amazon.com/dms/latest/userguide/CHAP_Endpoints.Creating.html in the *AWS Database Migration Service User Guide*.

- For API details, see [DeleteConnection](#) in *AWS CLI Command Reference*.

delete-endpoint

The following code example shows how to use delete-endpoint.

AWS CLI

To delete an endpoint

The following delete-endpoint example deletes an endpoint.

```
aws dms delete-endpoint \  
  --endpoint-arn arn:aws:dms:us-  
east-1:123456789012:endpoint:0UJJVX04XZ4CYTSEG5XGMN2R3Y
```

Output:

```
{
```

```

    "Endpoint": {
      "EndpointIdentifier": "src-endpoint",
      "EndpointType": "SOURCE",
      "EngineName": "s3",
      "EngineDisplayName": "Amazon S3",
      "ExtraConnectionAttributes": "bucketFolder=sourcedata;bucketName=my-corp-
data;compressionType=NONE;csvDelimiter=,;csvRowDelimiter=\\n;",
      "Status": "deleting",
      "EndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:0UJJVX04XZ4CYTSEG5XGMN2R3Y",
      "SslMode": "none",
      "ServiceAccessRoleArn": "arn:aws:iam::123456789012:role/my-s3-access-role",
      "S3Settings": {
        "ServiceAccessRoleArn": "arn:aws:iam::123456789012:role/my-s3-access-
role",
        "CsvRowDelimiter": "\\n",
        "CsvDelimiter": ",",
        "BucketFolder": "sourcedata",
        "BucketName": "my-corp-data",
        "CompressionType": "NONE",
        "EnableStatistics": true
      }
    }
  }
}

```

For more information, see [Working with AWS DMS Endpoints](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [DeleteEndpoint](#) in *AWS CLI Command Reference*.

delete-event-subscription

The following code example shows how to use delete-event-subscription.

AWS CLI

To delete an event subscription

The following delete-event-subscription example deletes a subscription to an Amazon SNS topic.

```

aws dms delete-event-subscription \
  --subscription-name "my-dms-events"

```

Output:

```
{
  "EventSubscription": {
    "CustomerAwsId": "123456789012",
    "CustSubscriptionId": "my-dms-events",
    "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:my-sns-topic",
    "Status": "deleting",
    "SubscriptionCreationTime": "2020-05-21 21:58:38.598",
    "Enabled": true
  }
}
```

For more information, see [Working with Events and Notifications](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [DeleteEventSubscription](#) in *AWS CLI Command Reference*.

delete-replication-instance

The following code example shows how to use `delete-replication-instance`.

AWS CLI**To delete a replication instance**

The following `delete-replication-instance` example deletes a replication instance.

```
aws dms delete-replication-instance \
  --replication-instance-arn arn:aws:dms:us-
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE
```

Output:

```
{
  "ReplicationInstance": {
    "ReplicationInstanceIdentifier": "my-repl-instance",
    "ReplicationInstanceClass": "dms.t2.micro",
    "ReplicationInstanceStatus": "deleting",
    "AllocatedStorage": 5,
    "InstanceCreateTime": 1590011235.952,
    "VpcSecurityGroups": [
      {
```

```
        "VpcSecurityGroupId": "sg-f839b688",
        "Status": "active"
    }
],
"AvailabilityZone": "us-east-1e",
"ReplicationSubnetGroup": {
    "ReplicationSubnetGroupIdentifier": "default",
    "ReplicationSubnetGroupDescription": "default",
    "VpcId": "vpc-136a4c6a",
    "SubnetGroupStatus": "Complete",
    "Subnets": [
        {
            "SubnetIdentifier": "subnet-da327bf6",
            "SubnetAvailabilityZone": {
                "Name": "us-east-1a"
            },
            "SubnetStatus": "Active"
        },
        {
            "SubnetIdentifier": "subnet-42599426",
            "SubnetAvailabilityZone": {
                "Name": "us-east-1d"
            },
            "SubnetStatus": "Active"
        },
        {
            "SubnetIdentifier": "subnet-bac383e0",
            "SubnetAvailabilityZone": {
                "Name": "us-east-1c"
            },
            "SubnetStatus": "Active"
        },
        {
            "SubnetIdentifier": "subnet-6746046b",
            "SubnetAvailabilityZone": {
                "Name": "us-east-1f"
            },
            "SubnetStatus": "Active"
        },
        {
            "SubnetIdentifier": "subnet-d7c825e8",
            "SubnetAvailabilityZone": {
                "Name": "us-east-1e"
            },
        },
    ]
}
```



```

        "SubnetStatus": "Active"
    },
    {
        "SubnetIdentifier": "subnet-cbfff283",
        "SubnetAvailabilityZone": {
            "Name": "us-east-1b"
        },
        "SubnetStatus": "Active"
    }
]
},
"PreferredMaintenanceWindow": "wed:11:42-wed:12:12",
"PendingModifiedValues": {},
"MultiAZ": true,
"EngineVersion": "3.3.2",
"AutoMinorVersionUpgrade": true,
"KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/f7bc0f8e-1a3a-4ace-9faa-
e8494fa3921a",
"ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE",
"ReplicationInstancePublicIpAddress": "54.225.120.92",
"ReplicationInstancePrivateIpAddress": "172.31.30.121",
"ReplicationInstancePublicIpAddresses": [
    "54.225.120.92",
    "3.230.18.248"
],
"ReplicationInstancePrivateIpAddresses": [
    "172.31.30.121",
    "172.31.75.90"
],
"PubliclyAccessible": true,
"SecondaryAvailabilityZone": "us-east-1b"
}
}

```

For more information, see [Working with an AWS DMS Replication Instance](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [DeleteReplicationInstance](#) in *AWS CLI Command Reference*.

delete-replication-subnet-group

The following code example shows how to use delete-replication-subnet-group.

AWS CLI

To delete a subnet group

The following `delete-replication-subnet-group` example deletes a subnet group.

```
aws dms delete-replication-subnet-group \  
--replication-subnet-group-identifier my-subnet-group
```

Output:

```
(none)
```

For more information, see [Setting Up a Network for a Replication Instance](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [DeleteReplicationSubnetGroup](#) in *AWS CLI Command Reference*.

delete-replication-task

The following code example shows how to use `delete-replication-task`.

AWS CLI

To delete a replication task

The following `delete-replication-task` example deletes a replication task.

```
aws dms delete-replication-task \  
--replication-task-arn arn:aws:dms:us-  
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII
```

Output:

```
{  
  "ReplicationTask": {  
    "ReplicationTaskIdentifier": "moveit2",  
    "SourceEndpointArn": "arn:aws:dms:us-  
east-1:123456789012:endpoint:6GGI6YPWYGAYUVLKIB732KEVWA",  
    "TargetEndpointArn": "arn:aws:dms:us-  
east-1:123456789012:endpoint:E0M4SFKCZEYHZBFGAGZT3QEC5U",
```

```

    "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE",
    "MigrationType": "full-load",
    "TableMappings": ...output omitted...,
    "ReplicationTaskSettings": ...output omitted...,
    "Status": "deleting",
    "StopReason": "Stop Reason FULL_LOAD_ONLY_FINISHED",
    "ReplicationTaskCreationDate": 1590524772.505,
    "ReplicationTaskStartDate": 1590789988.677,
    "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII"
  }
}

```

For more information, see [Working with AWS DMS Tasks](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [DeleteReplicationTask](#) in *AWS CLI Command Reference*.

describe-account-attributes

The following code example shows how to use `describe-account-attributes`.

AWS CLI

To describe account attributes

The following `describe-account-attributes` example lists the attributes for your AWS account.

```
aws dms describe-account-attributes
```

Output:

```

{
  "AccountQuotas": [
    {
      "AccountQuotaName": "ReplicationInstances",
      "Used": 1,
      "Max": 20
    },
    {

```

```

        "AccountQuotaName": "AllocatedStorage",
        "Used": 5,
        "Max": 10000
    },
    ...remaining output omitted...

],
"UniqueAccountIdentifier": "cqahfbfy5xee"
}

```

- For API details, see [DescribeAccountAttributes](#) in *AWS CLI Command Reference*.

describe-certificates

The following code example shows how to use describe-certificates.

AWS CLI

To list the available certificates

The following describe-certificates example lists the available certificates in your AWS account.

```
aws dms describe-certificates
```

Output:

```

{
  "Certificates": [
    {
      "CertificateIdentifier": "my-cert",
      "CertificateCreationDate": 1543259542.506,
      "CertificatePem": "-----BEGIN CERTIFICATE-----
\nMIID9DCCAtygAwIBAgIBQjANBgkqhkiG9w0BAQ ...U"

      ... remaining output omitted ...

    }
  ]
}

```

For more information, see [Using SSL](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [DescribeCertificates](#) in *AWS CLI Command Reference*.

describe-connections

The following code example shows how to use describe-connections.

AWS CLI

To describe connections

The following describe-connections example lists the connections that you have tested between a replication instance and an endpoint.

```
aws dms describe-connections
```

Output:

```
{
  "Connections": [
    {
      "Status": "successful",
      "ReplicationInstanceIdentifier": "test",
      "EndpointArn": "arn:aws:dms:us-east-1:123456789012:endpoint:ZW5UAN6P4E77EC7YWHK4RZZ3BE",
      "EndpointIdentifier": "testsrc1",
      "ReplicationInstanceArn": "arn:aws:dms:us-east-1:123456789012:rep:6UTDJGB0US3VI3SUWA66XFJCJQ"
    }
  ]
}
```

For more information, see [Creating Source and Target Endpoints](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [DescribeConnections](#) in *AWS CLI Command Reference*.

describe-endpoint-types

The following code example shows how to use describe-endpoint-types.

AWS CLI

To list the available endpoint types

The following `describe-endpoint-types` example lists the MySQL endpoint types that are available.

```
aws dms describe-endpoint-types \  
  --filters "Name=engine-name,Values=mysql"
```

Output:

```
{  
  "SupportedEndpointTypes": [  
    {  
      "EngineName": "mysql",  
      "SupportsCDC": true,  
      "EndpointType": "source",  
      "EngineDisplayName": "MySQL"  
    },  
    {  
      "EngineName": "mysql",  
      "SupportsCDC": true,  
      "EndpointType": "target",  
      "EngineDisplayName": "MySQL"  
    }  
  ]  
}
```

For more information, see [Working with AWS DMS Endpoints](https://docs.aws.amazon.com/dms/latest/userguide/CHAP_Endpoints.html) <https://docs.aws.amazon.com/dms/latest/userguide/CHAP_Endpoints.html>`__ in the *AWS Database Migration Service User Guide*.

- For API details, see [DescribeEndpointTypes](#) in *AWS CLI Command Reference*.

describe-endpoints

The following code example shows how to use `describe-endpoints`.

AWS CLI

To describe endpoints

The following describe-endpoints example lists the endpoints in your AWS account.

```
aws dms describe-endpoints
```

Output:

```
{
  "Endpoints": [
    {
      "Username": "dms",
      "Status": "active",
      "EndpointArn": "arn:aws:dms:us-east-1:123456789012:endpoint:SF2W0FLWYWKVE0HID2EKLP3SJI",
      "ServerName": "ec2-52-32-48-61.us-west-2.compute.amazonaws.com",
      "EndpointType": "SOURCE",
      "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/94d5c4e7-4e4c-44be-b58a-c8da7adf57cd",
      "DatabaseName": "test",
      "EngineName": "mysql",
      "EndpointIdentifier": "pri100",
      "Port": 8193
    },
    {
      "Username": "admin",
      "Status": "active",
      "EndpointArn": "arn:aws:dms:us-east-1:123456789012:endpoint:TJJZCIH3CJ24TJRU4VC32WEWFR",
      "ServerName": "test.example.com",
      "EndpointType": "SOURCE",
      "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/2431021b-1cf2-a2d4-77b2-59a9e4bce323",
      "DatabaseName": "EMPL",
      "EngineName": "oracle",
      "EndpointIdentifier": "test",
      "Port": 1521
    }
  ]
}
```

For more information, see [Working with AWS DMS Endpoints](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [DescribeEndpoints](#) in *AWS CLI Command Reference*.

describe-event-categories

The following code example shows how to use `describe-event-categories`.

AWS CLI

To describe event categories

The following `describe-event-categories` example lists the available event categories.

```
aws dms describe-event-categories
```

Output:

```
{
  "EventCategoryGroupList": [
    {
      "SourceType": "replication-instance",
      "EventCategories": [
        "low storage",
        "configuration change",
        "maintenance",
        "deletion",
        "creation",
        "failover",
        "failure"
      ]
    },
    {
      "SourceType": "replication-task",
      "EventCategories": [
        "configuration change",
        "state change",
        "deletion",
        "creation",
        "failure"
      ]
    }
  ]
}
```

For more information, see [Working with Events and Notifications](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [DescribeEventCategories](#) in *AWS CLI Command Reference*.

describe-event-subscriptions

The following code example shows how to use describe-event-subscriptions.

AWS CLI

To describe event subscriptions

The following describe-event-subscriptions example lists the event subscriptions to an Amazon SNS topic.

```
aws dms describe-event-subscriptions
```

Output:

```
{
  "EventSubscriptionsList": [
    {
      "CustomerAwsId": "123456789012",
      "CustSubscriptionId": "my-dms-events",
      "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:my-sns-topic",
      "Status": "deleting",
      "SubscriptionCreationTime": "2020-05-21 22:28:51.924",
      "Enabled": true
    }
  ]
}
```

For more information, see [Working with Events and Notifications](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [DescribeEventSubscriptions](#) in *AWS CLI Command Reference*.

describe-events

The following code example shows how to use describe-events.

AWS CLI

To list DMS events

The following `describe-events` example lists the events that originated from a replication instance.

```
aws dms describe-events \  
  --source-type "replication-instance"
```

Output:

```
{  
  "Events": [  
    {  
      "SourceIdentifier": "my-repl-instance",  
      "SourceType": "replication-instance",  
      "Message": "Replication application shutdown",  
      "EventCategories": [],  
      "Date": 1590771645.776  
    }  
  ]  
}
```

For more information, see [Working with Events and Notifications](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [DescribeEvents](#) in *AWS CLI Command Reference*.

describe-orderable-replication-instances

The following code example shows how to use `describe-orderable-replication-instances`.

AWS CLI

To describe orderable replication instances

The following `describe-orderable-replication-instances` example lists replication instance types that you can order.

```
aws dms describe-orderable-replication-instances
```

Output:

```
{
```

```
"OrderableReplicationInstances": [  
  {  
    "EngineVersion": "3.3.2",  
    "ReplicationInstanceClass": "dms.c4.2xlarge",  
    "StorageType": "gp2",  
    "MinAllocatedStorage": 5,  
    "MaxAllocatedStorage": 6144,  
    "DefaultAllocatedStorage": 100,  
    "IncludedAllocatedStorage": 100,  
    "AvailabilityZones": [  
      "us-east-1a",  
      "us-east-1b",  
      "us-east-1c",  
      "us-east-1d",  
      "us-east-1e",  
      "us-east-1f"  
    ]  
  },  
  {  
    "EngineVersion": "3.3.2",  
    "ReplicationInstanceClass": "dms.c4.4xlarge",  
    "StorageType": "gp2",  
    "MinAllocatedStorage": 5,  
    "MaxAllocatedStorage": 6144,  
    "DefaultAllocatedStorage": 100,  
    "IncludedAllocatedStorage": 100,  
    "AvailabilityZones": [  
      "us-east-1a",  
      "us-east-1b",  
      "us-east-1c",  
      "us-east-1d",  
      "us-east-1e",  
      "us-east-1f"  
    ]  
  },  
  ...remaining output omitted...  
]
```

For more information, see [Working with an AWS DMS Replication Instance](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [DescribeOrderableReplicationInstances](#) in *AWS CLI Command Reference*.

describe-refresh-schemas-status

The following code example shows how to use `describe-refresh-schemas-status`.

AWS CLI

To list the refresh status for an endpoint

The following `describe-refresh-schemas-status` example returns the status of a previous refresh request.

```
aws dms describe-refresh-schemas-status \  
  --endpoint-arn arn:aws:dms:us-  
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA
```

Output:

```
{  
  "RefreshSchemasStatus": {  
    "EndpointArn": "arn:aws:dms:us-  
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA",  
    "ReplicationInstanceArn": "arn:aws:dms:us-  
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE",  
    "Status": "successful",  
    "LastRefreshDate": 1590786544.605  
  }  
}
```

- For API details, see [DescribeRefreshSchemasStatus](#) in *AWS CLI Command Reference*.

describe-replication-instances

The following code example shows how to use `describe-replication-instances`.

AWS CLI

To describe replication instances

The following `describe-replication-instances` example lists the replication instances in your AWS account.

```
aws dms describe-replication-instances
```

Output:

```
{
  "ReplicationInstances": [
    {
      "ReplicationInstanceIdentifier": "my-repl-instance",
      "ReplicationInstanceClass": "dms.t2.micro",
      "ReplicationInstanceStatus": "available",
      "AllocatedStorage": 5,
      "InstanceCreateTime": 1590011235.952,
      "VpcSecurityGroups": [
        {
          "VpcSecurityGroupId": "sg-f839b688",
          "Status": "active"
        }
      ],
      "AvailabilityZone": "us-east-1e",
      "ReplicationSubnetGroup": {
        "ReplicationSubnetGroupIdentifier": "default",
        "ReplicationSubnetGroupDescription": "default",
        "VpcId": "vpc-136a4c6a",
        "SubnetGroupStatus": "Complete",
        "Subnets": [
          {
            "SubnetIdentifier": "subnet-da327bf6",
            "SubnetAvailabilityZone": {
              "Name": "us-east-1a"
            },
            "SubnetStatus": "Active"
          },
          {
            "SubnetIdentifier": "subnet-42599426",
            "SubnetAvailabilityZone": {
              "Name": "us-east-1d"
            },
            "SubnetStatus": "Active"
          },
          {
            "SubnetIdentifier": "subnet-bac383e0",
            "SubnetAvailabilityZone": {
              "Name": "us-east-1c"
            },
            "SubnetStatus": "Active"
          }
        ]
      }
    }
  ]
}
```

```
    {
      "SubnetIdentifier": "subnet-6746046b",
      "SubnetAvailabilityZone": {
        "Name": "us-east-1f"
      },
      "SubnetStatus": "Active"
    },
    {
      "SubnetIdentifier": "subnet-d7c825e8",
      "SubnetAvailabilityZone": {
        "Name": "us-east-1e"
      },
      "SubnetStatus": "Active"
    },
    {
      "SubnetIdentifier": "subnet-cbfff283",
      "SubnetAvailabilityZone": {
        "Name": "us-east-1b"
      },
      "SubnetStatus": "Active"
    }
  ]
},
"PreferredMaintenanceWindow": "wed:11:42-wed:12:12",
"PendingModifiedValues": {
  "MultiAZ": true
},
"MultiAZ": false,
"EngineVersion": "3.3.2",
"AutoMinorVersionUpgrade": true,
"KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/
f7bc0f8e-1a3a-4ace-9faa-e8494fa3921a",
"ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE",
"ReplicationInstancePublicIpAddress": "3.230.18.248",
"ReplicationInstancePrivateIpAddress": "172.31.75.90",
"ReplicationInstancePublicIpAddresses": [
  "3.230.18.248"
],
"ReplicationInstancePrivateIpAddresses": [
  "172.31.75.90"
],
"PubliclyAccessible": true,
"FreeUntil": 1590194829.267
```

```

    }
  ]
}

```

For more information, see [Working with an AWS DMS Replication Instance](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [DescribeReplicationInstances](#) in *AWS CLI Command Reference*.

describe-replication-subnet-groups

The following code example shows how to use describe-replication-subnet-groups.

AWS CLI

To display the available subnet groups

The following describe-replication-subnet-groups example lists the available subnet groups.

```

aws dms describe-replication-subnet-groups \
  --filter "Name=replication-subnet-group-id,Values=my-subnet-group"

```

Output:

```

{
  "ReplicationSubnetGroups": [
    {
      "ReplicationSubnetGroupIdentifier": "my-subnet-group",
      "ReplicationSubnetGroupDescription": "my subnet group",
      "VpcId": "vpc-136a4c6a",
      "SubnetGroupStatus": "Complete",
      "Subnets": [
        {
          "SubnetIdentifier": "subnet-da327bf6",
          "SubnetAvailabilityZone": {
            "Name": "us-east-1a"
          },
          "SubnetStatus": "Active"
        },
        {
          "SubnetIdentifier": "subnet-bac383e0",

```

```

        "SubnetAvailabilityZone": {
            "Name": "us-east-1c"
        },
        "SubnetStatus": "Active"
    },
    {
        "SubnetIdentifier": "subnet-d7c825e8",
        "SubnetAvailabilityZone": {
            "Name": "us-east-1e"
        },
        "SubnetStatus": "Active"
    }
]
}

```

For more information, see [Setting Up a Network for a Replication Instance](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [DescribeReplicationSubnetGroups](#) in *AWS CLI Command Reference*.

describe-replication-task-assessment-results

The following code example shows how to use `describe-replication-task-assessment-results`.

AWS CLI

To list the results of replication task assessmentss

The following `describe-replication-task-assessment-results` example lists the results of a prior task assessment.

```
aws dms describe-replication-task-assessment-results
```

Output:

```
{
  "ReplicationTaskAssessmentResults": [
    {
```



```

        "ReplicationTaskIdentifier": "moveit2",
        "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII",
        "ReplicationTaskLastAssessmentDate": 1590790230.0,
        "AssessmentStatus": "No issues found",
        "AssessmentResultsFile": "moveit2/2020-05-29-22-10"
    }
]
}

```

For more information, see [Creating a Task Assessment Report](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [DescribeReplicationTaskAssessmentResults](#) in *AWS CLI Command Reference*.

describe-replication-tasks

The following code example shows how to use describe-replication-tasks.

AWS CLI

To describe a replication task

The following describe-replication-tasks example describes current replication tasks.

```
aws dms describe-replication-tasks
```

Output:

```

{
  "ReplicationTasks": [
    {
      "ReplicationTaskIdentifier": "moveit2",
      "SourceEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA",
      "TargetEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:E0M4SFKCZEYHZBFGAGZT3QEC5U",
      "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE",
      "MigrationType": "full-load",
      "TableMappings": ...output omitted... ,
    }
  ]
}

```

```

    "ReplicationTaskSettings": ...output omitted... ,
    "Status": "stopped",
    "StopReason": "Stop Reason FULL_LOAD_ONLY_FINISHED",
    "ReplicationTaskCreationDate": 1590524772.505,
    "ReplicationTaskStartDate": 1590619805.212,
    "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII",
    "ReplicationTaskStats": {
      "FullLoadProgressPercent": 100,
      "ElapsedTimeMillis": 0,
      "TablesLoaded": 0,
      "TablesLoading": 0,
      "TablesQueued": 0,
      "TablesErrored": 0,
      "FreshStartDate": 1590619811.528,
      "StartDate": 1590619811.528,
      "StopDate": 1590619842.068
    }
  }
]
}

```

For more information, see [Working with AWS DMS Tasks](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [DescribeReplicationTasks](#) in *AWS CLI Command Reference*.

describe-schemas

The following code example shows how to use `describe-schemas`.

AWS CLI

To describe database schemas

The following `describe-schemas` example lists the available tables at an endpoint.

```

aws dms describe-schemas \
  --endpoint-arn "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA"

```

Output:

```
{
  "Schemas": [
    "prodrep"
  ]
}
```

For more information, see [This is the topic title](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [DescribeSchemas](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags for a resource

The following `list-tags-for-resource` example lists the tags for a replication instance.

```
aws dms list-tags-for-resource \
  --resource-arn arn:aws:dms:us-east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE
```

Output:

```
{
  "TagList": [
    {
      "Key": "Project",
      "Value": "dbMigration"
    },
    {
      "Key": "Environment",
      "Value": "PROD"
    }
  ]
}
```

For more information, see [Tagging Resources](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

modify-endpoint

The following code example shows how to use `modify-endpoint`.

AWS CLI

To modify an endpoint

The following `modify-endpoint` example adds an extra connection attribute to an endpoint.

```
aws dms modify-endpoint \  
  --endpoint-arn "arn:aws:dms:us-  
east-1:123456789012:endpoint:GUVAFG34EECU0J6QVZ56DAHT3U" \  
  --extra-connection-attributes "compressionType=GZIP"
```

Output:

```
{  
  "Endpoint": {  
    "EndpointIdentifier": "src-endpoint",  
    "EndpointType": "SOURCE",  
    "EngineName": "s3",  
    "EngineDisplayName": "Amazon S3",  
    "ExtraConnectionAttributes":  
"compressionType=GZIP;csvDelimiter=,;csvRowDelimiter=\\n;",  
    "Status": "active",  
    "EndpointArn": "arn:aws:dms:us-  
east-1:123456789012:endpoint:GUVAFG34EECU0J6QVZ56DAHT3U",  
    "SslMode": "none",  
    "ServiceAccessRoleArn": "arn:aws:iam::123456789012:role/my-s3-access-role",  
    "S3Settings": {  
      "ServiceAccessRoleArn": "arn:aws:iam::123456789012:role/my-s3-access-  
role",  
      "CsvRowDelimiter": "\\n",  
      "CsvDelimiter": ",",  
      "BucketFolder": "",  
      "BucketName": "",  
      "CompressionType": "GZIP",  
      "EnableStatistics": true  
    }  
  }  
}
```

```
}  
}
```

For more information, see [Working with AWS DMS Endpoints](https://docs.aws.amazon.com/dms/latest/userguide/CHAP_Endpoints.html) <https://docs.aws.amazon.com/dms/latest/userguide/CHAP_Endpoints.html>`__ in the *AWS Database Migration Service User Guide*.

- For API details, see [ModifyEndpoint](#) in *AWS CLI Command Reference*.

modify-event-subscription

The following code example shows how to use `modify-event-subscription`.

AWS CLI

To modify an event subscription

The following `modify-event-subscription` example changes the source type of an event subscription.

```
aws dms modify-event-subscription \  
  --subscription-name "my-dms-events" \  
  --source-type replication-task
```

Output:

```
{  
  "EventSubscription": {  
    "CustomerAwsId": "123456789012",  
    "CustSubscriptionId": "my-dms-events",  
    "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:my-sns-topic",  
    "Status": "modifying",  
    "SubscriptionCreationTime": "2020-05-29 17:04:40.262",  
    "SourceType": "replication-task",  
    "Enabled": true  
  }  
}
```

For more information, see [Working with Events and Notifications](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [ModifyEventSubscription](#) in *AWS CLI Command Reference*.

modify-replication-instance

The following code example shows how to use `modify-replication-instance`.

AWS CLI

To modify a replication instance

The following `modify-replication-instance` example modifies a replication instance so that it uses a Multi-AZ deployment.

```
aws dms modify-replication-instance \
  --replication-instance-arn arn:aws:dms:us-
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE \
  --multi-az
```

Output:

```
{
  "ReplicationInstance": {
    "ReplicationInstanceIdentifier": "my-repl-instance",
    "ReplicationInstanceClass": "dms.t2.micro",
    "ReplicationInstanceStatus": "available",
    "AllocatedStorage": 5,
    "InstanceCreateTime": 1590011235.952,
    ...output omitted...

    "PendingModifiedValues": {
      "MultiAZ": true
    },
    "MultiAZ": false,
    "EngineVersion": "3.3.2",
    "AutoMinorVersionUpgrade": true,
    "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/f7bc0f8e-1a3a-4ace-9faa-
e8494fa3921a",
    ...output omitted...
  }
}
```

```
}
```

For more information, see [Working with an AWS DMS Replication Instance](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [ModifyReplicationInstance](#) in *AWS CLI Command Reference*.

modify-replication-subnet-group

The following code example shows how to use `modify-replication-subnet-group`.

AWS CLI

To modify a subnet group

The following `modify-replication-subnet-group` example changes the lists of subnets associated with a subnet group.

```
aws dms modify-replication-subnet-group \  
  --replication-subnet-group-identifier my-subnet-group \  
  --subnet-id subnet-da327bf6 subnet-bac383e0
```

Output:

```
{  
  "ReplicationSubnetGroup": {  
    "ReplicationSubnetGroupIdentifier": "my-subnet-group",  
    "ReplicationSubnetGroupDescription": "my subnet group",  
    "VpcId": "vpc-136a4c6a",  
    "SubnetGroupStatus": "Complete",  
    "Subnets": [  
      {  
        "SubnetIdentifier": "subnet-da327bf6",  
        "SubnetAvailabilityZone": {  
          "Name": "us-east-1a"  
        },  
        "SubnetStatus": "Active"  
      },  
      {  
        "SubnetIdentifier": "subnet-bac383e0",  
        "SubnetAvailabilityZone": {  
          "Name": "us-east-1c"  
        }  
      }  
    ]  
  }  
}
```

```

        },
        "SubnetStatus": "Active"
    }
]
}
}

```

For more information, see [Setting Up a Network for a Replication Instance](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [ModifyReplicationSubnetGroup](#) in *AWS CLI Command Reference*.

modify-replication-task

The following code example shows how to use `modify-replication-task`.

AWS CLI

To modify a replication task

The following `modify-replication-task` example changes the table mappings for a task.

```

aws dms modify-replication-task \
  --replication-task-arn "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII" \
  --table-mappings file://table-mappings.json

```

Contents of `table-mappings.json`:

```

{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "prodrep",
        "table-name": "ACCT_%"
      },
      "rule-action": "include",
      "filters": []
    }
  ]
}

```



```
]
}
```

Output:

```
{
  "ReplicationTask": {
    "ReplicationTaskIdentifier": "moveit2",
    "SourceEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWGWAYUVLKIB732KEVWA",
    "TargetEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:E0M4SFKCZEYHZBFGAGZT3QEC5U",
    "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE",
    "MigrationType": "full-load",
    "TableMappings": "...output omitted...",
    "ReplicationTaskSettings": "...output omitted...",
    "Status": "modifying",
    "StopReason": "Stop Reason FULL_LOAD_ONLY_FINISHED",
    "ReplicationTaskCreationDate": 1590524772.505,
    "ReplicationTaskStartDate": 1590789424.653,
    "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII"
  }
}
```

For more information, see [Working with AWS DMS Tasks](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [ModifyReplicationTask](#) in *AWS CLI Command Reference*.

reboot-replication-instance

The following code example shows how to use `reboot-replication-instance`.

AWS CLI

To reboot a replication instance

The following `reboot-replication-instance` example reboots a replication instance.

```
aws dms reboot-replication-instance \
```

```
--replication-instance-arn arn:aws:dms:us-
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE
```

Output:

```
{
  "ReplicationInstance": {
    "ReplicationInstanceIdentifier": "my-repl-instance",
    "ReplicationInstanceClass": "dms.t2.micro",
    "ReplicationInstanceStatus": "rebooting",
    "AllocatedStorage": 5,
    "InstanceCreateTime": 1590011235.952,
    ... output omitted ...
  }
}
```

For more information, see [Working with an AWS DMS Replication Instance](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [RebootReplicationInstance](#) in *AWS CLI Command Reference*.

refresh-schemas

The following code example shows how to use `refresh-schemas`.

AWS CLI

To refresh database schemas

The following `refresh-schemas` example requests that AWS DMS refresh the list of schemas at an endpoint.

```
aws dms refresh-schemas \
  --replication-instance-arn arn:aws:dms:us-
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE \
  --endpoint-arn "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA"
```

Output:

```
{
  "RefreshSchemasStatus": {
```

```
    "EndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWGWAYUVLKIB732KEVWA",
    "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE",
    "Status": "refreshing",
    "LastRefreshDate": 1590019949.103
  }
}
```

- For API details, see [RefreshSchemas](#) in *AWS CLI Command Reference*.

reload-tables

The following code example shows how to use `reload-tables`.

AWS CLI

To refresh the list of tables available at an endpoint

The following `reload-tables` example reloads the list of available tables at an endpoint.

```
aws dms reload-tables \
  --replication-task-arn "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII" \
  --tables-to-reload "SchemaName=prodrep,TableName=ACCT_BAL"
```

Output:

```
{
  "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII"
}
```

- For API details, see [ReloadTables](#) in *AWS CLI Command Reference*.

remove-tags-from-resource

The following code example shows how to use `remove-tags-from-resource`.

AWS CLI

To remove tags from a replication instance

The following `remove-tags-from-resource` example removes tags from a replication instance.

```
aws dms remove-tags-from-resource \  
  --resource-arn arn:aws:dms:us-east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE \  
  --tag-keys Environment Project
```

This command produces no output.

For more information, see [Tagging Resources](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [RemoveTagsFromResource](#) in *AWS CLI Command Reference*.

start-replication-task-assessment

The following code example shows how to use `start-replication-task-assessment`.

AWS CLI

To start a task assessment

The following `start-replication-task-assessment` example starts a replication task assessment.

```
aws dms start-replication-task-assessment \  
  --replication-task-arn arn:aws:dms:us-   
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII
```

Output:

```
{  
  "ReplicationTask": {  
    "ReplicationTaskIdentifier": "moveit2",  
    "SourceEndpointArn": "arn:aws:dms:us-   
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA",  
    "TargetEndpointArn": "arn:aws:dms:us-   
east-1:123456789012:endpoint:E0M4SFKCZEYHZBFGAGZT3QEC5U",  
    "ReplicationInstanceArn": "arn:aws:dms:us-   
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE",  
    "MigrationType": "full-load",
```

```

    "TableMappings": ...output omitted...,
    "ReplicationTaskSettings": ...output omitted...,
    "Status": "testing",
    "StopReason": "Stop Reason FULL_LOAD_ONLY_FINISHED",
    "ReplicationTaskCreationDate": 1590524772.505,
    "ReplicationTaskStartDate": 1590789988.677,
    "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII"
  }
}

```

For more information, see [Creating a Task Assessment Report](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [StartReplicationTaskAssessment](#) in *AWS CLI Command Reference*.

start-replication-task

The following code example shows how to use start-replication-task.

AWS CLI

To start a replication task

The following command-name example lists the available widgets in your AWS account.

```

aws dms start-replication-task \
  --replication-task-arn arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII \
  --start-replication-task-type reload-target

```

Output:

```

{
  "ReplicationTask": {
    "ReplicationTaskIdentifier": "moveit2",
    "SourceEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWYGAYUVLKIB732KEVWA",
    "TargetEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:E0M4SFKCZEYHZBFGAGZT3QEC5U",
    "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE",

```

```

    "MigrationType": "full-load",
    "TableMappings": ...output omitted... ,
    "ReplicationTaskSettings": ...output omitted... ,
    "Status": "starting",
    "ReplicationTaskCreationDate": 1590524772.505,
    "ReplicationTaskStartDate": 1590619805.212,
    "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII"
  }
}

```

For more information, see [Working with AWS DMS Tasks](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [StartReplicationTask](#) in *AWS CLI Command Reference*.

stop-replication-task

The following code example shows how to use stop-replication-task.

AWS CLI

To stop a task

The following stop-replication-task example stops a task.

```

aws dms stop-replication-task \
  --replication-task-arn arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII

```

Output:

```

{
  "ReplicationTask": {
    "ReplicationTaskIdentifier": "moveit2",
    "SourceEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA",
    "TargetEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:E0M4SFKCZEYHZBFGAGZT3QEC5U",
    "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE",
    "MigrationType": "full-load",

```

```

    "TableMappings": ...output omitted...,
    "ReplicationTaskSettings": ...output omitted...,
    "Status": "stopping",
    "ReplicationTaskCreationDate": 1590524772.505,
    "ReplicationTaskStartDate": 1590789424.653,
    "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII"
  }
}

```

For more information, see [Working with AWS DMS Tasks](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [StopReplicationTask](#) in *AWS CLI Command Reference*.

test-connection

The following code example shows how to use test-connection.

AWS CLI

To test a connection to an endpoint

The following test-connection example tests whether an endpoint can be accessed from a replication instance.

```

aws dms test-connection \
  --replication-instance-arn arn:aws:dms:us-
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE \
  --endpoint-arn arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA

```

Output:

```

{
  "Connection": {
    "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE",
    "EndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA",
    "Status": "testing",
  }
}

```

```
    "EndpointIdentifier": "src-database-1",  
    "ReplicationInstanceIdentifier": "my-repl-instance"  
  }  
}
```

For more information, see [Creating source and target endpoints](#) in the *AWS Database Migration Service User Guide*.

- For API details, see [TestConnection](#) in *AWS CLI Command Reference*.

Amazon DocumentDB examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon DocumentDB.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-tags-to-resource

The following code example shows how to use `add-tags-to-resource`.

AWS CLI

To add one or more tags to a specified resource

The following `add-tags-to-resource` example adds three tags to `sample-cluster`. One tag (`CropB`) has a key name but no value.


```
aws docdb add-tags-to-resource \  
  --resource-name arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster \  
  --tags Key="CropA",Value="Apple" Key="CropB" Key="CropC",Value="Corn"
```

This command produces no output.

For more information, see [Tagging Amazon DocumentDB Resources](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [AddTagsToResource](#) in *AWS CLI Command Reference*.

apply-pending-maintenance-action

The following code example shows how to use `apply-pending-maintenance-action`.

AWS CLI

To have pending maintenance actions take place during the next maintenance window

The following `apply-pending-maintenance-action` example causes all system-update actions to be performed during the next scheduled maintenance window.

```
aws docdb apply-pending-maintenance-action \  
  --resource-identifier arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster \  
  --apply-action system-update \  
  --opt-in-type next-maintenance
```

This command produces no output.

For more information, see [Applying Amazon DocumentDB Updates](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [ApplyPendingMaintenanceAction](#) in *AWS CLI Command Reference*.

copy-db-cluster-parameter-group

The following code example shows how to use `copy-db-cluster-parameter-group`.

AWS CLI

To duplicate an existing DB cluster parameter group

The following `copy-db-cluster-parameter-group` example makes a copy of the parameter group `custom-docdb3-6` named `custom-docdb3-6-copy`. When making the copy it adds tags to the new parameter group.

```
aws docdb copy-db-cluster-parameter-group \  
  --source-db-cluster-parameter-group-identifier custom-docdb3-6 \  
  --target-db-cluster-parameter-group-identifier custom-docdb3-6-copy \  
  --target-db-cluster-parameter-group-description "Copy of custom-docdb3-6" \  
  --tags Key="CopyNumber",Value="1" Key="Modifiable",Value="Yes"
```

Output:

```
{  
  "DBClusterParameterGroup": {  
    "DBParameterGroupFamily": "docdb3.6",  
    "DBClusterParameterGroupArn": "arn:aws:rds:us-east-1:12345678901:cluster-  
pg:custom-docdb3-6-copy",  
    "DBClusterParameterGroupName": "custom-docdb3-6-copy",  
    "Description": "Copy of custom-docdb3-6"  
  }  
}
```

For more information, see [Copying an Amazon DocumentDB Cluster Parameter Group](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [CopyDbClusterParameterGroup](#) in *AWS CLI Command Reference*.

copy-db-cluster-snapshot

The following code example shows how to use `copy-db-cluster-snapshot`.

AWS CLI

To create a copy of a snapshot

The following `copy-db-cluster-snapshot` example makes a copy of `sample-cluster-snapshot` named `sample-cluster-snapshot-copy`. The copy has all the tags of the original plus a new tag with the key name `CopyNumber`.

```
aws docdb copy-db-cluster-snapshot \  
  --source-db-cluster-snapshot-identifier sample-cluster-snapshot \  
  --target-db-cluster-snapshot-identifier sample-cluster-snapshot-copy \  
  --tags Key="CopyNumber",Value="1" Key="Modifiable",Value="Yes"
```

```
--target-db-cluster-snapshot-identifier sample-cluster-snapshot-copy \  
--copy-tags \  
--tags Key="CopyNumber",Value="1"
```

This command produces no output.

For more information, see [Copying a Cluster Snapshot](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [CopyDbClusterSnapshot](#) in *AWS CLI Command Reference*.

create-db-cluster-parameter-group

The following code example shows how to use `create-db-cluster-parameter-group`.

AWS CLI

To create an Amazon DocumentDB cluster parameter group

The following `create-db-cluster-parameter-group` example creates the DB cluster parameter group `sample-parameter-group` using the `docdb3.6` family.

```
aws docdb create-db-cluster-parameter-group \  
  --db-cluster-parameter-group-name sample-parameter-group \  
  --db-parameter-group-family docdb3.6 \  
  --description "Sample parameter group based on docdb3.6"
```

Output:

```
{  
  "DBClusterParameterGroup": {  
    "Description": "Sample parameter group based on docdb3.6",  
    "DBParameterGroupFamily": "docdb3.6",  
    "DBClusterParameterGroupArn": "arn:aws:rds:us-west-2:123456789012:cluster-  
pg:sample-parameter-group",  
    "DBClusterParameterGroupName": "sample-parameter-group"  
  }  
}
```

For more information, see [Creating an Amazon DocumentDB Cluster Parameter Group](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [CreateDbClusterParameterGroup](#) in *AWS CLI Command Reference*.

create-db-cluster-snapshot

The following code example shows how to use `create-db-cluster-snapshot`.

AWS CLI

To create a manual Amazon DocumentDB cluster snapshot

The following `create-db-cluster-snapshot` example creates an Amazon DB cluster snapshot named `sample-cluster-snapshot`.

```
aws docdb create-db-cluster-snapshot \  
  --db-cluster-identifier sample-cluster \  
  --db-cluster-snapshot-identifier sample-cluster-snapshot
```

Output:

```
{  
  "DBClusterSnapshot": {  
    "MasterUsername": "master-user",  
    "SnapshotCreateTime": "2019-03-18T18:27:14.794Z",  
    "AvailabilityZones": [  
      "us-west-2a",  
      "us-west-2b",  
      "us-west-2c",  
      "us-west-2d",  
      "us-west-2e",  
      "us-west-2f"  
    ],  
    "SnapshotType": "manual",  
    "DBClusterSnapshotArn": "arn:aws:rds:us-west-2:123456789012:cluster-  
snapshot:sample-cluster-snapshot",  
    "EngineVersion": "3.6.0",  
    "PercentProgress": 0,  
    "DBClusterSnapshotIdentifier": "sample-cluster-snapshot",  
    "Engine": "docdb",  
    "DBClusterIdentifier": "sample-cluster",  
    "Status": "creating",  
    "ClusterCreateTime": "2019-03-15T20:29:58.836Z",  
    "Port": 0,  
  }  
}
```

```
    "StorageEncrypted": false,  
    "VpcId": "vpc-91280df6"  
  }  
}
```

For more information, see [Creating a Manual Cluster Snapshot](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [CreateDbClusterSnapshot](#) in *AWS CLI Command Reference*.

create-db-cluster

The following code example shows how to use `create-db-cluster`.

AWS CLI

To create an Amazon DocumentDB cluster

The following `create-db-cluster` example creates an Amazon DocumentDB cluster named `sample-cluster` with the preferred maintenance window on Sundays between 20:30 and 11:00.

```
aws docdb create-db-cluster \  
  --db-cluster-identifier sample-cluster \  
  --engine docdb \  
  --master-username master-user \  
  --master-user-password password \  
  --preferred-maintenance-window Sun:20:30-Sun:21:00
```

Output:

```
{  
  "DBCluster": {  
    "DBClusterParameterGroup": "default.docdb3.6",  
    "AssociatedRoles": [],  
    "DBSubnetGroup": "default",  
    "ClusterCreateTime": "2019-03-18T18:06:34.616Z",  
    "Status": "creating",  
    "Port": 27017,  
    "PreferredMaintenanceWindow": "sun:20:30-sun:21:00",  
    "HostedZoneId": "ZNKXH85TT8WVW",  
    "DBClusterMembers": [],  
  }  
}
```

```
"Engine": "docdb",
"DBClusterIdentifier": "sample-cluster",
"PreferredBackupWindow": "10:12-10:42",
"AvailabilityZones": [
  "us-west-2d",
  "us-west-2f",
  "us-west-2e"
],
"MasterUsername": "master-user",
"BackupRetentionPeriod": 1,
"ReaderEndpoint": "sample-cluster.cluster-ro-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
"VpcSecurityGroups": [
  {
    "VpcSecurityGroupId": "sg-77186e0d",
    "Status": "active"
  }
],
"StorageEncrypted": false,
"DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster",
"DbClusterResourceId": "cluster-L3R4YRSBUYDP4GLMTJ2WF5GH5Q",
"MultiAZ": false,
"Endpoint": "sample-cluster.cluster-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
"EngineVersion": "3.6.0"
}
}
```

For more information, see [Creating an Amazon DocumentDB Cluster](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [CreateDbCluster](#) in *AWS CLI Command Reference*.

create-db-instance

The following code example shows how to use `create-db-instance`.

AWS CLI

To create an Amazon DocumentDB cluster instance

The following `create-db-instance` example code creates the instance `sample-cluster-instance-2` in the Amazon DocumentDB cluster `sample-cluster`.

```
aws docdb create-db-instance \  
  --db-cluster-identifier sample-cluster \  
  --db-instance-class db.r4.xlarge \  
  --db-instance-identifier sample-cluster-instance-2 \  
  --engine docdb
```

Output:

```
{  
  "DBInstance": {  
    "DBInstanceStatus": "creating",  
    "PendingModifiedValues": {  
      "PendingCloudwatchLogsExports": {  
        "LogTypesToEnable": [  
          "audit"  
        ]  
      }  
    },  
    "PubliclyAccessible": false,  
    "PreferredBackupWindow": "00:00-00:30",  
    "PromotionTier": 1,  
    "EngineVersion": "3.6.0",  
    "BackupRetentionPeriod": 3,  
    "DBInstanceIdentifier": "sample-cluster-instance-2",  
    "PreferredMaintenanceWindow": "tue:10:28-tue:10:58",  
    "StorageEncrypted": false,  
    "Engine": "docdb",  
    "DBClusterIdentifier": "sample-cluster",  
    "DBSubnetGroup": {  
      "Subnets": [  
        {  
          "SubnetAvailabilityZone": {  
            "Name": "us-west-2a"  
          },  
          "SubnetStatus": "Active",  
          "SubnetIdentifier": "subnet-4e26d263"  
        },  
        {  
          "SubnetAvailabilityZone": {  
            "Name": "us-west-2c"  
          },  
          "SubnetStatus": "Active",  
          "SubnetIdentifier": "subnet-afc329f4"  
        }  
      ]  
    }  
  }  
}
```

```

    },
    {
      "SubnetAvailabilityZone": {
        "Name": "us-west-2d"
      },
      "SubnetStatus": "Active",
      "SubnetIdentifier": "subnet-53ab3636"
    },
    {
      "SubnetAvailabilityZone": {
        "Name": "us-west-2b"
      },
      "SubnetStatus": "Active",
      "SubnetIdentifier": "subnet-991cb8d0"
    }
  ],
  "DBSubnetGroupDescription": "default",
  "SubnetGroupStatus": "Complete",
  "VpcId": "vpc-91280df6",
  "DBSubnetGroupName": "default"
},
"DBInstanceClass": "db.r4.xlarge",
"VpcSecurityGroups": [
  {
    "Status": "active",
    "VpcSecurityGroupId": "sg-77186e0d"
  }
],
"DBInstanceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster-
instance-2",
"DbiResourceId": "db-XEKJLEMGRV5ZKCARUVA4H03ITE"
}
}

```

For more information, see [Adding an Amazon DocumentDB Instance to a Cluster](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [CreateDbInstance](#) in *AWS CLI Command Reference*.

create-db-subnet-group

The following code example shows how to use `create-db-subnet-group`.

AWS CLI

To create an Amazon DocumentDB subnet group

The following `create-db-subnet-group` example creates an Amazon DocumentDB subnet group named `sample-subnet-group`.

```
aws docdb create-db-subnet-group \  
  --db-subnet-group-description "a sample subnet group" \  
  --db-subnet-group-name sample-subnet-group \  
  --subnet-ids "subnet-29ab1025" "subnet-991cb8d0" "subnet-53ab3636"
```

Output:

```
{  
  "DBSubnetGroup": {  
    "SubnetGroupStatus": "Complete",  
    "DBSubnetGroupName": "sample-subnet-group",  
    "DBSubnetGroupDescription": "a sample subnet group",  
    "VpcId": "vpc-91280df6",  
    "DBSubnetGroupArn": "arn:aws:rds:us-west-2:123456789012:subgrp:sample-  
subnet-group",  
    "Subnets": [  
      {  
        "SubnetStatus": "Active",  
        "SubnetIdentifier": "subnet-53ab3636",  
        "SubnetAvailabilityZone": {  
          "Name": "us-west-2d"  
        }  
      },  
      {  
        "SubnetStatus": "Active",  
        "SubnetIdentifier": "subnet-991cb8d0",  
        "SubnetAvailabilityZone": {  
          "Name": "us-west-2b"  
        }  
      },  
      {  
        "SubnetStatus": "Active",  
        "SubnetIdentifier": "subnet-29ab1025",  
        "SubnetAvailabilityZone": {  
          "Name": "us-west-2c"  
        }  
      }  
    ]  
  }  
}
```

```
    }  
  ]  
}
```

For more information, see [Creating an Amazon DocumentDB Subnet Group](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [CreateDbSubnetGroup](#) in *AWS CLI Command Reference*.

delete-db-cluster-parameter-group

The following code example shows how to use `delete-db-cluster-parameter-group`.

AWS CLI

To delete an Amazon DocumentDB cluster parameter group

The following `delete-db-cluster-parameter-group` example deletes the Amazon DocumentDB parameter group `sample-parameter-group`.

```
aws docdb delete-db-cluster-parameter-group \  
  --db-cluster-parameter-group-name sample-parameter-group
```

This command produces no output.

For more information, see [Deleting an Amazon DocumentDB Cluster Parameter Group](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [DeleteDbClusterParameterGroup](#) in *AWS CLI Command Reference*.

delete-db-cluster-snapshot

The following code example shows how to use `delete-db-cluster-snapshot`.

AWS CLI

To delete an Amazon DocumentDB cluster snapshot

The following `delete-db-cluster-snapshot` example deletes the Amazon DocumentDB cluster snapshot `sample-cluster-snapshot`.

```
aws docdb delete-db-cluster-snapshot \  
  --db-cluster-snapshot-identifier sample-cluster-snapshot
```

```
--db-cluster-snapshot-identifier sample-cluster-snapshot
```

Output:

```
{
  "DBClusterSnapshot": {
    "DBClusterIdentifier": "sample-cluster",
    "AvailabilityZones": [
      "us-west-2a",
      "us-west-2b",
      "us-west-2c",
      "us-west-2d"
    ],
    "DBClusterSnapshotIdentifier": "sample-cluster-snapshot",
    "VpcId": "vpc-91280df6",
    "DBClusterSnapshotArn": "arn:aws:rds:us-west-2:123456789012:cluster-
snapshot:sample-cluster-snapshot",
    "EngineVersion": "3.6.0",
    "Engine": "docdb",
    "SnapshotCreateTime": "2019-03-18T18:27:14.794Z",
    "Status": "available",
    "MasterUsername": "master-user",
    "ClusterCreateTime": "2019-03-15T20:29:58.836Z",
    "PercentProgress": 100,
    "StorageEncrypted": false,
    "SnapshotType": "manual",
    "Port": 0
  }
}
```

For more information, see [Deleting a Cluster Snapshot](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [DeleteDbClusterSnapshot](#) in *AWS CLI Command Reference*.

delete-db-cluster

The following code example shows how to use `delete-db-cluster`.

AWS CLI

To delete an Amazon DocumentDB cluster

The following `delete-db-cluster` example deletes the Amazon DocumentDB cluster `sample-cluster`. No backup of the cluster is made prior to deleting it. NOTE: You must delete all instances associated with the cluster before you can delete it.

```
aws docdb delete-db-cluster \  
  --db-cluster-identifier sample-cluster \  
  --skip-final-snapshot
```

Output:

```
{  
  "DBCluster": {  
    "DBClusterIdentifier": "sample-cluster",  
    "DBSubnetGroup": "default",  
    "EngineVersion": "3.6.0",  
    "Engine": "docdb",  
    "LatestRestorableTime": "2019-03-18T18:07:24.610Z",  
    "PreferredMaintenanceWindow": "sun:20:30-sun:21:00",  
    "StorageEncrypted": false,  
    "EarliestRestorableTime": "2019-03-18T18:07:24.610Z",  
    "Port": 27017,  
    "VpcSecurityGroups": [  
      {  
        "Status": "active",  
        "VpcSecurityGroupId": "sg-77186e0d"  
      }  
    ],  
    "MultiAZ": false,  
    "MasterUsername": "master-user",  
    "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster",  
    "Status": "available",  
    "PreferredBackupWindow": "10:12-10:42",  
    "ReaderEndpoint": "sample-cluster.cluster-ro-corcjozrlsfc.us-  
west-2.docdb.amazonaws.com",  
    "AvailabilityZones": [  
      "us-west-2c",  
      "us-west-2b",  
      "us-west-2a"  
    ],  
    "Endpoint": "sample-cluster.cluster-corcjozrlsfc.us-  
west-2.docdb.amazonaws.com",  
    "DbClusterResourceId": "cluster-L3R4YRSBUYDP4GLMTJ2WF5GH5Q",  
    "ClusterCreateTime": "2019-03-18T18:06:34.616Z",
```

```

    "AssociatedRoles": [],
    "DBClusterParameterGroup": "default.docdb3.6",
    "HostedZoneId": "ZNKXH85TT8WVW",
    "BackupRetentionPeriod": 1,
    "DBClusterMembers": []
  }
}

```

For more information, see [Deleting an Amazon DocumentDB Cluster](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [DeleteDbCluster](#) in *AWS CLI Command Reference*.

delete-db-instance

The following code example shows how to use `delete-db-instance`.

AWS CLI

To delete an Amazon DocumentDB instance

The following `delete-db-instance` example deletes the Amazon DocumentDB instance `sample-cluster-instance-2`.

```

aws docdb delete-db-instance \
  --db-instance-identifier sample-cluster-instance-2

```

Output:

```

{
  "DBInstance": {
    "DBSubnetGroup": {
      "Subnets": [
        {
          "SubnetAvailabilityZone": {
            "Name": "us-west-2a"
          },
          "SubnetStatus": "Active",
          "SubnetIdentifier": "subnet-4e26d263"
        },
        {
          "SubnetAvailabilityZone": {
            "Name": "us-west-2c"
          }
        }
      ]
    }
  }
}

```

```
    },
    "SubnetStatus": "Active",
    "SubnetIdentifier": "subnet-afc329f4"
  },
  {
    "SubnetAvailabilityZone": {
      "Name": "us-west-2d"
    },
    "SubnetStatus": "Active",
    "SubnetIdentifier": "subnet-53ab3636"
  },
  {
    "SubnetAvailabilityZone": {
      "Name": "us-west-2b"
    },
    "SubnetStatus": "Active",
    "SubnetIdentifier": "subnet-991cb8d0"
  }
],
"DBSubnetGroupName": "default",
"DBSubnetGroupDescription": "default",
"VpcId": "vpc-91280df6",
"SubnetGroupStatus": "Complete"
},
"PreferredBackupWindow": "00:00-00:30",
"InstanceCreateTime": "2019-03-18T18:37:33.709Z",
"DBInstanceClass": "db.r4.xlarge",
"DbiResourceId": "db-XEKJLEMGRV5ZKCARUVA4H03ITE",
"BackupRetentionPeriod": 3,
"Engine": "docdb",
"VpcSecurityGroups": [
  {
    "Status": "active",
    "VpcSecurityGroupId": "sg-77186e0d"
  }
],
"AutoMinorVersionUpgrade": true,
"PromotionTier": 1,
"EngineVersion": "3.6.0",
"Endpoint": {
  "Address": "sample-cluster-instance-2.corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
  "HostedZoneId": "ZNKXH85TT8WVW",
  "Port": 27017
}
```

```
    },
    "DBInstanceIdentifier": "sample-cluster-instance-2",
    "PreferredMaintenanceWindow": "tue:10:28-tue:10:58",
    "EnabledCloudwatchLogsExports": [
        "audit"
    ],
    "PendingModifiedValues": {},
    "DBInstanceStatus": "deleting",
    "PubliclyAccessible": false,
    "DBInstanceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster-
instance-2",
    "DBClusterIdentifier": "sample-cluster",
    "AvailabilityZone": "us-west-2c",
    "StorageEncrypted": false
}
}
```

For more information, see [Deleting an Amazon DocumentDB Instance](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [DeleteDbInstance](#) in *AWS CLI Command Reference*.

delete-db-subnet-group

The following code example shows how to use `delete-db-subnet-group`.

AWS CLI

To delete an Amazon DocumentDB subnet group

The following `delete-db-subnet-group` example deletes the Amazon DocumentDB subnet group `sample-subnet-group`.

```
aws docdb delete-db-subnet-group \
    --db-subnet-group-name sample-subnet-group
```

This command produces no output.

For more information, see [Deleting an Amazon DocumentDB Subnet Group](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [DeleteDbSubnetGroup](#) in *AWS CLI Command Reference*.

describe-db-cluster-parameter-groups

The following code example shows how to use `describe-db-cluster-parameter-groups`.

AWS CLI

To see the details of one or more Amazon DocumentDB cluster parameter groups

The following `describe-db-cluster-parameter-groups` example displays details for the Amazon DocumentDB cluster parameter group `custom3-6-param-grp`.

```
aws docdb describe-db-cluster-parameter-groups \  
  --db-cluster-parameter-group-name custom3-6-param-grp
```

Output:

```
{  
  "DBClusterParameterGroups": [  
    {  
      "DBParameterGroupFamily": "docdb3.6",  
      "DBClusterParameterGroupArn": "arn:aws:rds:us-  
east-1:123456789012:cluster-pg:custom3-6-param-grp",  
      "Description": "Custom docdb3.6 parameter group",  
      "DBClusterParameterGroupName": "custom3-6-param-grp"  
    }  
  ]  
}
```

For more information, see [Viewing Amazon DocumentDB Cluster Parameter Groups](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [DescribeDbClusterParameterGroups](#) in *AWS CLI Command Reference*.

describe-db-cluster-parameters

The following code example shows how to use `describe-db-cluster-parameters`.

AWS CLI

To view the detailed parameter list for an Amazon DocumentDB cluster parameter group.

The following `describe-db-cluster-parameters` example lists the parameters for the Amazon DocumentDB parameter group `custom3-6-param-grp`.


```
aws docdb describe-db-cluster-parameters \  
  --db-cluster-parameter-group-name custom3-6-param-grp
```

Output:

```
{  
  "Parameters": [  
    {  
      "DataType": "string",  
      "ParameterName": "audit_logs",  
      "IsModifiable": true,  
      "ApplyMethod": "pending-reboot",  
      "Source": "system",  
      "ApplyType": "dynamic",  
      "AllowedValues": "enabled,disabled",  
      "Description": "Enables auditing on cluster.",  
      "ParameterValue": "disabled"  
    },  
    {  
      "DataType": "string",  
      "ParameterName": "tls",  
      "IsModifiable": true,  
      "ApplyMethod": "pending-reboot",  
      "Source": "system",  
      "ApplyType": "static",  
      "AllowedValues": "disabled,enabled",  
      "Description": "Config to enable/disable TLS",  
      "ParameterValue": "enabled"  
    },  
    {  
      "DataType": "string",  
      "ParameterName": "ttl_monitor",  
      "IsModifiable": true,  
      "ApplyMethod": "pending-reboot",  
      "Source": "user",  
      "ApplyType": "dynamic",  
      "AllowedValues": "disabled,enabled",  
      "Description": "Enables TTL Monitoring",  
      "ParameterValue": "enabled"  
    }  
  ]  
}
```

For more information, see [Viewing Amazon DocumentDB Cluster Parameters](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [DescribeDbClusterParameters](#) in *AWS CLI Command Reference*.

describe-db-cluster-snapshot-attributes

The following code example shows how to use `describe-db-cluster-snapshot-attributes`.

AWS CLI

To list an Amazon DocumentDB snapshot attribute names and values

The following `describe-db-cluster-snapshot-attributes` example lists the attribute names and values for the Amazon DocumentDB snapshot `sample-cluster-snapshot`.

```
aws docdb describe-db-cluster-snapshot-attributes \
  --db-cluster-snapshot-identifier sample-cluster-snapshot
```

Output:

```
{
  "DBClusterSnapshotAttributesResult": {
    "DBClusterSnapshotAttributes": [
      {
        "AttributeName": "restore",
        "AttributeValues": []
      }
    ],
    "DBClusterSnapshotIdentifier": "sample-cluster-snapshot"
  }
}
```

For more information, see [DescribeDBClusterSnapshotAttributes](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [DescribeDbClusterSnapshotAttributes](#) in *AWS CLI Command Reference*.

describe-db-cluster-snapshots

The following code example shows how to use `describe-db-cluster-snapshots`.

AWS CLI

To describe Amazon DocumentDB snapshots

The following `describe-db-cluster-snapshots` example displays details for the Amazon DocumentDB snapshot `sample-cluster-snapshot`.

```
aws docdb describe-db-cluster-snapshots \  
  --db-cluster-snapshot-identifier sample-cluster-snapshot
```

Output:

```
{  
  "DBClusterSnapshots": [  
    {  
      "AvailabilityZones": [  
        "us-west-2a",  
        "us-west-2b",  
        "us-west-2c",  
        "us-west-2d"  
      ],  
      "Status": "available",  
      "DBClusterSnapshotArn": "arn:aws:rds:us-west-2:123456789012:cluster-  
snapshot:sample-cluster-snapshot",  
      "SnapshotCreateTime": "2019-03-15T20:41:26.515Z",  
      "SnapshotType": "manual",  
      "DBClusterSnapshotIdentifier": "sample-cluster-snapshot",  
      "DBClusterIdentifier": "sample-cluster",  
      "MasterUsername": "master-user",  
      "StorageEncrypted": false,  
      "VpcId": "vpc-91280df6",  
      "EngineVersion": "3.6.0",  
      "PercentProgress": 100,  
      "Port": 0,  
      "Engine": "docdb",  
      "ClusterCreateTime": "2019-03-15T20:29:58.836Z"  
    }  
  ]  
}
```

For more information, see [DescribeDBClusterSnapshots](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [DescribeDbClusterSnapshots](#) in *AWS CLI Command Reference*.

describe-db-clusters

The following code example shows how to use `describe-db-clusters`.

AWS CLI

To get detailed information about one or more Amazon DocumentDB clusters.

The following `describe-db-clusters` example displays details for the Amazon DocumentDB cluster `sample-cluster`. By omitting the `--db-cluster-identifier` parameter you can get information of up to 100 clusters.

```
aws docdb describe-db-clusters
  --db-cluster-identifier sample-cluster
```

Output:

```
{
  "DBClusters": [
    {
      "DBClusterParameterGroup": "default.docdb3.6",
      "Endpoint": "sample-cluster.cluster-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
      "PreferredBackupWindow": "00:00-00:30",
      "DBClusterIdentifier": "sample-cluster",
      "ClusterCreateTime": "2019-03-15T20:29:58.836Z",
      "LatestRestorableTime": "2019-03-18T20:28:03.239Z",
      "MasterUsername": "master-user",
      "DBClusterMembers": [
        {
          "PromotionTier": 1,
          "DBClusterParameterGroupStatus": "in-sync",
          "IsClusterWriter": false,
          "DBInstanceIdentifier": "sample-cluster"
        },
        {
          "PromotionTier": 1,
          "DBClusterParameterGroupStatus": "in-sync",
          "IsClusterWriter": true,
          "DBInstanceIdentifier": "sample-cluster2"
        }
      ]
    }
  ]
}
```

```

    }
  ],
  "PreferredMaintenanceWindow": "sat:04:30-sat:05:00",
  "VpcSecurityGroups": [
    {
      "VpcSecurityGroupId": "sg-77186e0d",
      "Status": "active"
    }
  ],
  "Engine": "docdb",
  "ReaderEndpoint": "sample-cluster.cluster-ro-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
  "DBSubnetGroup": "default",
  "MultiAZ": true,
  "AvailabilityZones": [
    "us-west-2a",
    "us-west-2c",
    "us-west-2b"
  ],
  "EarliestRestorableTime": "2019-03-15T20:30:47.020Z",
  "DbClusterResourceId": "cluster-UP4EF2PVDDFVHHDJQTYDAIGHLE",
  "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-
cluster",
  "BackupRetentionPeriod": 3,
  "HostedZoneId": "ZNKXH85TT8WWW",
  "StorageEncrypted": false,
  "EnabledCloudwatchLogsExports": [
    "audit"
  ],
  "AssociatedRoles": [],
  "EngineVersion": "3.6.0",
  "Port": 27017,
  "Status": "available"
}
]
}

```

For more information, see [Describing Amazon DocumentDB Clusters](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [DescribeDbClusters](#) in *AWS CLI Command Reference*.

describe-db-engine-versions

The following code example shows how to use `describe-db-engine-versions`.

AWS CLI

To list available Amazon DocumentDB engine versions

The following `describe-db-engine-versions` example lists all available Amazon DocumentDB engine versions.

```
aws docdb describe-db-engine-versions \  
  --engine docdb
```

Output:

```
{  
  "DBEngineVersions": [  
    {  
      "DBEngineVersionDescription": "DocDB version 1.0.200837",  
      "DBParameterGroupFamily": "docdb3.6",  
      "EngineVersion": "3.6.0",  
      "ValidUpgradeTarget": [],  
      "DBEngineDescription": "Amazon DocumentDB (with MongoDB compatibility)",  
      "SupportsLogExportsToCloudwatchLogs": true,  
      "Engine": "docdb",  
      "ExportableLogTypes": [  
        "audit"  
      ]  
    }  
  ]  
}
```

For more information, see [DescribeDBEngineVersions](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [DescribeDbEngineVersions](#) in *AWS CLI Command Reference*.

describe-db-instances

The following code example shows how to use `describe-db-instances`.

AWS CLI

To find information about provisioned Amazon DocumentDB instances

The following `describe-db-instances` example displays details for about the Amazon DocumentDB instance `sample-cluster-instance`. By omitting the `--db-instance-identifier` parameter you get information on up to 100 instances.

```
aws docdb describe-db-instances \  
  --db-instance-identifier sample-cluster-instance
```

Output:

```
{  
  "DBInstances": [  
    {  
      "Endpoint": {  
        "HostedZoneId": "ZNKXH85TT8WVW",  
        "Address": "sample-cluster-instance.corcjozrlsfc.us-  
west-2.docdb.amazonaws.com",  
        "Port": 27017  
      },  
      "PreferredBackupWindow": "00:00-00:30",  
      "DBInstanceStatus": "available",  
      "DBInstanceClass": "db.r4.large",  
      "EnabledCloudwatchLogsExports": [  
        "audit"  
      ],  
      "DBInstanceIdentifier": "sample-cluster-instance",  
      "DBSubnetGroup": {  
        "Subnets": [  
          {  
            "SubnetStatus": "Active",  
            "SubnetIdentifier": "subnet-4e26d263",  
            "SubnetAvailabilityZone": {  
              "Name": "us-west-2a"  
            }  
          },  
          {  
            "SubnetStatus": "Active",  
            "SubnetIdentifier": "subnet-afc329f4",  
            "SubnetAvailabilityZone": {  
              "Name": "us-west-2c"  
            }  
          }  
        ]  
      }  
    }  
  ]  
}
```

```

    }
  },
  {
    "SubnetStatus": "Active",
    "SubnetIdentifier": "subnet-53ab3636",
    "SubnetAvailabilityZone": {
      "Name": "us-west-2d"
    }
  },
  {
    "SubnetStatus": "Active",
    "SubnetIdentifier": "subnet-991cb8d0",
    "SubnetAvailabilityZone": {
      "Name": "us-west-2b"
    }
  }
],
"DBSubnetGroupName": "default",
"SubnetGroupStatus": "Complete",
"DBSubnetGroupDescription": "default",
"VpcId": "vpc-91280df6"
},
"InstanceCreateTime": "2019-03-15T20:36:06.338Z",
"Engine": "docdb",
"StorageEncrypted": false,
"AutoMinorVersionUpgrade": true,
"DBInstanceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster-
instance",
"PreferredMaintenanceWindow": "tue:08:39-tue:09:09",
"VpcSecurityGroups": [
  {
    "Status": "active",
    "VpcSecurityGroupId": "sg-77186e0d"
  }
],
"DBClusterIdentifier": "sample-cluster",
"PendingModifiedValues": {},
"BackupRetentionPeriod": 3,
"PubliclyAccessible": false,
"EngineVersion": "3.6.0",
"PromotionTier": 1,
"AvailabilityZone": "us-west-2c",
"DbiResourceId": "db-A2GIKUV6KPOHITGGKI2NHVISZA"
}

```



```
]
}
```

For more information, see [Describing Amazon DocumentDB Instances](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [DescribeDbInstances](#) in *AWS CLI Command Reference*.

describe-db-subnet-groups

The following code example shows how to use `describe-db-subnet-groups`.

AWS CLI

To retrieve a list of Amazon DocumentDB subnet descriptions

The following `describe-db-subnet-groups` example describes details for the Amazon DocumentDB subnet named `default`.

```
aws docdb describe-db-subnet-groups \
  --db-subnet-group-name default
```

Output:

```
{
  "DBSubnetGroups": [
    {
      "VpcId": "vpc-91280df6",
      "DBSubnetGroupArn": "arn:aws:rds:us-west-2:123456789012:subgrp:default",
      "Subnets": [
        {
          "SubnetIdentifier": "subnet-4e26d263",
          "SubnetStatus": "Active",
          "SubnetAvailabilityZone": {
            "Name": "us-west-2a"
          }
        },
        {
          "SubnetIdentifier": "subnet-afc329f4",
          "SubnetStatus": "Active",
          "SubnetAvailabilityZone": {
```

```

        "Name": "us-west-2c"
      }
    },
    {
      "SubnetIdentifier": "subnet-53ab3636",
      "SubnetStatus": "Active",
      "SubnetAvailabilityZone": {
        "Name": "us-west-2d"
      }
    },
    {
      "SubnetIdentifier": "subnet-991cb8d0",
      "SubnetStatus": "Active",
      "SubnetAvailabilityZone": {
        "Name": "us-west-2b"
      }
    }
  ],
  "DBSubnetGroupName": "default",
  "SubnetGroupStatus": "Complete",
  "DBSubnetGroupDescription": "default"
}
]
}

```

For more information, see [Describing Subnet Groups](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [DescribeDbSubnetGroups](#) in *AWS CLI Command Reference*.

describe-engine-default-cluster-parameters

The following code example shows how to use `describe-engine-default-cluster-parameters`.

AWS CLI

To describe the default engine and system parameter information for Amazon DocumentDB

The following `describe-engine-default-cluster-parameters` example displays details for the default engine and system parameter information for the Amazon DocumentDB parameter group `docdb3.6`.

```
aws docdb describe-engine-default-cluster-parameters \  
--db-parameter-group-family docdb3.6
```

Output:

```
{  
  "EngineDefaults": {  
    "DBParameterGroupFamily": "docdb3.6",  
    "Parameters": [  
      {  
        "ApplyType": "dynamic",  
        "ParameterValue": "disabled",  
        "Description": "Enables auditing on cluster.",  
        "Source": "system",  
        "DataType": "string",  
        "MinimumEngineVersion": "3.6.0",  
        "AllowedValues": "enabled,disabled",  
        "ParameterName": "audit_logs",  
        "IsModifiable": true  
      },  
      {  
        "ApplyType": "static",  
        "ParameterValue": "enabled",  
        "Description": "Config to enable/disable TLS",  
        "Source": "system",  
        "DataType": "string",  
        "MinimumEngineVersion": "3.6.0",  
        "AllowedValues": "disabled,enabled",  
        "ParameterName": "tls",  
        "IsModifiable": true  
      },  
      {  
        "ApplyType": "dynamic",  
        "ParameterValue": "enabled",  
        "Description": "Enables TTL Monitoring",  
        "Source": "system",  
        "DataType": "string",  
        "MinimumEngineVersion": "3.6.0",  
        "AllowedValues": "disabled,enabled",  
        "ParameterName": "ttl_monitor",  
        "IsModifiable": true  
      }  
    ]  
  }  
}
```

```
}  
}
```

For more information, see [DescribeEngineDefaultClusterParameters](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [DescribeEngineDefaultClusterParameters](#) in *AWS CLI Command Reference*.

describe-event-categories

The following code example shows how to use `describe-event-categories`.

AWS CLI

To describe all Amazon DocumentDB event categories

The following `describe-event-categories` example lists all categories for the Amazon DocumentDB event source type `db-instance`.

```
aws docdb describe-event-categories \  
  --source-type db-cluster
```

Output:

```
{  
  "EventCategoriesMapList": [  
    {  
      "SourceType": "db-cluster",  
      "EventCategories": [  
        "failover",  
        "maintenance",  
        "notification",  
        "failure"  
      ]  
    }  
  ]  
}
```

For more information, see [Viewing Event Categories](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [DescribeEventCategories](#) in *AWS CLI Command Reference*.

describe-events

The following code example shows how to use describe-events.

AWS CLI

To list Amazon DocumentDB events

The following describe-events example list all the Amazon DocumentDB events for the last 24 hours (1440 minutes).

```
aws docdb describe-events \  
  --duration 1440
```

This command produces no output. Output:

```
{  
  "Events": [  
    {  
      "EventCategories": [  
        "failover"  
      ],  
      "Message": "Started cross AZ failover to DB instance: sample-cluster",  
      "Date": "2019-03-18T21:36:29.807Z",  
      "SourceArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-  
cluster",  
      "SourceIdentifier": "sample-cluster",  
      "SourceType": "db-cluster"  
    },  
    {  
      "EventCategories": [  
        "availability"  
      ],  
      "Message": "DB instance restarted",  
      "Date": "2019-03-18T21:36:40.793Z",  
      "SourceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster",  
      "SourceIdentifier": "sample-cluster",  
      "SourceType": "db-instance"  
    },  
    {  
      "EventCategories": [],  
      "Message": "A new writer was promoted. Restarting database as a  
reader.",
```

```

    "Date": "2019-03-18T21:36:43.873Z",
    "SourceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
    "SourceIdentifier": "sample-cluster2",
    "SourceType": "db-instance"
  },
  {
    "EventCategories": [
      "availability"
    ],
    "Message": "DB instance restarted",
    "Date": "2019-03-18T21:36:51.257Z",
    "SourceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
    "SourceIdentifier": "sample-cluster2",
    "SourceType": "db-instance"
  },
  {
    "EventCategories": [
      "failover"
    ],
    "Message": "Completed failover to DB instance: sample-cluster",
    "Date": "2019-03-18T21:36:53.462Z",
    "SourceArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-
cluster",
    "SourceIdentifier": "sample-cluster",
    "SourceType": "db-cluster"
  },
  {
    "Date": "2019-03-19T16:51:48.847Z",
    "EventCategories": [
      "configuration change"
    ],
    "Message": "Updated parameter audit_logs to enabled with apply method
pending-reboot",
    "SourceIdentifier": "custom3-6-param-grp",
    "SourceType": "db-parameter-group"
  },
  {
    "EventCategories": [
      "configuration change"
    ],
    "Message": "Applying modification to database instance class",
    "Date": "2019-03-19T17:55:20.095Z",
    "SourceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
    "SourceIdentifier": "sample-cluster2",

```

```
    "SourceType": "db-instance"
  },
  {
    "EventCategories": [
      "availability"
    ],
    "Message": "DB instance shutdown",
    "Date": "2019-03-19T17:56:31.127Z",
    "SourceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
    "SourceIdentifier": "sample-cluster2",
    "SourceType": "db-instance"
  },
  {
    "EventCategories": [
      "configuration change"
    ],
    "Message": "Finished applying modification to DB instance class",
    "Date": "2019-03-19T18:00:45.822Z",
    "SourceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
    "SourceIdentifier": "sample-cluster2",
    "SourceType": "db-instance"
  },
  {
    "EventCategories": [
      "availability"
    ],
    "Message": "DB instance restarted",
    "Date": "2019-03-19T18:00:53.397Z",
    "SourceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
    "SourceIdentifier": "sample-cluster2",
    "SourceType": "db-instance"
  },
  {
    "EventCategories": [
      "availability"
    ],
    "Message": "DB instance shutdown",
    "Date": "2019-03-19T18:23:36.045Z",
    "SourceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
    "SourceIdentifier": "sample-cluster2",
    "SourceType": "db-instance"
  },
  {
    "EventCategories": [
```

```

        "availability"
    ],
    "Message": "DB instance restarted",
    "Date": "2019-03-19T18:23:46.209Z",
    "SourceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
    "SourceIdentifier": "sample-cluster2",
    "SourceType": "db-instance"
},
{
    "Date": "2019-03-19T18:39:05.822Z",
    "EventCategories": [
        "configuration change"
    ],
    "Message": "Updated parameter ttl_monitor to enabled with apply method
immediate",
    "SourceIdentifier": "custom3-6-param-grp",
    "SourceType": "db-parameter-group"
},
{
    "Date": "2019-03-19T18:39:48.067Z",
    "EventCategories": [
        "configuration change"
    ],
    "Message": "Updated parameter audit_logs to disabled with apply method
immediate",
    "SourceIdentifier": "custom3-6-param-grp",
    "SourceType": "db-parameter-group"
}
]
}

```

For more information, see [Viewing Amazon DocumentDB Events](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [DescribeEvents](#) in *AWS CLI Command Reference*.

describe-orderable-db-instance-options

The following code example shows how to use `describe-orderable-db-instance-options`.

AWS CLI

To find the Amazon DocumentDB instance options you can order

The following `describe-orderable-db-instance-options` example lists all instance options for Amazon DocumentDB for a region.

```
aws docdb describe-orderable-db-instance-options \  
  --engine docdb \  
  --region us-east-1
```

Output:

```
{  
  "OrderableDBInstanceOptions": [  
    {  
      "Vpc": true,  
      "AvailabilityZones": [  
        {  
          "Name": "us-east-1a"  
        },  
        {  
          "Name": "us-east-1b"  
        },  
        {  
          "Name": "us-east-1c"  
        },  
        {  
          "Name": "us-east-1d"  
        }  
      ],  
      "EngineVersion": "3.6.0",  
      "DBInstanceClass": "db.r4.16xlarge",  
      "LicenseModel": "na",  
      "Engine": "docdb"  
    },  
    {  
      "Vpc": true,  
      "AvailabilityZones": [  
        {  
          "Name": "us-east-1a"  
        },  
        {  
          "Name": "us-east-1b"  
        },  
        {  
          "Name": "us-east-1c"  
        }  
      ]  
    }  
  ]  
}
```

```
    },
    {
      "Name": "us-east-1d"
    }
  ],
  "EngineVersion": "3.6.0",
  "DBInstanceClass": "db.r4.2xlarge",
  "LicenseModel": "na",
  "Engine": "docdb"
},
{
  "Vpc": true,
  "AvailabilityZones": [
    {
      "Name": "us-east-1a"
    },
    {
      "Name": "us-east-1b"
    },
    {
      "Name": "us-east-1c"
    },
    {
      "Name": "us-east-1d"
    }
  ],
  "EngineVersion": "3.6.0",
  "DBInstanceClass": "db.r4.4xlarge",
  "LicenseModel": "na",
  "Engine": "docdb"
},
{
  "Vpc": true,
  "AvailabilityZones": [
    {
      "Name": "us-east-1a"
    },
    {
      "Name": "us-east-1b"
    },
    {
      "Name": "us-east-1c"
    }
  ],
```

```
        {
            "Name": "us-east-1d"
        }
    ],
    "EngineVersion": "3.6.0",
    "DBInstanceClass": "db.r4.8xlarge",
    "LicenseModel": "na",
    "Engine": "docdb"
},
{
    "Vpc": true,
    "AvailabilityZones": [
        {
            "Name": "us-east-1a"
        },
        {
            "Name": "us-east-1b"
        },
        {
            "Name": "us-east-1c"
        },
        {
            "Name": "us-east-1d"
        }
    ],
    "EngineVersion": "3.6.0",
    "DBInstanceClass": "db.r4.large",
    "LicenseModel": "na",
    "Engine": "docdb"
},
{
    "Vpc": true,
    "AvailabilityZones": [
        {
            "Name": "us-east-1a"
        },
        {
            "Name": "us-east-1b"
        },
        {
            "Name": "us-east-1c"
        },
        {
            "Name": "us-east-1d"
        }
    ]
}
```

```
    }
  ],
  "EngineVersion": "3.6.0",
  "DBInstanceClass": "db.r4.xlarge",
  "LicenseModel": "na",
  "Engine": "docdb"
}
]
```

For more information, see [Adding an Amazon DocumentDB Instance to a Cluster](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [DescribeOrderableDbInstanceOptions](#) in *AWS CLI Command Reference*.

describe-pending-maintenance-actions

The following code example shows how to use `describe-pending-maintenance-actions`.

AWS CLI

To list your pending Amazon DocumentDB maintenance actions

The following `describe-pending-maintenance-actions` example lists all your pending Amazon DocumentDB maintenance actions.

```
aws docdb describe-pending-maintenance-actions
```

Output:

```
{
  "PendingMaintenanceActions": []
}
```

For more information, see [Maintaining Amazon DocumentDB](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [DescribePendingMaintenanceActions](#) in *AWS CLI Command Reference*.

failover-db-cluster

The following code example shows how to use `failover-db-cluster`.

AWS CLI

To force an Amazon DocumentDB cluster to failover to a replica

The following `failover-db-cluster` example causes the primary instance in the Amazon DocumentDB cluster `sample-cluster` to failover to a replica.

```
aws docdb failover-db-cluster \  
  --db-cluster-identifier sample-cluster
```

Output:

```
{  
  "DBCluster": {  
    "AssociatedRoles": [],  
    "DBClusterIdentifier": "sample-cluster",  
    "EngineVersion": "3.6.0",  
    "DBSubnetGroup": "default",  
    "MasterUsername": "master-user",  
    "EarliestRestorableTime": "2019-03-15T20:30:47.020Z",  
    "Endpoint": "sample-cluster.cluster-corcjozrlsfc.us-  
west-2.docdb.amazonaws.com",  
    "AvailabilityZones": [  
      "us-west-2a",  
      "us-west-2c",  
      "us-west-2b"  
    ],  
    "LatestRestorableTime": "2019-03-18T21:35:23.548Z",  
    "PreferredMaintenanceWindow": "sat:04:30-sat:05:00",  
    "PreferredBackupWindow": "00:00-00:30",  
    "Port": 27017,  
    "VpcSecurityGroups": [  
      {  
        "VpcSecurityGroupId": "sg-77186e0d",  
        "Status": "active"  
      }  
    ],  
    "StorageEncrypted": false,  
    "ClusterCreateTime": "2019-03-15T20:29:58.836Z",  
    "MultiAZ": true,  
    "Status": "available",  
    "DBClusterMembers": [  
      {
```

```

        "DBClusterParameterGroupStatus": "in-sync",
        "IsClusterWriter": false,
        "DBInstanceIdentifier": "sample-cluster",
        "PromotionTier": 1
    },
    {
        "DBClusterParameterGroupStatus": "in-sync",
        "IsClusterWriter": true,
        "DBInstanceIdentifier": "sample-cluster2",
        "PromotionTier": 2
    }
],
"EnabledCloudwatchLogsExports": [
    "audit"
],
"DBClusterParameterGroup": "default.docdb3.6",
"HostedZoneId": "ZNKXH85TT8WVW",
"DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster",
"BackupRetentionPeriod": 3,
"DbClusterResourceId": "cluster-UP4EF2PVDDFVHHDJQTYDAIGHLE",
"ReaderEndpoint": "sample-cluster.cluster-ro-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
"Engine": "docdb"
}
}

```

For more information, see [Amazon DocumentDB Failover](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [FailoverDbCluster](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list all the tags on an Amazon DocumentDB resource

The following `list-tags-for-resource` example lists all tags on the Amazon DocumentDB cluster `sample-cluster`.

```
aws docdb list-tags-for-resource \
```

```
--resource-name arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster
```

Output:

```
{
  "TagList": [
    {
      "Key": "A",
      "Value": "ALPHA"
    },
    {
      "Key": "B",
      "Value": ""
    },
    {
      "Key": "C",
      "Value": "CHARLIE"
    }
  ]
}
```

For more information, see [Listing Tags on an Amazon DocumentDB Resource](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

modify-db-cluster-parameter-group

The following code example shows how to use `modify-db-cluster-parameter-group`.

AWS CLI

To modify an Amazon DocumentDB DB cluster parameter group

The following `modify-db-cluster-parameter-group` example modifies the Amazon DocumentDB cluster parameter group `custom3-6-param-grp` by setting the two parameters `audit_logs` and `ttl_monitor` to enabled. The changes are applied at the next reboot.

```
aws docdb modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name custom3-6-param-grp \
  --parameters
  ParameterName=audit_logs,ParameterValue=enabled,ApplyMethod=pending-reboot \
```

```
ParameterName=ttl_monitor,ParameterValue=enabled,ApplyMethod=pending-reboot
```

Output:

```
{
  "DBClusterParameterGroupName": "custom3-6-param-grp"
}
```

For more information, see [Modifying an Amazon DocumentDB Cluster Parameter Group](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [ModifyDbClusterParameterGroup](#) in *AWS CLI Command Reference*.

modify-db-cluster-snapshot-attribute

The following code example shows how to use `modify-db-cluster-snapshot-attribute`.

AWS CLI**Example 1: To add an attribute to an Amazon DocumentDB snapshot**

The following `modify-db-cluster-snapshot-attribute` example adds four attribute values to an Amazon DocumentDB cluster snapshot.

```
aws docdb modify-db-cluster-snapshot-attribute \
  --db-cluster-snapshot-identifier sample-cluster-snapshot \
  --attribute-name restore \
  --values-to-add all 123456789011 123456789012 123456789013
```

Output:

```
{
  "DBClusterSnapshotAttributesResult": {
    "DBClusterSnapshotAttributes": [
      {
        "AttributeName": "restore",
        "AttributeValues": [
          "all",
          "123456789011",
          "123456789012",
          "123456789013"
        ]
      }
    ]
  }
}
```



```

    ]
  },
  "DBClusterSnapshotIdentifier": "sample-cluster-snapshot"
}

```

Example 2: To remove attributes from an Amazon DocumentDB snapshot

The following `modify-db-cluster-snapshot-attribute` example removes two attribute values from an Amazon DocumentDB cluster snapshot.

```

aws docdb modify-db-cluster-snapshot-attribute \
  --db-cluster-snapshot-identifier sample-cluster-snapshot \
  --attribute-name restore \
  --values-to-remove 123456789012 all

```

Output:

```

{
  "DBClusterSnapshotAttributesResult": {
    "DBClusterSnapshotAttributes": [
      {
        "AttributeName": "restore",
        "AttributeValues": [
          "123456789011",
          "123456789013"
        ]
      }
    ],
    "DBClusterSnapshotIdentifier": "sample-cluster-snapshot"
  }
}

```

For more information, see [ModifyDBClusterSnapshotAttribute](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [ModifyDbClusterSnapshotAttribute](#) in *AWS CLI Command Reference*.

modify-db-cluster

The following code example shows how to use `modify-db-cluster`.

AWS CLI

To modify an Amazon DocumentDB cluster

The following `modify-db-cluster` example modifies the Amazon DocumentDB cluster `sample-cluster` by making the retention period for automatic backups 7 days, and changing the preferred windows for both backups and maintenance. All changes are applied at the next maintenance window.

```
aws docdb modify-db-cluster \  
  --db-cluster-identifier sample-cluster \  
  --no-apply-immediately \  
  --backup-retention-period 7 \  
  --preferred-backup-window 18:00-18:30 \  
  --preferred-maintenance-window sun:20:00-sun:20:30
```

Output:

```
{  
  "DBCluster": {  
    "Endpoint": "sample-cluster.cluster-corcjozrlsfc.us-  
west-2.docdb.amazonaws.com",  
    "DBClusterMembers": [  
      {  
        "DBClusterParameterGroupStatus": "in-sync",  
        "DBInstanceIdentifier": "sample-cluster",  
        "IsClusterWriter": true,  
        "PromotionTier": 1  
      },  
      {  
        "DBClusterParameterGroupStatus": "in-sync",  
        "DBInstanceIdentifier": "sample-cluster2",  
        "IsClusterWriter": false,  
        "PromotionTier": 2  
      }  
    ],  
    "HostedZoneId": "ZNKXH85TT8WVW",  
    "StorageEncrypted": false,  
    "PreferredBackupWindow": "18:00-18:30",  
    "MultiAZ": true,  
    "EngineVersion": "3.6.0",  
    "MasterUsername": "master-user",
```

```

    "ReaderEndpoint": "sample-cluster.cluster-ro-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
    "DBSubnetGroup": "default",
    "LatestRestorableTime": "2019-03-18T22:08:13.408Z",
    "EarliestRestorableTime": "2019-03-15T20:30:47.020Z",
    "PreferredMaintenanceWindow": "sun:20:00-sun:20:30",
    "AssociatedRoles": [],
    "EnabledCloudwatchLogsExports": [
        "audit"
    ],
    "Engine": "docdb",
    "DBClusterParameterGroup": "default.docdb3.6",
    "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster",
    "BackupRetentionPeriod": 7,
    "DBClusterIdentifier": "sample-cluster",
    "AvailabilityZones": [
        "us-west-2a",
        "us-west-2c",
        "us-west-2b"
    ],
    "Status": "available",
    "DbClusterResourceId": "cluster-UP4EF2PVDDFVHHDJQTYDAIGHLE",
    "ClusterCreateTime": "2019-03-15T20:29:58.836Z",
    "VpcSecurityGroups": [
        {
            "VpcSecurityGroupId": "sg-77186e0d",
            "Status": "active"
        }
    ],
    "Port": 27017
}
}

```

For more information, see [Modifying an Amazon DocumentDB Cluster](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [ModifyDbCluster](#) in *AWS CLI Command Reference*.

modify-db-instance

The following code example shows how to use `modify-db-instance`.

AWS CLI

To modify an Amazon DocumentDB instance

The following `modify-db-instance` example modifies the Amazon DocumentDB instance `sample-cluster2` by changing its instance class to `db.r4.4xlarge` and its promotion tier to 5. The changes are applied immediately but can only be seen after the instances status is available.

```
aws docdb modify-db-instance \  
  --db-instance-identifier sample-cluster2 \  
  --apply-immediately \  
  --db-instance-class db.r4.4xlarge \  
  --promotion-tier 5
```

Output:

```
{  
  "DBInstance": {  
    "EngineVersion": "3.6.0",  
    "StorageEncrypted": false,  
    "DBInstanceClass": "db.r4.large",  
    "PreferredMaintenanceWindow": "mon:08:39-mon:09:09",  
    "AutoMinorVersionUpgrade": true,  
    "VpcSecurityGroups": [  
      {  
        "VpcSecurityGroupId": "sg-77186e0d",  
        "Status": "active"  
      }  
    ],  
    "PreferredBackupWindow": "18:00-18:30",  
    "EnabledCloudwatchLogsExports": [  
      "audit"  
    ],  
    "AvailabilityZone": "us-west-2f",  
    "DBInstanceIdentifier": "sample-cluster2",  
    "InstanceCreateTime": "2019-03-15T20:36:06.338Z",  
    "Engine": "docdb",  
    "BackupRetentionPeriod": 7,  
    "DBSubnetGroup": {  
      "DBSubnetGroupName": "default",  
      "DBSubnetGroupDescription": "default",  
      "SubnetGroupStatus": "Complete",
```

```
    "Subnets": [
      {
        "SubnetIdentifier": "subnet-4e26d263",
        "SubnetAvailabilityZone": {
          "Name": "us-west-2a"
        },
        "SubnetStatus": "Active"
      },
      {
        "SubnetIdentifier": "subnet-afc329f4",
        "SubnetAvailabilityZone": {
          "Name": "us-west-2c"
        },
        "SubnetStatus": "Active"
      },
      {
        "SubnetIdentifier": "subnet-53ab3636",
        "SubnetAvailabilityZone": {
          "Name": "us-west-2d"
        },
        "SubnetStatus": "Active"
      },
      {
        "SubnetIdentifier": "subnet-991cb8d0",
        "SubnetAvailabilityZone": {
          "Name": "us-west-2b"
        },
        "SubnetStatus": "Active"
      }
    ],
    "VpcId": "vpc-91280df6"
  },
  "PromotionTier": 2,
  "Endpoint": {
    "Address": "sample-cluster2.corcjozrlsfc.us-west-2.docdb.amazonaws.com",
    "HostedZoneId": "ZNKXH85TT8WWW",
    "Port": 27017
  },
  "DbiResourceId": "db-A2GIKUV6KPOHITGGKI2NHVISZA",
  "DBClusterIdentifier": "sample-cluster",
  "DBInstanceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
  "PendingModifiedValues": {
    "DBInstanceClass": "db.r4.4xlarge"
  }
},
```

```
    "PubliclyAccessible": false,  
    "DBInstanceStatus": "available"  
  }  
}
```

For more information, see [Modifying an Amazon DocumentDB Instance](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [ModifyDbInstance](#) in *AWS CLI Command Reference*.

modify-db-subnet-group

The following code example shows how to use `modify-db-subnet-group`.

AWS CLI

To modify an Amazon DocumentDB subnet group

The following `modify-db-subnet-group` example modifies the subnet group `sample-subnet-group` by adding the specified subnets and a new description.

```
aws docdb modify-db-subnet-group \  
  --db-subnet-group-name sample-subnet-group \  
  --subnet-ids subnet-b3806e8f subnet-53ab3636 subnet-991cb8d0 \  
  --db-subnet-group-description "New subnet description"
```

Output:

```
{  
  "DBSubnetGroup": {  
    "DBSubnetGroupName": "sample-subnet-group",  
    "SubnetGroupStatus": "Complete",  
    "DBSubnetGroupArn": "arn:aws:rds:us-west-2:123456789012:subgrp:sample-  
subnet-group",  
    "VpcId": "vpc-91280df6",  
    "DBSubnetGroupDescription": "New subnet description",  
    "Subnets": [  
      {  
        "SubnetIdentifier": "subnet-b3806e8f",  
        "SubnetStatus": "Active",  
        "SubnetAvailabilityZone": {  
          "Name": "us-west-2a"  
        }  
      }  
    ]  
  }  
}
```

```

    },
    {
      "SubnetIdentifier": "subnet-53ab3636",
      "SubnetStatus": "Active",
      "SubnetAvailabilityZone": {
        "Name": "us-west-2c"
      }
    },
    {
      "SubnetIdentifier": "subnet-991cb8d0",
      "SubnetStatus": "Active",
      "SubnetAvailabilityZone": {
        "Name": "us-west-2b"
      }
    }
  ]
}

```

For more information, see [Modifying an Amazon DocumentDB Subnet Group](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [ModifyDbSubnetGroup](#) in *AWS CLI Command Reference*.

reboot-db-instance

The following code example shows how to use `reboot-db-instance`.

AWS CLI

To reboot an Amazon DocumentDB instance

The following `reboot-db-instance` example reboots the Amazon DocumentDB instance `sample-cluster2`.

```
aws docdb reboot-db-instance \
  --db-instance-identifier sample-cluster2
```

This command produces no output. Output:

```
{
  "DBInstance": {
    "PreferredBackupWindow": "18:00-18:30",
```

```
"DBInstanceIdentifier": "sample-cluster2",
"VpcSecurityGroups": [
  {
    "Status": "active",
    "VpcSecurityGroupId": "sg-77186e0d"
  }
],
"DBSubnetGroup": {
  "VpcId": "vpc-91280df6",
  "Subnets": [
    {
      "SubnetStatus": "Active",
      "SubnetAvailabilityZone": {
        "Name": "us-west-2a"
      },
      "SubnetIdentifier": "subnet-4e26d263"
    },
    {
      "SubnetStatus": "Active",
      "SubnetAvailabilityZone": {
        "Name": "us-west-2c"
      },
      "SubnetIdentifier": "subnet-afc329f4"
    },
    {
      "SubnetStatus": "Active",
      "SubnetAvailabilityZone": {
        "Name": "us-west-2d"
      },
      "SubnetIdentifier": "subnet-53ab3636"
    },
    {
      "SubnetStatus": "Active",
      "SubnetAvailabilityZone": {
        "Name": "us-west-2b"
      },
      "SubnetIdentifier": "subnet-991cb8d0"
    }
  ],
  "SubnetGroupStatus": "Complete",
  "DBSubnetGroupName": "default",
  "DBSubnetGroupDescription": "default"
},
"PendingModifiedValues": {},
```



```

    "Endpoint": {
      "Address": "sample-cluster2.corcjozrlsfc.us-west-2.docdb.amazonaws.com",
      "HostedZoneId": "ZNKXH85TT8WVW",
      "Port": 27017
    },
    "EnabledCloudwatchLogsExports": [
      "audit"
    ],
    "StorageEncrypted": false,
    "DbiResourceId": "db-A2GIKUV6KPOHITGGKI2NHVISZA",
    "AutoMinorVersionUpgrade": true,
    "Engine": "docdb",
    "InstanceCreateTime": "2019-03-15T20:36:06.338Z",
    "EngineVersion": "3.6.0",
    "PromotionTier": 5,
    "BackupRetentionPeriod": 7,
    "DBClusterIdentifier": "sample-cluster",
    "PreferredMaintenanceWindow": "mon:08:39-mon:09:09",
    "PubliclyAccessible": false,
    "DBInstanceClass": "db.r4.4xlarge",
    "AvailabilityZone": "us-west-2d",
    "DBInstanceArn": "arn:aws:rds:us-west-2:123456789012:db:sample-cluster2",
    "DBInstanceStatus": "rebooting"
  }
}

```

For more information, see [Rebooting an Amazon DocumentDB Instance](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [RebootDbInstance](#) in *AWS CLI Command Reference*.

remove-tags-from-resource

The following code example shows how to use `remove-tags-from-resource`.

AWS CLI

To remove tags from an Amazon DocumentDB resource

The following `remove-tags-from-resource` example removes the tag with the key named `B` from the Amazon DocumentDB cluster `sample-cluster`.

```
aws docdb remove-tags-from-resource \
```

```
--resource-name arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster \  
--tag-keys B
```

This command produces no output.

For more information, see [Removing Tags from an Amazon DocumentDBResource](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [RemoveTagsFromResource](#) in *AWS CLI Command Reference*.

reset-db-cluster-parameter-group

The following code example shows how to use `reset-db-cluster-parameter-group`.

AWS CLI

To reset the specified parameter value to its defaults in an Amazon DocumentDB parameter group

The following `reset-db-cluster-parameter-group` example resets the parameter `ttl_monitor` in the Amazon DocumentDB parameter group `custom3-6-param-grp` to its default value.

```
aws docdb reset-db-cluster-parameter-group \  
  --db-cluster-parameter-group-name custom3-6-param-grp \  
  --parameters ParameterName=ttl_monitor,ApplyMethod=immediate
```

Output:

```
{  
  "DBClusterParameterGroupName": "custom3-6-param-grp"  
}
```

For more information, see title in the *Amazon DocumentDB Developer Guide*.

To reset specified or all parameter values to their defaults in an Amazon DocumentDB parameter group

The following `reset-db-cluster-parameter-group` example resets all parameters in the Amazon DocumentDB parameter group `custom3-6-param-grp` to their default value.

```
aws docdb reset-db-cluster-parameter-group \  
  --db-cluster-parameter-group-name custom3-6-param-grp
```

```
--db-cluster-parameter-group-name custom3-6-param-grp \  
--reset-all-parameters
```

Output:

```
{  
  "DBClusterParameterGroupName": "custom3-6-param-grp"  
}
```

For more information, see [Resetting an Amazon DocumentDB Cluster Parameter Group](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [ResetDbClusterParameterGroup](#) in *AWS CLI Command Reference*.

restore-db-cluster-from-snapshot

The following code example shows how to use `restore-db-cluster-from-snapshot`.

AWS CLI**To restore an Amazon DocumentDB cluster from an automatic or manual snapshot**

The following `restore-db-cluster-from-snapshot` example creates a new Amazon DocumentDB cluster named `sample-cluster-2019-03-16-00-01-restored` from the snapshot `rds:sample-cluster-2019-03-16-00-01`.

```
aws docdb restore-db-cluster-from-snapshot \  
  --db-cluster-identifier sample-cluster-2019-03-16-00-01-restored \  
  --engine docdb \  
  --snapshot-identifier rds:sample-cluster-2019-03-16-00-01
```

Output:

```
{  
  "DBCluster": {  
    "ClusterCreateTime": "2019-03-19T18:45:01.857Z",  
    "HostedZoneId": "ZNKXH85TT8WVW",  
    "Engine": "docdb",  
    "DBClusterMembers": [],  
    "MultiAZ": false,  
    "AvailabilityZones": [  
      "us-west-2a",
```

```

        "us-west-2c",
        "us-west-2b"
    ],
    "StorageEncrypted": false,
    "ReaderEndpoint": "sample-cluster-2019-03-16-00-01-restored.cluster-ro-
corcjzrlsfc.us-west-2.docdb.amazonaws.com",
    "Endpoint": "sample-cluster-2019-03-16-00-01-restored.cluster-
corcjzrlsfc.us-west-2.docdb.amazonaws.com",
    "Port": 27017,
    "PreferredBackupWindow": "00:00-00:30",
    "DBSubnetGroup": "default",
    "DBClusterIdentifier": "sample-cluster-2019-03-16-00-01-restored",
    "PreferredMaintenanceWindow": "sat:04:30-sat:05:00",
    "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-
cluster-2019-03-16-00-01-restored",
    "DBClusterParameterGroup": "default.docdb3.6",
    "DbClusterResourceId": "cluster-X0046Q3RH4LWSYNH3NMZKXPISU",
    "MasterUsername": "master-user",
    "EngineVersion": "3.6.0",
    "BackupRetentionPeriod": 3,
    "AssociatedRoles": [],
    "Status": "creating",
    "VpcSecurityGroups": [
        {
            "Status": "active",
            "VpcSecurityGroupId": "sg-77186e0d"
        }
    ]
}
}

```

For more information, see [Restoring from a Cluster Snapshot](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [RestoreDbClusterFromSnapshot](#) in *AWS CLI Command Reference*.

restore-db-cluster-to-point-in-time

The following code example shows how to use `restore-db-cluster-to-point-in-time`.

AWS CLI

To restore an Amazon DocumentDB cluster to a point-in-time from a manual snapshot

The following `restore-db-cluster-to-point-in-time` example uses the `sample-cluster-snapshot` to create a new Amazon DocumentDB cluster, `sample-cluster-pit`, using the latest restorable time.

```
aws docdb restore-db-cluster-to-point-in-time \  
  --db-cluster-identifier sample-cluster-pit \  
  --source-db-cluster-identifier arn:aws:rds:us-  
west-2:123456789012:cluster:sample-cluster \  
  --use-latest-restorable-time
```

Output:

```
{  
  "DBCluster": {  
    "StorageEncrypted": false,  
    "BackupRetentionPeriod": 3,  
    "MasterUsername": "master-user",  
    "HostedZoneId": "ZNKXH85TT8WVW",  
    "PreferredBackupWindow": "00:00-00:30",  
    "MultiAZ": false,  
    "DBClusterIdentifier": "sample-cluster-pit",  
    "DBSubnetGroup": "default",  
    "ClusterCreateTime": "2019-04-03T15:55:21.320Z",  
    "AssociatedRoles": [],  
    "DBClusterParameterGroup": "default.docdb3.6",  
    "DBClusterMembers": [],  
    "Status": "creating",  
    "AvailabilityZones": [  
      "us-west-2a",  
      "us-west-2d",  
      "us-west-2b"  
    ],  
    "ReaderEndpoint": "sample-cluster-pit.cluster-ro-corcjozrlsfc.us-  
west-2.docdb.amazonaws.com",  
    "Port": 27017,  
    "Engine": "docdb",  
    "EngineVersion": "3.6.0",  
    "VpcSecurityGroups": [  
      {  
        "VpcSecurityGroupId": "sg-77186e0d",  
        "Status": "active"  
      }  
    ],  
  },  
}
```

```
    "PreferredMaintenanceWindow": "sat:04:30-sat:05:00",
    "Endpoint": "sample-cluster-pit.cluster-corcjozrlsfc.us-
west-2.docdb.amazonaws.com",
    "DbClusterResourceId": "cluster-NLCABBX0SE2QPQ4GOLZIFWEPLM",
    "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster-
pit"
  }
}
```

For more information, see [Restoring a Snapshot to a Point in Time](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [RestoreDbClusterToPointInTime](#) in *AWS CLI Command Reference*.

start-db-cluster

The following code example shows how to use `start-db-cluster`.

AWS CLI

To start a stopped Amazon DocumentDB cluster

The following `start-db-cluster` example starts the specified Amazon DocumentDB cluster.

```
aws docdb start-db-cluster \
  --db-cluster-identifier sample-cluster
```

Output:

```
{
  "DBCluster": {
    "ClusterCreateTime": "2019-03-19T18:45:01.857Z",
    "HostedZoneId": "ZNKXH85TT8WVW",
    "Engine": "docdb",
    "DBClusterMembers": [],
    "MultiAZ": false,
    "AvailabilityZones": [
      "us-east-1a",
      "us-east-1c",
      "us-east-1f"
    ],
    "StorageEncrypted": false,
```

```

    "ReaderEndpoint": "sample-cluster-2019-03-16-00-01-restored.cluster-ro-
corcjzrlsfc.us-east-1.docdb.amazonaws.com",
    "Endpoint": "sample-cluster-2019-03-16-00-01-restored.cluster-
corcjzrlsfc.us-east-1.docdb.amazonaws.com",
    "Port": 27017,
    "PreferredBackupWindow": "00:00-00:30",
    "DBSubnetGroup": "default",
    "DBClusterIdentifier": "sample-cluster-2019-03-16-00-01-restored",
    "PreferredMaintenanceWindow": "sat:04:30-sat:05:00",
    "DBClusterArn": "arn:aws:rds:us-east-1:123456789012:cluster:sample-
cluster-2019-03-16-00-01-restored",
    "DBClusterParameterGroup": "default.docdb3.6",
    "DbClusterResourceId": "cluster-X0046Q3RH4LWSYNH3NMZKXPISU",
    "MasterUsername": "master-user",
    "EngineVersion": "3.6.0",
    "BackupRetentionPeriod": 3,
    "AssociatedRoles": [],
    "Status": "creating",
    "VpcSecurityGroups": [
      {
        "Status": "active",
        "VpcSecurityGroupId": "sg-77186e0d"
      }
    ]
  }
}

```

For more information, see [Stopping and Starting an Amazon DocumentDB Cluster](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [StartDbCluster](#) in *AWS CLI Command Reference*.

stop-db-cluster

The following code example shows how to use stop-db-cluster.

AWS CLI

To stop a running Amazon DocumentDB cluster

The following stop-db-cluster example stops the specified Amazon DocumentDB cluster.

```
aws docdb stop-db-cluster \
```

```
--db-cluster-identifier sample-cluster
```

Output:

```
{
  "DBCluster": {
    "ClusterCreateTime": "2019-03-19T18:45:01.857Z",
    "HostedZoneId": "ZNKXH85TT8WVW",
    "Engine": "docdb",
    "DBClusterMembers": [],
    "MultiAZ": false,
    "AvailabilityZones": [
      "us-east-1a",
      "us-east-1c",
      "us-east-1f"
    ],
    "StorageEncrypted": false,
    "ReaderEndpoint": "sample-cluster-2019-03-16-00-01-restored.cluster-ro-
corcjorzrlsfc.us-east-1.docdb.amazonaws.com",
    "Endpoint": "sample-cluster-2019-03-16-00-01-restored.cluster-
corcjorzrlsfc.us-east-1.docdb.amazonaws.com",
    "Port": 27017,
    "PreferredBackupWindow": "00:00-00:30",
    "DBSubnetGroup": "default",
    "DBClusterIdentifier": "sample-cluster-2019-03-16-00-01-restored",
    "PreferredMaintenanceWindow": "sat:04:30-sat:05:00",
    "DBClusterArn": "arn:aws:rds:us-east-1:123456789012:cluster:sample-
cluster-2019-03-16-00-01-restored",
    "DBClusterParameterGroup": "default.docdb3.6",
    "DbClusterResourceId": "cluster-X0046Q3RH4LWSYNH3NMZKXPISU",
    "MasterUsername": "master-user",
    "EngineVersion": "3.6.0",
    "BackupRetentionPeriod": 3,
    "AssociatedRoles": [],
    "Status": "creating",
    "VpcSecurityGroups": [
      {
        "Status": "active",
        "VpcSecurityGroupId": "sg-77186e0d"
      }
    ]
  }
}
```


For more information, see [Stopping and Starting an Amazon DocumentDB Cluster](#) in the *Amazon DocumentDB Developer Guide*.

- For API details, see [StopDbCluster](#) in *AWS CLI Command Reference*.

DynamoDB examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with DynamoDB.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

batch-get-item

The following code example shows how to use `batch-get-item`.

AWS CLI

To retrieve multiple items from a table

The following `batch-get-items` example reads multiple items from the `MusicCollection` table using a batch of three `GetItem` requests, and requests the number of read capacity units consumed by the operation. The command returns only the `AlbumTitle` attribute.

```
aws dynamodb batch-get-item \  
  --request-items file://request-items.json \  
  --return-consumed-capacity TOTAL
```

Contents of request-items.json:

```
{
  "MusicCollection": {
    "Keys": [
      {
        "Artist": {"S": "No One You Know"},
        "SongTitle": {"S": "Call Me Today"}
      },
      {
        "Artist": {"S": "Acme Band"},
        "SongTitle": {"S": "Happy Day"}
      },
      {
        "Artist": {"S": "No One You Know"},
        "SongTitle": {"S": "Scared of My Shadow"}
      }
    ],
    "ProjectionExpression": "AlbumTitle"
  }
}
```

Output:

```
{
  "Responses": {
    "MusicCollection": [
      {
        "AlbumTitle": {
          "S": "Somewhat Famous"
        }
      },
      {
        "AlbumTitle": {
          "S": "Blue Sky Blues"
        }
      },
      {
        "AlbumTitle": {
          "S": "Louder Than Ever"
        }
      }
    ]
  }
}
```

```
    },
    "UnprocessedKeys": {},
    "ConsumedCapacity": [
      {
        "TableName": "MusicCollection",
        "CapacityUnits": 1.5
      }
    ]
  }
}
```

For more information, see [Batch Operations](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [BatchGetItem](#) in *AWS CLI Command Reference*.

batch-write-item

The following code example shows how to use `batch-write-item`.

AWS CLI

To add multiple items to a table

The following `batch-write-item` example adds three new items to the `MusicCollection` table using a batch of three `PutItem` requests. It also requests information about the number of write capacity units consumed by the operation and any item collections modified by the operation.

```
aws dynamodb batch-write-item \
  --request-items file://request-items.json \
  --return-consumed-capacity INDEXES \
  --return-item-collection-metrics SIZE
```

Contents of `request-items.json`:

```
{
  "MusicCollection": [
    {
      "PutRequest": {
        "Item": {
          "Artist": {"S": "No One You Know"},
          "SongTitle": {"S": "Call Me Today"},
          "AlbumTitle": {"S": "Somewhat Famous"}
        }
      }
    }
  ]
}
```

```

    }
  },
  {
    "PutRequest": {
      "Item": {
        "Artist": {"S": "Acme Band"},
        "SongTitle": {"S": "Happy Day"},
        "AlbumTitle": {"S": "Songs About Life"}
      }
    }
  },
  {
    "PutRequest": {
      "Item": {
        "Artist": {"S": "No One You Know"},
        "SongTitle": {"S": "Scared of My Shadow"},
        "AlbumTitle": {"S": "Blue Sky Blues"}
      }
    }
  }
]
}

```

Output:

```

{
  "UnprocessedItems": {},
  "ItemCollectionMetrics": {
    "MusicCollection": [
      {
        "ItemCollectionKey": {
          "Artist": {
            "S": "No One You Know"
          }
        },
        "SizeEstimateRangeGB": [
          0.0,
          1.0
        ]
      },
      {
        "ItemCollectionKey": {

```

```

        "Artist": {
            "S": "Acme Band"
        }
    },
    "SizeEstimateRangeGB": [
        0.0,
        1.0
    ]
}
]
},
"ConsumedCapacity": [
    {
        "TableName": "MusicCollection",
        "CapacityUnits": 6.0,
        "Table": {
            "CapacityUnits": 3.0
        },
        "LocalSecondaryIndexes": {
            "AlbumTitleIndex": {
                "CapacityUnits": 3.0
            }
        }
    }
]
}

```

For more information, see [Batch Operations](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [BatchWriteItem](#) in *AWS CLI Command Reference*.

create-backup

The following code example shows how to use create-backup.

AWS CLI

To create a backup for an existing DynamoDB table

The following create-backup example creates a backup of the MusicCollection table.

```

aws dynamodb create-backup \
    --table-name MusicCollection \

```

```
--backup-name MusicCollectionBackup
```

Output:

```
{
  "BackupDetails": {
    "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection/
backup/01576616366715-b4e58d3a",
    "BackupName": "MusicCollectionBackup",
    "BackupSizeBytes": 0,
    "BackupStatus": "CREATING",
    "BackupType": "USER",
    "BackupCreationDateTime": 1576616366.715
  }
}
```

For more information, see [On-Demand Backup and Restore for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [CreateBackup](#) in *AWS CLI Command Reference*.

create-global-table

The following code example shows how to use `create-global-table`.

AWS CLI

To create a global table

The following `create-global-table` example creates a global table from two identical tables in the specified, separate AWS Regions.

```
aws dynamodb create-global-table \  
  --global-table-name MusicCollection \  
  --replication-group RegionName=us-east-2 RegionName=us-east-1 \  
  --region us-east-2
```

Output:

```
{
```

```
"GlobalTableDescription": {
  "ReplicationGroup": [
    {
      "RegionName": "us-east-2"
    },
    {
      "RegionName": "us-east-1"
    }
  ],
  "GlobalTableArn": "arn:aws:dynamodb::123456789012:global-table/
MusicCollection",
  "CreationDateTime": 1576625818.532,
  "GlobalTableStatus": "CREATING",
  "GlobalTableName": "MusicCollection"
}
}
```

For more information, see [DynamoDB Global Tables](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [CreateGlobalTable](#) in *AWS CLI Command Reference*.

create-table

The following code example shows how to use `create-table`.

AWS CLI

Example 1: To create a table with tags

The following `create-table` example uses the specified attributes and key schema to create a table named `MusicCollection`. This table uses provisioned throughput and is encrypted at rest using the default AWS owned CMK. The command also applies a tag to the table, with a key of `Owner` and a value of `blueTeam`.

```
aws dynamodb create-table \
  --table-name MusicCollection \
  --attribute-definitions AttributeName=Artist,AttributeType=S
AttributeName=SongTitle,AttributeType=S \
  --key-schema AttributeName=Artist,KeyType=HASH
AttributeName=SongTitle,KeyType=RANGE \
  --provisioned-throughput ReadCapacityUnits=5,WriteCapacityUnits=5 \
  --tags Key=Owner,Value=blueTeam
```

Output:

```
{
  "TableDescription": {
    "AttributeDefinitions": [
      {
        "AttributeName": "Artist",
        "AttributeType": "S"
      },
      {
        "AttributeName": "SongTitle",
        "AttributeType": "S"
      }
    ],
    "ProvisionedThroughput": {
      "NumberOfDecreasesToday": 0,
      "WriteCapacityUnits": 5,
      "ReadCapacityUnits": 5
    },
    "TableSizeBytes": 0,
    "TableName": "MusicCollection",
    "TableStatus": "CREATING",
    "KeySchema": [
      {
        "KeyType": "HASH",
        "AttributeName": "Artist"
      },
      {
        "KeyType": "RANGE",
        "AttributeName": "SongTitle"
      }
    ],
    "ItemCount": 0,
    "CreationDateTime": "2020-05-26T16:04:41.627000-07:00",
    "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection",
    "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
  }
}
```

For more information, see [Basic Operations for Tables](#) in the *Amazon DynamoDB Developer Guide*.

Example 2: To create a table in On-Demand Mode

The following example creates a table called `MusicCollection` using on-demand mode, rather than provisioned throughput mode. This is useful for tables with unpredictable workloads.

```
aws dynamodb create-table \  
  --table-name MusicCollection \  
  --attribute-definitions AttributeName=Artist,AttributeType=S \  
  AttributeName=SongTitle,AttributeType=S \  
  --key-schema AttributeName=Artist,KeyType=HASH \  
  AttributeName=SongTitle,KeyType=RANGE \  
  --billing-mode PAY_PER_REQUEST
```

Output:

```
{  
  "TableDescription": {  
    "AttributeDefinitions": [  
      {  
        "AttributeName": "Artist",  
        "AttributeType": "S"  
      },  
      {  
        "AttributeName": "SongTitle",  
        "AttributeType": "S"  
      }  
    ],  
    "TableName": "MusicCollection",  
    "KeySchema": [  
      {  
        "AttributeName": "Artist",  
        "KeyType": "HASH"  
      },  
      {  
        "AttributeName": "SongTitle",  
        "KeyType": "RANGE"  
      }  
    ],  
    "TableStatus": "CREATING",  
    "CreationDateTime": "2020-05-27T11:44:10.807000-07:00",  
    "ProvisionedThroughput": {  
      "NumberOfDecreasesToday": 0,  
      "ReadCapacityUnits": 0,  
      "WriteCapacityUnits": 0  
    }  
  }  
}
```

```

    },
    "TableSizeBytes": 0,
    "ItemCount": 0,
    "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection",
    "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "BillingModeSummary": {
      "BillingMode": "PAY_PER_REQUEST"
    }
  }
}

```

For more information, see [Basic Operations for Tables](#) in the *Amazon DynamoDB Developer Guide*.

Example 3: To create a table and encrypt it with a Customer Managed CMK

The following example creates a table named `MusicCollection` and encrypts it using a customer managed CMK.

```

aws dynamodb create-table \
  --table-name MusicCollection \
  --attribute-definitions AttributeName=Artist,AttributeType=S \
  AttributeName=SongTitle,AttributeType=S \
  --key-schema AttributeName=Artist,KeyType=HASH \
  AttributeName=SongTitle,KeyType=RANGE \
  --provisioned-throughput ReadCapacityUnits=5,WriteCapacityUnits=5 \
  --sse-specification Enabled=true,SSEType=KMS,KMSMasterKeyId=abcd1234-abcd-1234-
a123-ab1234a1b234

```

Output:

```

{
  "TableDescription": {
    "AttributeDefinitions": [
      {
        "AttributeName": "Artist",
        "AttributeType": "S"
      },
      {
        "AttributeName": "SongTitle",
        "AttributeType": "S"
      }
    ],

```

```

    "TableName": "MusicCollection",
    "KeySchema": [
      {
        "AttributeName": "Artist",
        "KeyType": "HASH"
      },
      {
        "AttributeName": "SongTitle",
        "KeyType": "RANGE"
      }
    ],
    "TableStatus": "CREATING",
    "CreationDateTime": "2020-05-27T11:12:16.431000-07:00",
    "ProvisionedThroughput": {
      "NumberOfDecreasesToday": 0,
      "ReadCapacityUnits": 5,
      "WriteCapacityUnits": 5
    },
    "TableSizeBytes": 0,
    "ItemCount": 0,
    "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection",
    "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "SSEDescription": {
      "Status": "ENABLED",
      "SSEType": "KMS",
      "KMSMasterKeyArn": "arn:aws:kms:us-west-2:123456789012:key/abcd1234-
abcd-1234-a123-ab1234a1b234"
    }
  }
}

```

For more information, see [Basic Operations for Tables](#) in the *Amazon DynamoDB Developer Guide*.

Example 4: To create a table with a Local Secondary Index

The following example uses the specified attributes and key schema to create a table named `MusicCollection` with a Local Secondary Index named `AlbumTitleIndex`.

```

aws dynamodb create-table \
  --table-name MusicCollection \
  --attribute-definitions AttributeName=Artist,AttributeType=S
  AttributeName=SongTitle,AttributeType=S AttributeName=AlbumTitle,AttributeType=S \

```

```

--key-schema AttributeName=Artist,KeyType=HASH
AttributeName=SongTitle,KeyType=RANGE \
--provisioned-throughput ReadCapacityUnits=10,WriteCapacityUnits=5 \
--local-secondary-indexes \
  "[
    {
      \"IndexName\": \"AlbumTitleIndex\",
      \"KeySchema\": [
        {\"AttributeName\": \"Artist\", \"KeyType\": \"HASH\"},
        {\"AttributeName\": \"AlbumTitle\", \"KeyType\": \"RANGE\"}
      ],
      \"Projection\": {
        \"ProjectionType\": \"INCLUDE\",
        \"NonKeyAttributes\": [\"Genre\", \"Year\"]
      }
    }
  ]"

```

Output:

```

{
  "TableDescription": {
    "AttributeDefinitions": [
      {
        "AttributeName": "AlbumTitle",
        "AttributeType": "S"
      },
      {
        "AttributeName": "Artist",
        "AttributeType": "S"
      },
      {
        "AttributeName": "SongTitle",
        "AttributeType": "S"
      }
    ],
    "TableName": "MusicCollection",
    "KeySchema": [
      {
        "AttributeName": "Artist",
        "KeyType": "HASH"
      },
      {

```

```
        "AttributeName": "SongTitle",
        "KeyType": "RANGE"
    }
],
"TableStatus": "CREATING",
"CreationDateTime": "2020-05-26T15:59:49.473000-07:00",
"ProvisionedThroughput": {
    "NumberOfDecreasesToday": 0,
    "ReadCapacityUnits": 10,
    "WriteCapacityUnits": 5
},
"TableSizeBytes": 0,
"ItemCount": 0,
"TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection",
"TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
"LocalSecondaryIndexes": [
    {
        "IndexName": "AlbumTitleIndex",
        "KeySchema": [
            {
                "AttributeName": "Artist",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "AlbumTitle",
                "KeyType": "RANGE"
            }
        ],
        "Projection": {
            "ProjectionType": "INCLUDE",
            "NonKeyAttributes": [
                "Genre",
                "Year"
            ]
        },
        "IndexSizeBytes": 0,
        "ItemCount": 0,
        "IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/index/AlbumTitleIndex"
    }
]
}
}
```

For more information, see [Basic Operations for Tables](#) in the *Amazon DynamoDB Developer Guide*.

Example 5: To create a table with a Global Secondary Index

The following example creates a table named `GameScores` with a Global Secondary Index called `GameTitleIndex`. The base table has a partition key of `UserId` and a sort key of `GameTitle`, allowing you to find an individual user's best score for a specific game efficiently, whereas the GSI has a partition key of `GameTitle` and a sort key of `TopScore`, allowing you to quickly find the overall highest score for a particular game.

```
aws dynamodb create-table \
  --table-name GameScores \
  --attribute-definitions AttributeName=UserId,AttributeType=S
  AttributeName=GameTitle,AttributeType=S AttributeName=TopScore,AttributeType=N \
  --key-schema AttributeName=UserId,KeyType=HASH \
    AttributeName=GameTitle,KeyType=RANGE \
  --provisioned-throughput ReadCapacityUnits=10,WriteCapacityUnits=5 \
  --global-secondary-indexes \
    "[
      {
        \"IndexName\": \"GameTitleIndex\",
        \"KeySchema\": [
          {\"AttributeName\": \"GameTitle\", \"KeyType\": \"HASH\"},
          {\"AttributeName\": \"TopScore\", \"KeyType\": \"RANGE\"}
        ],
        \"Projection\": {
          \"ProjectionType\": \"INCLUDE\",
          \"NonKeyAttributes\": [\"UserId\"]
        },
        \"ProvisionedThroughput\": {
          \"ReadCapacityUnits\": 10,
          \"WriteCapacityUnits\": 5
        }
      }
    ]"
```

Output:

```
{
  "TableDescription": {
    "AttributeDefinitions": [
```

```
    {
      "AttributeName": "GameTitle",
      "AttributeType": "S"
    },
    {
      "AttributeName": "TopScore",
      "AttributeType": "N"
    },
    {
      "AttributeName": "UserId",
      "AttributeType": "S"
    }
  ],
  "TableName": "GameScores",
  "KeySchema": [
    {
      "AttributeName": "UserId",
      "KeyType": "HASH"
    },
    {
      "AttributeName": "GameTitle",
      "KeyType": "RANGE"
    }
  ],
  "TableStatus": "CREATING",
  "CreationDateTime": "2020-05-26T17:28:15.602000-07:00",
  "ProvisionedThroughput": {
    "NumberOfDecreasesToday": 0,
    "ReadCapacityUnits": 10,
    "WriteCapacityUnits": 5
  },
  "TableSizeBytes": 0,
  "ItemCount": 0,
  "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/GameScores",
  "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "GlobalSecondaryIndexes": [
    {
      "IndexName": "GameTitleIndex",
      "KeySchema": [
        {
          "AttributeName": "GameTitle",
          "KeyType": "HASH"
        }
      ]
    }
  ]
}
```

```

        "AttributeName": "TopScore",
        "KeyType": "RANGE"
    }
],
"Projection": {
    "ProjectionType": "INCLUDE",
    "NonKeyAttributes": [
        "UserId"
    ]
},
"IndexStatus": "CREATING",
"ProvisionedThroughput": {
    "NumberOfDecreasesToday": 0,
    "ReadCapacityUnits": 10,
    "WriteCapacityUnits": 5
},
"IndexSizeBytes": 0,
"ItemCount": 0,
"IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
GameScores/index/GameTitleIndex"
    }
]
}
}

```

For more information, see [Basic Operations for Tables](#) in the *Amazon DynamoDB Developer Guide*.

Example 6: To create a table with multiple Global Secondary Indexes at once

The following example creates a table named GameScores with two Global Secondary Indexes. The GSI schemas are passed via a file, rather than on the command line.

```

aws dynamodb create-table \
  --table-name GameScores \
  --attribute-definitions AttributeName=UserId,AttributeType=S
AttributeName=GameTitle,AttributeType=S AttributeName=TopScore,AttributeType=N
AttributeName=Date,AttributeType=S \
  --key-schema AttributeName=UserId,KeyType=HASH
AttributeName=GameTitle,KeyType=RANGE \
  --provisioned-throughput ReadCapacityUnits=10,WriteCapacityUnits=5 \
  --global-secondary-indexes file://gsi.json

```


Contents of `gsi.json`:

```
[
  {
    "IndexName": "GameTitleIndex",
    "KeySchema": [
      {
        "AttributeName": "GameTitle",
        "KeyType": "HASH"
      },
      {
        "AttributeName": "TopScore",
        "KeyType": "RANGE"
      }
    ],
    "Projection": {
      "ProjectionType": "ALL"
    },
    "ProvisionedThroughput": {
      "ReadCapacityUnits": 10,
      "WriteCapacityUnits": 5
    }
  },
  {
    "IndexName": "GameDateIndex",
    "KeySchema": [
      {
        "AttributeName": "GameTitle",
        "KeyType": "HASH"
      },
      {
        "AttributeName": "Date",
        "KeyType": "RANGE"
      }
    ],
    "Projection": {
      "ProjectionType": "ALL"
    },
    "ProvisionedThroughput": {
      "ReadCapacityUnits": 5,
      "WriteCapacityUnits": 5
    }
  }
]
```

```
]
```

Output:

```
{
  "TableDescription": {
    "AttributeDefinitions": [
      {
        "AttributeName": "Date",
        "AttributeType": "S"
      },
      {
        "AttributeName": "GameTitle",
        "AttributeType": "S"
      },
      {
        "AttributeName": "TopScore",
        "AttributeType": "N"
      },
      {
        "AttributeName": "UserId",
        "AttributeType": "S"
      }
    ],
    "TableName": "GameScores",
    "KeySchema": [
      {
        "AttributeName": "UserId",
        "KeyType": "HASH"
      },
      {
        "AttributeName": "GameTitle",
        "KeyType": "RANGE"
      }
    ],
    "TableStatus": "CREATING",
    "CreationDateTime": "2020-08-04T16:40:55.524000-07:00",
    "ProvisionedThroughput": {
      "NumberOfDecreasesToday": 0,
      "ReadCapacityUnits": 10,
      "WriteCapacityUnits": 5
    },
    "TableSizeBytes": 0,
  }
}
```

```
"ItemCount": 0,
"TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/GameScores",
"TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
"GlobalSecondaryIndexes": [
  {
    "IndexName": "GameTitleIndex",
    "KeySchema": [
      {
        "AttributeName": "GameTitle",
        "KeyType": "HASH"
      },
      {
        "AttributeName": "TopScore",
        "KeyType": "RANGE"
      }
    ],
    "Projection": {
      "ProjectionType": "ALL"
    },
    "IndexStatus": "CREATING",
    "ProvisionedThroughput": {
      "NumberOfDecreasesToday": 0,
      "ReadCapacityUnits": 10,
      "WriteCapacityUnits": 5
    },
    "IndexSizeBytes": 0,
    "ItemCount": 0,
    "IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
GameScores/index/GameTitleIndex"
  },
  {
    "IndexName": "GameDateIndex",
    "KeySchema": [
      {
        "AttributeName": "GameTitle",
        "KeyType": "HASH"
      },
      {
        "AttributeName": "Date",
        "KeyType": "RANGE"
      }
    ],
    "Projection": {
      "ProjectionType": "ALL"
    }
  }
]
```

```

    },
    "IndexStatus": "CREATING",
    "ProvisionedThroughput": {
      "NumberOfDecreasesToday": 0,
      "ReadCapacityUnits": 5,
      "WriteCapacityUnits": 5
    },
    "IndexSizeBytes": 0,
    "ItemCount": 0,
    "IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
GameScores/index/GameDateIndex"
  }
]
}
}

```

For more information, see [Basic Operations for Tables](#) in the *Amazon DynamoDB Developer Guide*.

Example 7: To create a table with Streams enabled

The following example creates a table called GameScores with DynamoDB Streams enabled. Both new and old images of each item will be written to the stream.

```

aws dynamodb create-table \
  --table-name GameScores \
  --attribute-definitions AttributeName=UserId,AttributeType=S
AttributeName=GameTitle,AttributeType=S \
  --key-schema AttributeName=UserId,KeyType=HASH
AttributeName=GameTitle,KeyType=RANGE \
  --provisioned-throughput ReadCapacityUnits=10,WriteCapacityUnits=5 \
  --stream-specification StreamEnabled=TRUE,StreamViewType=NEW_AND_OLD_IMAGES

```

Output:

```

{
  "TableDescription": {
    "AttributeDefinitions": [
      {
        "AttributeName": "GameTitle",
        "AttributeType": "S"
      },
      {

```

```

        "AttributeName": "UserId",
        "AttributeType": "S"
    }
],
"TableName": "GameScores",
"KeySchema": [
    {
        "AttributeName": "UserId",
        "KeyType": "HASH"
    },
    {
        "AttributeName": "GameTitle",
        "KeyType": "RANGE"
    }
],
"TableStatus": "CREATING",
"CreationDateTime": "2020-05-27T10:49:34.056000-07:00",
"ProvisionedThroughput": {
    "NumberOfDecreasesToday": 0,
    "ReadCapacityUnits": 10,
    "WriteCapacityUnits": 5
},
"TableSizeBytes": 0,
"ItemCount": 0,
"TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/GameScores",
"TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
"StreamSpecification": {
    "StreamEnabled": true,
    "StreamViewType": "NEW_AND_OLD_IMAGES"
},
"LatestStreamLabel": "2020-05-27T17:49:34.056",
"LatestStreamArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
GameScores/stream/2020-05-27T17:49:34.056"
}
}

```

For more information, see [Basic Operations for Tables](#) in the *Amazon DynamoDB Developer Guide*.

Example 8: To create a table with Keys-Only Stream enabled

The following example creates a table called `GameScores` with DynamoDB Streams enabled. Only the key attributes of modified items are written to the stream.

```
aws dynamodb create-table \  
  --table-name GameScores \  
  --attribute-definitions AttributeName=UserId,AttributeType=S  
  AttributeName=GameTitle,AttributeType=S \  
  --key-schema AttributeName=UserId,KeyType=HASH  
  AttributeName=GameTitle,KeyType=RANGE \  
  --provisioned-throughput ReadCapacityUnits=10,WriteCapacityUnits=5 \  
  --stream-specification StreamEnabled=TRUE,StreamViewType=KEYS_ONLY
```

Output:

```
{  
  "TableDescription": {  
    "AttributeDefinitions": [  
      {  
        "AttributeName": "GameTitle",  
        "AttributeType": "S"  
      },  
      {  
        "AttributeName": "UserId",  
        "AttributeType": "S"  
      }  
    ],  
    "TableName": "GameScores",  
    "KeySchema": [  
      {  
        "AttributeName": "UserId",  
        "KeyType": "HASH"  
      },  
      {  
        "AttributeName": "GameTitle",  
        "KeyType": "RANGE"  
      }  
    ],  
    "TableStatus": "CREATING",  
    "CreationDateTime": "2023-05-25T18:45:34.140000+00:00",  
    "ProvisionedThroughput": {  
      "NumberOfDecreasesToday": 0,  
      "ReadCapacityUnits": 10,  
      "WriteCapacityUnits": 5  
    },  
    "TableSizeBytes": 0,  
    "ItemCount": 0,  
  }  
}
```

```

    "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/GameScores",
    "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "StreamSpecification": {
      "StreamEnabled": true,
      "StreamViewType": "KEYS_ONLY"
    },
    "LatestStreamLabel": "2023-05-25T18:45:34.140",
    "LatestStreamArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
GameScores/stream/2023-05-25T18:45:34.140",
    "DeletionProtectionEnabled": false
  }
}

```

For more information, see [Change data capture for DynamoDB Streams](#) in the *Amazon DynamoDB Developer Guide*.

Example 9: To create a table with the Standard Infrequent Access class

The following example creates a table called GameScores and assigns the Standard-Infrequent Access (DynamoDB Standard-IA) table class. This table class is optimized for storage being the dominant cost.

```

aws dynamodb create-table \
  --table-name GameScores \
  --attribute-definitions AttributeName=UserId,AttributeType=S
AttributeName=GameTitle,AttributeType=S \
  --key-schema AttributeName=UserId,KeyType=HASH
AttributeName=GameTitle,KeyType=RANGE \
  --provisioned-throughput ReadCapacityUnits=10,WriteCapacityUnits=5 \
  --table-class STANDARD_INFREQUENT_ACCESS

```

Output:

```

{
  "TableDescription": {
    "AttributeDefinitions": [
      {
        "AttributeName": "GameTitle",
        "AttributeType": "S"
      },
      {
        "AttributeName": "UserId",
        "AttributeType": "S"
      }
    ]
  }
}

```

```

    }
  ],
  "TableName": "GameScores",
  "KeySchema": [
    {
      "AttributeName": "UserId",
      "KeyType": "HASH"
    },
    {
      "AttributeName": "GameTitle",
      "KeyType": "RANGE"
    }
  ],
  "TableStatus": "CREATING",
  "CreationDateTime": "2023-05-25T18:33:07.581000+00:00",
  "ProvisionedThroughput": {
    "NumberOfDecreasesToday": 0,
    "ReadCapacityUnits": 10,
    "WriteCapacityUnits": 5
  },
  "TableSizeBytes": 0,
  "ItemCount": 0,
  "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/GameScores",
  "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "TableClassSummary": {
    "TableClass": "STANDARD_INFREQUENT_ACCESS"
  },
  "DeletionProtectionEnabled": false
}
}

```

For more information, see [Table classes](#) in the *Amazon DynamoDB Developer Guide*.

Example 10: To Create a table with Delete Protection enabled

The following example creates a table called GameScores and enables deletion protection.

```

aws dynamodb create-table \
  --table-name GameScores \
  --attribute-definitions AttributeName=UserId,AttributeType=S \
  AttributeName=GameTitle,AttributeType=S \
  --key-schema AttributeName=UserId,KeyType=HASH \
  AttributeName=GameTitle,KeyType=RANGE \
  --provisioned-throughput ReadCapacityUnits=10,WriteCapacityUnits=5 \

```



```
--deletion-protection-enabled
```

Output:

```
{
  "TableDescription": {
    "AttributeDefinitions": [
      {
        "AttributeName": "GameTitle",
        "AttributeType": "S"
      },
      {
        "AttributeName": "UserId",
        "AttributeType": "S"
      }
    ],
    "TableName": "GameScores",
    "KeySchema": [
      {
        "AttributeName": "UserId",
        "KeyType": "HASH"
      },
      {
        "AttributeName": "GameTitle",
        "KeyType": "RANGE"
      }
    ],
    "TableStatus": "CREATING",
    "CreationDateTime": "2023-05-25T23:02:17.093000+00:00",
    "ProvisionedThroughput": {
      "NumberOfDecreasesToday": 0,
      "ReadCapacityUnits": 10,
      "WriteCapacityUnits": 5
    },
    "TableSizeBytes": 0,
    "ItemCount": 0,
    "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/GameScores",
    "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "DeletionProtectionEnabled": true
  }
}
```

For more information, see [Using deletion protection](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [CreateTable](#) in *AWS CLI Command Reference*.

delete-backup

The following code example shows how to use delete-backup.

AWS CLI

To delete an existing DynamoDB backup

The following delete-backup example deletes the specified existing backup.

```
aws dynamodb delete-backup \  
  --backup-arn arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection/  
  backup/01576616366715-b4e58d3a
```

Output:

```
{  
  "BackupDescription": {  
    "BackupDetails": {  
      "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/  
MusicCollection/backup/01576616366715-b4e58d3a",  
      "BackupName": "MusicCollectionBackup",  
      "BackupSizeBytes": 0,  
      "BackupStatus": "DELETED",  
      "BackupType": "USER",  
      "BackupCreationDateTime": 1576616366.715  
    },  
    "SourceTableDetails": {  
      "TableName": "MusicCollection",  
      "TableId": "b0c04bcc-309b-4352-b2ae-9088af169fe2",  
      "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/  
MusicCollection",  
      "TableSizeBytes": 0,  
      "KeySchema": [  
        {  
          "AttributeName": "Artist",  
          "KeyType": "HASH"  
        },  
        {  
          "AttributeName": "SongTitle",
```

```

        "KeyType": "RANGE"
      }
    ],
    "TableCreationDateTime": 1576615228.571,
    "ProvisionedThroughput": {
      "ReadCapacityUnits": 5,
      "WriteCapacityUnits": 5
    },
    "ItemCount": 0,
    "BillingMode": "PROVISIONED"
  },
  "SourceTableFeatureDetails": {}
}
}

```

For more information, see [On-Demand Backup and Restore for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DeleteBackup](#) in *AWS CLI Command Reference*.

delete-item

The following code example shows how to use `delete-item`.

AWS CLI

Example 1: To delete an item

The following `delete-item` example deletes an item from the `MusicCollection` table and requests details about the item that was deleted and the capacity used by the request.

```

aws dynamodb delete-item \
  --table-name MusicCollection \
  --key file://key.json \
  --return-values ALL_OLD \
  --return-consumed-capacity TOTAL \
  --return-item-collection-metrics SIZE

```

Contents of `key.json`:

```

{
  "Artist": {"S": "No One You Know"},

```

```
"SongTitle": {"S": "Scared of My Shadow"}
}
```

Output:

```
{
  "Attributes": {
    "AlbumTitle": {
      "S": "Blue Sky Blues"
    },
    "Artist": {
      "S": "No One You Know"
    },
    "SongTitle": {
      "S": "Scared of My Shadow"
    }
  },
  "ConsumedCapacity": {
    "TableName": "MusicCollection",
    "CapacityUnits": 2.0
  },
  "ItemCollectionMetrics": {
    "ItemCollectionKey": {
      "Artist": {
        "S": "No One You Know"
      }
    },
    "SizeEstimateRangeGB": [
      0.0,
      1.0
    ]
  }
}
```

For more information, see [Writing an Item](#) in the *Amazon DynamoDB Developer Guide*.

Example 2: To delete an item conditionally

The following example deletes an item from the ProductCatalog table only if its ProductCategory is either Sporting Goods or Gardening Supplies and its price is between 500 and 600. It returns details about the item that was deleted.

```
aws dynamodb delete-item \
```

```
--table-name ProductCatalog \  
--key '{"Id":{"N":"456"}}' \  
--condition-expression "(ProductCategory IN (:cat1, :cat2)) and (#P between :lo  
and :hi)" \  
--expression-attribute-names file://names.json \  
--expression-attribute-values file://values.json \  
--return-values ALL_OLD
```

Contents of names.json:

```
{  
  "#P": "Price"  
}
```

Contents of values.json:

```
{  
  ":cat1": {"S": "Sporting Goods"},  
  ":cat2": {"S": "Gardening Supplies"},  
  ":lo": {"N": "500"},  
  ":hi": {"N": "600"}  
}
```

Output:

```
{  
  "Attributes": {  
    "Id": {  
      "N": "456"  
    },  
    "Price": {  
      "N": "550"  
    },  
    "ProductCategory": {  
      "S": "Sporting Goods"  
    }  
  }  
}
```

For more information, see [Writing an Item](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DeleteItem](#) in *AWS CLI Command Reference*.

delete-table

The following code example shows how to use `delete-table`.

AWS CLI

To delete a table

The following `delete-table` example deletes the `MusicCollection` table.

```
aws dynamodb delete-table \  
  --table-name MusicCollection
```

Output:

```
{  
  "TableDescription": {  
    "TableStatus": "DELETING",  
    "TableSizeBytes": 0,  
    "ItemCount": 0,  
    "TableName": "MusicCollection",  
    "ProvisionedThroughput": {  
      "NumberOfDecreasesToday": 0,  
      "WriteCapacityUnits": 5,  
      "ReadCapacityUnits": 5  
    }  
  }  
}
```

For more information, see [Deleting a Table](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DeleteTable](#) in *AWS CLI Command Reference*.

describe-backup

The following code example shows how to use `describe-backup`.

AWS CLI

To get information about an existing backup of a table

The following `describe-backup` example displays information about the specified existing backup.

```
aws dynamodb describe-backup \  
  --backup-arn arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection/  
backup/01576616366715-b4e58d3a
```

Output:

```
{  
  "BackupDescription": {  
    "BackupDetails": {  
      "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/  
MusicCollection/backup/01576616366715-b4e58d3a",  
      "BackupName": "MusicCollectionBackup",  
      "BackupSizeBytes": 0,  
      "BackupStatus": "AVAILABLE",  
      "BackupType": "USER",  
      "BackupCreationDateTime": 1576616366.715  
    },  
    "SourceTableDetails": {  
      "TableName": "MusicCollection",  
      "TableId": "b0c04bcc-309b-4352-b2ae-9088af169fe2",  
      "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/  
MusicCollection",  
      "TableSizeBytes": 0,  
      "KeySchema": [  
        {  
          "AttributeName": "Artist",  
          "KeyType": "HASH"  
        },  
        {  
          "AttributeName": "SongTitle",  
          "KeyType": "RANGE"  
        }  
      ],  
      "TableCreationDateTime": 1576615228.571,  
      "ProvisionedThroughput": {  
        "ReadCapacityUnits": 5,  
        "WriteCapacityUnits": 5  
      },  
      "ItemCount": 0,  
      "BillingMode": "PROVISIONED"  
    },  
    "SourceTableFeatureDetails": {}  
  }  
}
```

```
}
```

For more information, see [On-Demand Backup and Restore for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DescribeBackup](#) in *AWS CLI Command Reference*.

describe-continuous-backups

The following code example shows how to use `describe-continuous-backups`.

AWS CLI

To get information about continuous backups for a DynamoDB table

The following `describe-continuous-backups` example displays details about the continuous backup settings for the `MusicCollection` table.

```
aws dynamodb describe-continuous-backups \  
  --table-name MusicCollection
```

Output:

```
{  
  "ContinuousBackupsDescription": {  
    "ContinuousBackupsStatus": "ENABLED",  
    "PointInTimeRecoveryDescription": {  
      "PointInTimeRecoveryStatus": "DISABLED"  
    }  
  }  
}
```

For more information, see [Point-in-Time Recovery for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DescribeContinuousBackups](#) in *AWS CLI Command Reference*.

describe-contributor-insights

The following code example shows how to use `describe-contributor-insights`.

AWS CLI

To view Contributor Insights settings for a DynamoDB table

The following `describe-contributor-insights` example displays the Contributor Insights settings for the `MusicCollection` table and the `AlbumTitle-index` global secondary index.

```
aws dynamodb describe-contributor-insights \  
  --table-name MusicCollection \  
  --index-name AlbumTitle-index
```

Output:

```
{  
  "TableName": "MusicCollection",  
  "IndexName": "AlbumTitle-index",  
  "ContributorInsightsRuleList": [  
    "DynamoDBContributorInsights-PKC-MusicCollection-1576629651520",  
    "DynamoDBContributorInsights-SKC-MusicCollection-1576629651520",  
    "DynamoDBContributorInsights-PKT-MusicCollection-1576629651520",  
    "DynamoDBContributorInsights-SKT-MusicCollection-1576629651520"  
  ],  
  "ContributorInsightsStatus": "ENABLED",  
  "LastUpdateDateTime": 1576629654.78  
}
```

For more information, see [Analyzing Data Access Using CloudWatch Contributor Insights for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DescribeContributorInsights](#) in *AWS CLI Command Reference*.

describe-endpoints

The following code example shows how to use `describe-endpoints`.

AWS CLI

To view regional endpoint information

The following `describe-endpoints` example displays details about the endpoints for the current AWS Region.

```
aws dynamodb describe-endpoints
```

Output:

```
{
  "Endpoints": [
    {
      "Address": "dynamodb.us-west-2.amazonaws.com",
      "CachePeriodInMinutes": 1440
    }
  ]
}
```

For more information, see [Amazon DynamoDB Endpoints and Quotas](#) in the *AWS General Reference*.

- For API details, see [DescribeEndpoints](#) in *AWS CLI Command Reference*.

describe-global-table-settings

The following code example shows how to use `describe-global-table-settings`.

AWS CLI**To get information about a DynamoDB global table's settings**

The following `describe-global-table-settings` example displays the settings for the `MusicCollection` global table.

```
aws dynamodb describe-global-table-settings \
  --global-table-name MusicCollection
```

Output:

```
{
  "GlobalTableName": "MusicCollection",
  "ReplicaSettings": [
    {
      "RegionName": "us-east-1",
      "ReplicaStatus": "ACTIVE",
      "ReplicaProvisionedReadCapacityUnits": 10,
      "ReplicaProvisionedReadCapacityAutoScalingSettings": {
```

```

        "AutoScalingDisabled": true
    },
    "ReplicaProvisionedWriteCapacityUnits": 5,
    "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
        "AutoScalingDisabled": true
    }
},
{
    "RegionName": "us-east-2",
    "ReplicaStatus": "ACTIVE",
    "ReplicaProvisionedReadCapacityUnits": 10,
    "ReplicaProvisionedReadCapacityAutoScalingSettings": {
        "AutoScalingDisabled": true
    },
    "ReplicaProvisionedWriteCapacityUnits": 5,
    "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
        "AutoScalingDisabled": true
    }
}
]
}

```

For more information, see [DynamoDB Global Tables](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DescribeGlobalTableSettings](#) in *AWS CLI Command Reference*.

describe-global-table

The following code example shows how to use `describe-global-table`.

AWS CLI

To display information about a DynamoDB global table

The following `describe-global-table` example displays details about the `MusicCollection` global table.

```
aws dynamodb describe-global-table \
  --global-table-name MusicCollection
```

Output:

```
{
```

```

    "GlobalTableDescription": {
      "ReplicationGroup": [
        {
          "RegionName": "us-east-2"
        },
        {
          "RegionName": "us-east-1"
        }
      ],
      "GlobalTableArn": "arn:aws:dynamodb::123456789012:global-table/
MusicCollection",
      "CreationDateTime": 1576625818.532,
      "GlobalTableStatus": "ACTIVE",
      "GlobalTableName": "MusicCollection"
    }
  }
}

```

For more information, see [DynamoDB Global Tables](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DescribeGlobalTable](#) in *AWS CLI Command Reference*.

describe-limits

The following code example shows how to use `describe-limits`.

AWS CLI

To view provisioned-capacity limits

The following `describe-limits` example displays provisioned-capacity limits for your account in the current AWS Region.

```
aws dynamodb describe-limits
```

Output:

```

{
  "AccountMaxReadCapacityUnits": 80000,
  "AccountMaxWriteCapacityUnits": 80000,
  "TableMaxReadCapacityUnits": 40000,
  "TableMaxWriteCapacityUnits": 40000
}

```

For more information, see [Limits in DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DescribeLimits](#) in *AWS CLI Command Reference*.

describe-table-replica-auto-scaling

The following code example shows how to use describe-table-replica-auto-scaling.

AWS CLI

To view auto scaling settings across replicas of a global table

The following describe-table-replica-auto-scaling example displays auto scaling settings across replicas of the MusicCollection global table.

```
aws dynamodb describe-table-replica-auto-scaling \  
  --table-name MusicCollection
```

Output:

```
{  
  "TableAutoScalingDescription": {  
    "TableName": "MusicCollection",  
    "TableStatus": "ACTIVE",  
    "Replicas": [  
      {  
        "RegionName": "us-east-1",  
        "GlobalSecondaryIndexes": [],  
        "ReplicaProvisionedReadCapacityAutoScalingSettings": {  
          "MinimumUnits": 5,  
          "MaximumUnits": 40000,  
          "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/  
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/  
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",  
          "ScalingPolicies": [  
            {  
              "PolicyName": "DynamoDBReadCapacityUtilization:table/  
MusicCollection",  
              "TargetTrackingScalingPolicyConfiguration": {  
                "TargetValue": 70.0  
              }  
            }  
          ]  
        }  
      ]  
    }  
  }
```

```

    },
    "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
      "MinimumUnits": 5,
      "MaximumUnits": 40000,
      "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",
      "ScalingPolicies": [
        {
          "PolicyName": "DynamoDBWriteCapacityUtilization:table/
MusicCollection",
          "TargetTrackingScalingPolicyConfiguration": {
            "TargetValue": 70.0
          }
        }
      ]
    },
    "ReplicaStatus": "ACTIVE"
  },
  {
    "RegionName": "us-east-2",
    "GlobalSecondaryIndexes": [],
    "ReplicaProvisionedReadCapacityAutoScalingSettings": {
      "MinimumUnits": 5,
      "MaximumUnits": 40000,
      "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",
      "ScalingPolicies": [
        {
          "PolicyName": "DynamoDBReadCapacityUtilization:table/
MusicCollection",
          "TargetTrackingScalingPolicyConfiguration": {
            "TargetValue": 70.0
          }
        }
      ]
    },
    "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
      "MinimumUnits": 5,
      "MaximumUnits": 40000,
      "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",

```

```

        "ScalingPolicies": [
            {
                "PolicyName": "DynamoDBWriteCapacityUtilization:table/
MusicCollection",
                "TargetTrackingScalingPolicyConfiguration": {
                    "TargetValue": 70.0
                }
            }
        ],
        "ReplicaStatus": "ACTIVE"
    }
]
}
}

```

For more information, see [DynamoDB Global Tables](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DescribeTableReplicaAutoScaling](#) in *AWS CLI Command Reference*.

describe-table

The following code example shows how to use `describe-table`.

AWS CLI

To describe a table

The following `describe-table` example describes the `MusicCollection` table.

```
aws dynamodb describe-table \
  --table-name MusicCollection
```

Output:

```
{
  "Table": {
    "AttributeDefinitions": [
      {
        "AttributeName": "Artist",
        "AttributeType": "S"
      },
      {

```

```

        "AttributeName": "SongTitle",
        "AttributeType": "S"
    }
],
"ProvisionedThroughput": {
    "NumberOfDecreasesToday": 0,
    "WriteCapacityUnits": 5,
    "ReadCapacityUnits": 5
},
"TableSizeBytes": 0,
"TableName": "MusicCollection",
"TableStatus": "ACTIVE",
"KeySchema": [
    {
        "KeyType": "HASH",
        "AttributeName": "Artist"
    },
    {
        "KeyType": "RANGE",
        "AttributeName": "SongTitle"
    }
],
"ItemCount": 0,
"CreationDateTime": 1421866952.062
}
}

```

For more information, see [Describing a Table](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DescribeTable](#) in *AWS CLI Command Reference*.

describe-time-to-live

The following code example shows how to use `describe-time-to-live`.

AWS CLI

To view Time to Live settings for a table

The following `describe-time-to-live` example displays Time to Live settings for the `MusicCollection` table.

```
aws dynamodb describe-time-to-live \
```



```
--table-name MusicCollection
```

Output:

```
{
  "TimeToLiveDescription": {
    "TimeToLiveStatus": "ENABLED",
    "AttributeName": "ttl"
  }
}
```

For more information, see [Time to Live](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DescribeTimeToLive](#) in *AWS CLI Command Reference*.

get-item

The following code example shows how to use `get-item`.

AWS CLI

Example 1: To read an item in a table

The following `get-item` example retrieves an item from the `MusicCollection` table. The table has a hash-and-range primary key (`Artist` and `SongTitle`), so you must specify both of these attributes. The command also requests information about the read capacity consumed by the operation.

```
aws dynamodb get-item \
  --table-name MusicCollection \
  --key file://key.json \
  --return-consumed-capacity TOTAL
```

Contents of `key.json`:

```
{
  "Artist": {"S": "Acme Band"},
  "SongTitle": {"S": "Happy Day"}
}
```

Output:

```
{
  "Item": {
    "AlbumTitle": {
      "S": "Songs About Life"
    },
    "SongTitle": {
      "S": "Happy Day"
    },
    "Artist": {
      "S": "Acme Band"
    }
  },
  "ConsumedCapacity": {
    "TableName": "MusicCollection",
    "CapacityUnits": 0.5
  }
}
```

For more information, see [Reading an Item](#) in the *Amazon DynamoDB Developer Guide*.

Example 2: To read an item using a consistent read

The following example retrieves an item from the `MusicCollection` table using strongly consistent reads.

```
aws dynamodb get-item \
  --table-name MusicCollection \
  --key file://key.json \
  --consistent-read \
  --return-consumed-capacity TOTAL
```

Contents of `key.json`:

```
{
  "Artist": {"S": "Acme Band"},
  "SongTitle": {"S": "Happy Day"}
}
```

Output:

```
{
```

```
"Item": {
  "AlbumTitle": {
    "S": "Songs About Life"
  },
  "SongTitle": {
    "S": "Happy Day"
  },
  "Artist": {
    "S": "Acme Band"
  }
},
"ConsumedCapacity": {
  "TableName": "MusicCollection",
  "CapacityUnits": 1.0
}
}
```

For more information, see [Reading an Item](#) in the *Amazon DynamoDB Developer Guide*.

Example 3: To retrieve specific attributes of an item

The following example uses a projection expression to retrieve only three attributes of the desired item.

```
aws dynamodb get-item \
  --table-name ProductCatalog \
  --key '{"Id": {"N": "102"}}' \
  --projection-expression "#T, #C, #P" \
  --expression-attribute-names file://names.json
```

Contents of `names.json`:

```
{
  "#T": "Title",
  "#C": "ProductCategory",
  "#P": "Price"
}
```

Output:

```
{
  "Item": {
```

```
    "Price": {
      "N": "20"
    },
    "Title": {
      "S": "Book 102 Title"
    },
    "ProductCategory": {
      "S": "Book"
    }
  }
}
```

For more information, see [Reading an Item](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [GetItem](#) in *AWS CLI Command Reference*.

list-backups

The following code example shows how to use `list-backups`.

AWS CLI

Example 1: To list all existing DynamoDB backups

The following `list-backups` example lists all of your existing backups.

```
aws dynamodb list-backups
```

Output:

```
{
  "BackupSummaries": [
    {
      "TableName": "MusicCollection",
      "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",
      "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/backup/01234567890123-a1bcd234",
      "BackupName": "MusicCollectionBackup1",
      "BackupCreationDateTime": "2020-02-12T14:41:51.617000-08:00",
      "BackupStatus": "AVAILABLE",
      "BackupType": "USER",
```

```

        "BackupSizeBytes": 170
    },
    {
        "TableName": "MusicCollection",
        "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",
        "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/backup/01234567890123-b2abc345",
        "BackupName": "MusicCollectionBackup2",
        "BackupCreationDateTime": "2020-06-26T11:08:35.431000-07:00",
        "BackupStatus": "AVAILABLE",
        "BackupType": "USER",
        "BackupSizeBytes": 400
    }
]
}

```

For more information, see [On-Demand Backup and Restore for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

Example 2: To list user-created backups in a specific time range

The following example lists only backups of the MusicCollection table that were created by the user (not those automatically created by DynamoDB) with a creation date between January 1, 2020 and March 1, 2020.

```

aws dynamodb list-backups \
  --table-name MusicCollection \
  --time-range-lower-bound 1577836800 \
  --time-range-upper-bound 1583020800 \
  --backup-type USER

```

Output:

```

{
  "BackupSummaries": [
    {
      "TableName": "MusicCollection",
      "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",

```

```

        "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/backup/01234567890123-a1bcd234",
        "BackupName": "MusicCollectionBackup1",
        "BackupCreationDateTime": "2020-02-12T14:41:51.617000-08:00",
        "BackupStatus": "AVAILABLE",
        "BackupType": "USER",
        "BackupSizeBytes": 170
    }
]
}

```

For more information, see [On-Demand Backup and Restore for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

Example 3: To limit page size

The following example returns a list of all existing backups, but retrieves only one item in each call, performing multiple calls if necessary to get the entire list. Limiting the page size is useful when running list commands on a large number of resources, which can result in a "timed out" error when using the default page size of 1000.

```

aws dynamodb list-backups \
  --page-size 1

```

Output:

```

{
  "BackupSummaries": [
    {
      "TableName": "MusicCollection",
      "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",
      "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/backup/01234567890123-a1bcd234",
      "BackupName": "MusicCollectionBackup1",
      "BackupCreationDateTime": "2020-02-12T14:41:51.617000-08:00",
      "BackupStatus": "AVAILABLE",
      "BackupType": "USER",
      "BackupSizeBytes": 170
    },
    {

```

```

        "TableName": "MusicCollection",
        "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",
        "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/backup/01234567890123-b2abc345",
        "BackupName": "MusicCollectionBackup2",
        "BackupCreationDateTime": "2020-06-26T11:08:35.431000-07:00",
        "BackupStatus": "AVAILABLE",
        "BackupType": "USER",
        "BackupSizeBytes": 400
    }
]
}

```

For more information, see [On-Demand Backup and Restore for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

Example 4: To limit the number of items returned

The following example limits the number of items returned to 1. The response includes a `NextToken` value with which to retrieve the next page of results.

```

aws dynamodb list-backups \
  --max-items 1

```

Output:

```

{
  "BackupSummaries": [
    {
      "TableName": "MusicCollection",
      "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",
      "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/backup/01234567890123-a1bcd234",
      "BackupName": "MusicCollectionBackup1",
      "BackupCreationDateTime": "2020-02-12T14:41:51.617000-08:00",
      "BackupStatus": "AVAILABLE",
      "BackupType": "USER",
      "BackupSizeBytes": 170
    }
  ]
}

```

```

    ],
    "NextToken":
    "abCDeFGhiJKlMnOPqrSTuvwxYZ1aBCdEFghijK7LM51n0pqRSTuv3WxY3ZabC5dEFGhI2Jk3LmnoPQ6RST9"
  }

```

For more information, see [On-Demand Backup and Restore for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

Example 5: To retrieve the next page of results

The following command uses the `NextToken` value from a previous call to the `list-backups` command to retrieve another page of results. Since the response in this case does not include a `NextToken` value, we know that we have reached the end of the results.

```

aws dynamodb list-backups \
  --starting-token
  abCDeFGhiJKlMnOPqrSTuvwxYZ1aBCdEFghijK7LM51n0pqRSTuv3WxY3ZabC5dEFGhI2Jk3LmnoPQ6RST9

```

Output

```

{
  "BackupSummaries": [
    {
      "TableName": "MusicCollection",
      "TableId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection",
      "BackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection/backup/01234567890123-b2abc345",
      "BackupName": "MusicCollectionBackup2",
      "BackupCreationDateTime": "2020-06-26T11:08:35.431000-07:00",
      "BackupStatus": "AVAILABLE",
      "BackupType": "USER",
      "BackupSizeBytes": 400
    }
  ]
}

```

For more information, see [On-Demand Backup and Restore for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [ListBackups](#) in *AWS CLI Command Reference*.

list-contributor-insights

The following code example shows how to use `list-contributor-insights`.

AWS CLI

Example 1: To view a list of Contributor Insights summaries

The following `list-contributor-insights` example displays a list of Contributor Insights summaries.

```
aws dynamodb list-contributor-insights
```

Output:

```
{
  "ContributorInsightsSummaries": [
    {
      "TableName": "MusicCollection",
      "IndexName": "AlbumTitle-index",
      "ContributorInsightsStatus": "ENABLED"
    },
    {
      "TableName": "ProductCatalog",
      "ContributorInsightsStatus": "ENABLED"
    },
    {
      "TableName": "Forum",
      "ContributorInsightsStatus": "ENABLED"
    },
    {
      "TableName": "Reply",
      "ContributorInsightsStatus": "ENABLED"
    },
    {
      "TableName": "Thread",
      "ContributorInsightsStatus": "ENABLED"
    }
  ]
}
```

For more information, see [Analyzing Data Access Using CloudWatch Contributor Insights for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

Example 2: To limit the number of items returned

The following example limits the number of items returned to 4. The response includes a `NextToken` value with which to retrieve the next page of results.

```
aws dynamodb list-contributor-insights \  
  --max-results 4
```

Output:

```
{  
  "ContributorInsightsSummaries": [  
    {  
      "TableName": "MusicCollection",  
      "IndexName": "AlbumTitle-index",  
      "ContributorInsightsStatus": "ENABLED"  
    },  
    {  
      "TableName": "ProductCatalog",  
      "ContributorInsightsStatus": "ENABLED"  
    },  
    {  
      "TableName": "Forum",  
      "ContributorInsightsStatus": "ENABLED"  
    }  
  ],  
  "NextToken":  
  "abCDeFGhiJKlMnOPqrSTuvwxYZ1aBCdEFghijK7LM51n0pqRSTuv3WxY3ZabC5dEFGhI2Jk3LmnoPQ6RST9"  
}
```

For more information, see [Analyzing Data Access Using CloudWatch Contributor Insights for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

Example 3: To retrieve the next page of results

The following command uses the `NextToken` value from a previous call to the `list-contributor-insights` command to retrieve another page of results. Since the response in this case does not include a `NextToken` value, we know that we have reached the end of the results.

```
aws dynamodb list-contributor-insights \  
  --max-results 4 \  
  --next-token "abCDeFGhiJKlMnOPqrSTuvwxYZ1aBCdEFghijK7LM51n0pqRSTuv3WxY3ZabC5dEFGhI2Jk3LmnoPQ6RST9"
```

```
--next-token
abCDeFGhiJKlMnOPqrSTuvwXYZ1aBCdEFghijK7LM51n0pqRSTuv3WxY3ZabC5dEFGhI2Jk3LmnoPQ6RST9
```

Output:

```
{
  "ContributorInsightsSummaries": [
    {
      "TableName": "Reply",
      "ContributorInsightsStatus": "ENABLED"
    },
    {
      "TableName": "Thread",
      "ContributorInsightsStatus": "ENABLED"
    }
  ]
}
```

For more information, see [Analyzing Data Access Using CloudWatch Contributor Insights for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [ListContributorInsights](#) in *AWS CLI Command Reference*.

list-global-tables

The following code example shows how to use `list-global-tables`.

AWS CLI

To list existing DynamoDB global tables

The following `list-global-tables` example lists all of your existing global tables.

```
aws dynamodb list-global-tables
```

Output:

```
{
  "GlobalTables": [
    {
      "GlobalTableName": "MusicCollection",
      "ReplicationGroup": [
```

```
{
  "RegionName": "us-east-2"
},
{
  "RegionName": "us-east-1"
}
]
}
```

For more information, see [DynamoDB Global Tables](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [ListGlobalTables](#) in *AWS CLI Command Reference*.

list-tables

The following code example shows how to use `list-tables`.

AWS CLI

Example 1: To list tables

The following `list-tables` example lists all of the tables associated with the current AWS account and Region.

```
aws dynamodb list-tables
```

Output:

```
{
  "TableNames": [
    "Forum",
    "ProductCatalog",
    "Reply",
    "Thread"
  ]
}
```

For more information, see [Listing Table Names](#) in the *Amazon DynamoDB Developer Guide*.

Example 2: To limit page size

The following example returns a list of all existing tables, but retrieves only one item in each call, performing multiple calls if necessary to get the entire list. Limiting the page size is useful when running list commands on a large number of resources, which can result in a "timed out" error when using the default page size of 1000.

```
aws dynamodb list-tables \  
  --page-size 1
```

Output:

```
{  
  "TableNames": [  
    "Forum",  
    "ProductCatalog",  
    "Reply",  
    "Thread"  
  ]  
}
```

For more information, see [Listing Table Names](#) in the *Amazon DynamoDB Developer Guide*.

Example 3: To limit the number of items returned

The following example limits the number of items returned to 2. The response includes a `NextToken` value with which to retrieve the next page of results.

```
aws dynamodb list-tables \  
  --max-items 2
```

Output:

```
{  
  "TableNames": [  
    "Forum",  
    "ProductCatalog"  
  ],  
  "NextToken":  
  "abCDeFGhiJKlmnOPqrSTuvwXYZ1aBCdEFghijK7LM51n0pqRSTuv3WxY3ZabC5dEFGhI2Jk3LmnoPQ6RST9"  
}
```

For more information, see [Listing Table Names](#) in the *Amazon DynamoDB Developer Guide*.

Example 4: To retrieve the next page of results

The following command uses the `NextToken` value from a previous call to the `list-tables` command to retrieve another page of results. Since the response in this case does not include a `NextToken` value, we know that we have reached the end of the results.

```
aws dynamodb list-tables \  
  --starting-token  
  abcDeFGhiJKlMnOPqrSTuvwXYZ1aBCdEFghijK7LM51n0pqRSTuv3WxY3ZabC5dEFGhI2Jk3LmnoPQ6RST9
```

Output:

```
{  
  "TableNames": [  
    "Reply",  
    "Thread"  
  ]  
}
```

For more information, see [Listing Table Names](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [ListTables](#) in *AWS CLI Command Reference*.

list-tags-of-resource

The following code example shows how to use `list-tags-of-resource`.

AWS CLI

Example 1: To list tags of a DynamoDB resource

The following `list-tags-of-resource` example displays tags for the `MusicCollection` table.

```
aws dynamodb list-tags-of-resource \  
  --resource-arn arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection
```

Output:

```
{  
  "Tags": [  
    {  
      "Key": "Name",  
      "Value": "MusicCollection"  
    }  
  ]  
}
```

```
{
  {
    "Key": "Owner",
    "Value": "blueTeam"
  },
  {
    "Key": "Environment",
    "Value": "Production"
  }
]
```

For more information, see [Tagging for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

Example 2: To limit the number of tags returned

The following example limits the number of tags returned to 1. The response includes a `NextToken` value with which to retrieve the next page of results.

```
aws dynamodb list-tags-of-resource \
  --resource-arn arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection \
  --max-items 1
```

Output:

```
{
  "Tags": [
    {
      "Key": "Owner",
      "Value": "blueTeam"
    }
  ],
  "NextToken":
  "abCDeFGhiJKlMnOPqrSTuvwxYZ1aBCdEFghijK7LM51nOpqRSTuv3WxY3ZabC5dEFGhI2Jk3LmnoPQ6RST9"
}
```

For more information, see [Tagging for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

Example 3: To retrieve the next page of results

The following command uses the `NextToken` value from a previous call to the `list-tags-of-resource` command to retrieve another page of results. Since the response in this case does not include a `NextToken` value, we know that we have reached the end of the results.

```
aws dynamodb list-tags-of-resource \  
  --resource-arn arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection \  
  --starting-token  
abCDeFGhiJKlmnOPqrSTuvwXYZ1aBCdEFghijK7LM51n0pqRSTuv3WxY3ZabC5dEFGhI2Jk3LmnoPQ6RST9
```

Output:

```
{  
  "Tags": [  
    {  
      "Key": "Environment",  
      "Value": "Production"  
    }  
  ]  
}
```

For more information, see [Tagging for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [ListTagsOfResource](#) in *AWS CLI Command Reference*.

put-item

The following code example shows how to use `put-item`.

AWS CLI

Example 1: To add an item to a table

The following `put-item` example adds a new item to the *MusicCollection* table.

```
aws dynamodb put-item \  
  --table-name MusicCollection \  
  --item file://item.json \  
  --return-consumed-capacity TOTAL \  
  --return-item-collection-metrics SIZE
```

Contents of `item.json`:

```
{  
  "Artist": {"S": "No One You Know"},  
  "SongTitle": {"S": "Call Me Today"},
```



```
"AlbumTitle": {"S": "Greatest Hits"}
}
```

Output:

```
{
  "ConsumedCapacity": {
    "TableName": "MusicCollection",
    "CapacityUnits": 1.0
  },
  "ItemCollectionMetrics": {
    "ItemCollectionKey": {
      "Artist": {
        "S": "No One You Know"
      }
    },
    "SizeEstimateRangeGB": [
      0.0,
      1.0
    ]
  }
}
```

For more information, see [Writing an Item](#) in the *Amazon DynamoDB Developer Guide*.

Example 2: To conditionally overwrite an item in a table

The following `put-item` example overwrites an existing item in the `MusicCollection` table only if that existing item has an `AlbumTitle` attribute with a value of `Greatest Hits`. The command returns the previous value of the item.

```
aws dynamodb put-item \
  --table-name MusicCollection \
  --item file://item.json \
  --condition-expression "#A = :A" \
  --expression-attribute-names file://names.json \
  --expression-attribute-values file://values.json \
  --return-values ALL_OLD
```

Contents of `item.json`:

```
{
```

```
"Artist": {"S": "No One You Know"},
"SongTitle": {"S": "Call Me Today"},
"AlbumTitle": {"S": "Somewhat Famous"}
}
```

Contents of `names.json`:

```
{
  "#A": "AlbumTitle"
}
```

Contents of `values.json`:

```
{
  ":A": {"S": "Greatest Hits"}
}
```

Output:

```
{
  "Attributes": {
    "AlbumTitle": {
      "S": "Greatest Hits"
    },
    "Artist": {
      "S": "No One You Know"
    },
    "SongTitle": {
      "S": "Call Me Today"
    }
  }
}
```

If the key already exists, you should see the following output:

```
A client error (ConditionalCheckFailedException) occurred when calling the PutItem
operation: The conditional request failed.
```

For more information, see [Writing an Item](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [PutItem](#) in *AWS CLI Command Reference*.

query

The following code example shows how to use query.

AWS CLI

Example 1: To query a table

The following query example queries items in the MusicCollection table. The table has a hash-and-range primary key (Artist and SongTitle), but this query only specifies the hash key value. It returns song titles by the artist named "No One You Know".

```
aws dynamodb query \  
  --table-name MusicCollection \  
  --projection-expression "SongTitle" \  
  --key-condition-expression "Artist = :v1" \  
  --expression-attribute-values file://expression-attributes.json \  
  --return-consumed-capacity TOTAL
```

Contents of expression-attributes.json:

```
{  
  ":v1": {"S": "No One You Know"}  
}
```

Output:

```
{  
  "Items": [  
    {  
      "SongTitle": {  
        "S": "Call Me Today"  
      },  
      "SongTitle": {  
        "S": "Scared of My Shadow"  
      }  
    }  
  ],  
  "Count": 2,  
  "ScannedCount": 2,  
  "ConsumedCapacity": {  
    "TableName": "MusicCollection",
```

```
    "CapacityUnits": 0.5
  }
}
```

For more information, see [Working with Queries in DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

Example 2: To query a table using strongly consistent reads and traverse the index in descending order

The following example performs the same query as the first example, but returns results in reverse order and uses strongly consistent reads.

```
aws dynamodb query \
  --table-name MusicCollection \
  --projection-expression "SongTitle" \
  --key-condition-expression "Artist = :v1" \
  --expression-attribute-values file://expression-attributes.json \
  --consistent-read \
  --no-scan-index-forward \
  --return-consumed-capacity TOTAL
```

Contents of `expression-attributes.json`:

```
{
  ":v1": {"S": "No One You Know"}
}
```

Output:

```
{
  "Items": [
    {
      "SongTitle": {
        "S": "Scared of My Shadow"
      }
    },
    {
      "SongTitle": {
        "S": "Call Me Today"
      }
    }
  ]
}
```

```

    ],
    "Count": 2,
    "ScannedCount": 2,
    "ConsumedCapacity": {
      "TableName": "MusicCollection",
      "CapacityUnits": 1.0
    }
  }
}

```

For more information, see [Working with Queries in DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

Example 3: To filter out specific results

The following example queries the MusicCollection but excludes results with specific values in the AlbumTitle attribute. Note that this does not affect the ScannedCount or ConsumedCapacity, because the filter is applied after the items have been read.

```

aws dynamodb query \
  --table-name MusicCollection \
  --key-condition-expression "#n1 = :v1" \
  --filter-expression "NOT (#n2 IN (:v2, :v3))" \
  --expression-attribute-names file://names.json \
  --expression-attribute-values file://values.json \
  --return-consumed-capacity TOTAL

```

Contents of values.json:

```

{
  ":v1": {"S": "No One You Know"},
  ":v2": {"S": "Blue Sky Blues"},
  ":v3": {"S": "Greatest Hits"}
}

```

Contents of names.json:

```

{
  "#n1": "Artist",
  "#n2": "AlbumTitle"
}

```

Output:

```
{
  "Items": [
    {
      "AlbumTitle": {
        "S": "Somewhat Famous"
      },
      "Artist": {
        "S": "No One You Know"
      },
      "SongTitle": {
        "S": "Call Me Today"
      }
    }
  ],
  "Count": 1,
  "ScannedCount": 2,
  "ConsumedCapacity": {
    "TableName": "MusicCollection",
    "CapacityUnits": 0.5
  }
}
```

For more information, see [Working with Queries in DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

Example 4: To retrieve only an item count

The following example retrieves a count of items matching the query, but does not retrieve any of the items themselves.

```
aws dynamodb query \
  --table-name MusicCollection \
  --select COUNT \
  --key-condition-expression "Artist = :v1" \
  --expression-attribute-values file://expression-attributes.json
```

Contents of `expression-attributes.json`:

```
{
  ":v1": {"S": "No One You Know"}
}
```

Output:

```
{
  "Count": 2,
  "ScannedCount": 2,
  "ConsumedCapacity": null
}
```

For more information, see [Working with Queries in DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

Example 5: To query an index

The following example queries the local secondary index AlbumTitleIndex. The query returns all attributes from the base table that have been projected into the local secondary index. Note that when querying a local secondary index or global secondary index, you must also provide the name of the base table using the `table-name` parameter.

```
aws dynamodb query \
  --table-name MusicCollection \
  --index-name AlbumTitleIndex \
  --key-condition-expression "Artist = :v1" \
  --expression-attribute-values file://expression-attributes.json \
  --select ALL_PROJECTED_ATTRIBUTES \
  --return-consumed-capacity INDEXES
```

Contents of expression-attributes.json:

```
{
  ":v1": {"S": "No One You Know"}
}
```

Output:

```
{
  "Items": [
    {
      "AlbumTitle": {
        "S": "Blue Sky Blues"
      },

```

```

    "Artist": {
      "S": "No One You Know"
    },
    "SongTitle": {
      "S": "Scared of My Shadow"
    }
  },
  {
    "AlbumTitle": {
      "S": "Somewhat Famous"
    },
    "Artist": {
      "S": "No One You Know"
    },
    "SongTitle": {
      "S": "Call Me Today"
    }
  }
],
"Count": 2,
"ScannedCount": 2,
"ConsumedCapacity": {
  "TableName": "MusicCollection",
  "CapacityUnits": 0.5,
  "Table": {
    "CapacityUnits": 0.0
  },
  "LocalSecondaryIndexes": {
    "AlbumTitleIndex": {
      "CapacityUnits": 0.5
    }
  }
}
}

```

For more information, see [Working with Queries in DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [Query](#) in *AWS CLI Command Reference*.

restore-table-from-backup

The following code example shows how to use `restore-table-from-backup`.

AWS CLI

To restore a DynamoDB table from an existing backup

The following `restore-table-from-backup` example restores the specified table from an existing backup.

```
aws dynamodb restore-table-from-backup \  
  --target-table-name MusicCollection \  
  --backup-arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection/  
  backup/01576616366715-b4e58d3a
```

Output:

```
{  
  "TableDescription": {  
    "AttributeDefinitions": [  
      {  
        "AttributeName": "Artist",  
        "AttributeType": "S"  
      },  
      {  
        "AttributeName": "SongTitle",  
        "AttributeType": "S"  
      }  
    ],  
    "TableName": "MusicCollection2",  
    "KeySchema": [  
      {  
        "AttributeName": "Artist",  
        "KeyType": "HASH"  
      },  
      {  
        "AttributeName": "SongTitle",  
        "KeyType": "RANGE"  
      }  
    ],  
    "TableStatus": "CREATING",  
    "CreationDateTime": 1576618274.326,  
    "ProvisionedThroughput": {  
      "NumberOfDecreasesToday": 0,  
      "ReadCapacityUnits": 5,  
      "WriteCapacityUnits": 5  
    }  
  }  
}
```

```

    },
    "TableSizeBytes": 0,
    "ItemCount": 0,
    "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection2",
    "TableId": "114865c9-5ef3-496c-b4d1-c4cbdd2d44fb",
    "BillingModeSummary": {
      "BillingMode": "PROVISIONED"
    },
    "RestoreSummary": {
      "SourceBackupArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/backup/01576616366715-b4e58d3a",
      "SourceTableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",
      "RestoreDateTime": 1576616366.715,
      "RestoreInProgress": true
    }
  }
}
}

```

For more information, see [On-Demand Backup and Restore for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [RestoreTableFromBackup](#) in *AWS CLI Command Reference*.

restore-table-to-point-in-time

The following code example shows how to use `restore-table-to-point-in-time`.

AWS CLI

To restore a DynamoDB table to a point in time

The following `restore-table-to-point-in-time` example restores the `MusicCollection` table to the specified point in time.

```

aws dynamodb restore-table-to-point-in-time \
  --source-table-name MusicCollection \
  --target-table-name MusicCollectionRestore \
  --restore-date-time 1576622404.0

```

Output:

```
{
  "TableDescription": {
    "AttributeDefinitions": [
      {
        "AttributeName": "Artist",
        "AttributeType": "S"
      },
      {
        "AttributeName": "SongTitle",
        "AttributeType": "S"
      }
    ],
    "TableName": "MusicCollectionRestore",
    "KeySchema": [
      {
        "AttributeName": "Artist",
        "KeyType": "HASH"
      },
      {
        "AttributeName": "SongTitle",
        "KeyType": "RANGE"
      }
    ],
    "TableStatus": "CREATING",
    "CreationDateTime": 1576623311.86,
    "ProvisionedThroughput": {
      "NumberOfDecreasesToday": 0,
      "ReadCapacityUnits": 5,
      "WriteCapacityUnits": 5
    },
    "TableSizeBytes": 0,
    "ItemCount": 0,
    "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollectionRestore",
    "TableId": "befd9e0e-1843-4dc6-a147-d6d00e85cb1f",
    "BillingModeSummary": {
      "BillingMode": "PROVISIONED"
    },
    "RestoreSummary": {
      "SourceTableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection",
      "RestoreDateTime": 1576622404.0,
      "RestoreInProgress": true
    }
  }
}
```

```

    }
  }
}

```

For more information, see [Point-in-Time Recovery for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [RestoreTableToPointInTime](#) in *AWS CLI Command Reference*.

scan

The following code example shows how to use scan.

AWS CLI

To scan a table

The following scan example scans the entire MusicCollection table, and then narrows the results to songs by the artist "No One You Know". For each item, only the album title and song title are returned.

```

aws dynamodb scan \
  --table-name MusicCollection \
  --filter-expression "Artist = :a" \
  --projection-expression "#ST, #AT" \
  --expression-attribute-names file://expression-attribute-names.json \
  --expression-attribute-values file://expression-attribute-values.json

```

Contents of expression-attribute-names.json:

```

{
  "#ST": "SongTitle",
  "#AT": "AlbumTitle"
}

```

Contents of expression-attribute-values.json:

```

{
  ":a": {"S": "No One You Know"}
}

```

Output:

```
{
  "Count": 2,
  "Items": [
    {
      "SongTitle": {
        "S": "Call Me Today"
      },
      "AlbumTitle": {
        "S": "Somewhat Famous"
      }
    },
    {
      "SongTitle": {
        "S": "Scared of My Shadow"
      },
      "AlbumTitle": {
        "S": "Blue Sky Blues"
      }
    }
  ],
  "ScannedCount": 3,
  "ConsumedCapacity": null
}
```

For more information, see [Working with Scans in DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [Scan](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI**To add tags to a DynamoDB resource**

The following tag-resource example adds a tag key/value pair to the MusicCollection table.

```
aws dynamodb tag-resource \
```

```
--resource-arn arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection \  
--tags Key=Owner,Value=blueTeam
```

This command produces no output.

For more information, see [Tagging for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

transact-get-items

The following code example shows how to use `transact-get-items`.

AWS CLI

To retrieve multiple items atomically from one or more tables

The following `transact-get-items` example retrieves multiple items atomically.

```
aws dynamodb transact-get-items \  
--transact-items file://transact-items.json \  
--return-consumed-capacity TOTAL
```

Contents of `transact-items.json`:

```
[  
  {  
    "Get": {  
      "Key": {  
        "Artist": {"S": "Acme Band"},  
        "SongTitle": {"S": "Happy Day"}  
      },  
      "TableName": "MusicCollection"  
    }  
  },  
  {  
    "Get": {  
      "Key": {  
        "Artist": {"S": "No One You Know"},  
        "SongTitle": {"S": "Call Me Today"}  
      },  
      "TableName": "MusicCollection"  
    }  
  }  
]
```

```
    }  
  }  
]
```

Output:

```
{  
  "ConsumedCapacity": [  
    {  
      "TableName": "MusicCollection",  
      "CapacityUnits": 4.0,  
      "ReadCapacityUnits": 4.0  
    }  
  ],  
  "Responses": [  
    {  
      "Item": {  
        "AlbumTitle": {  
          "S": "Songs About Life"  
        },  
        "Artist": {  
          "S": "Acme Band"  
        },  
        "SongTitle": {  
          "S": "Happy Day"  
        }  
      }  
    },  
    {  
      "Item": {  
        "AlbumTitle": {  
          "S": "Somewhat Famous"  
        },  
        "Artist": {  
          "S": "No One You Know"  
        },  
        "SongTitle": {  
          "S": "Call Me Today"  
        }  
      }  
    }  
  ]  
}
```

For more information, see [Managing Complex Workflows with DynamoDB Transactions](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [TransactGetItems](#) in *AWS CLI Command Reference*.

transact-write-items

The following code example shows how to use `transact-write-items`.

AWS CLI

Example 1: To write items atomically to one or more tables

The following `transact-write-items` example updates one item and deletes another. The operation fails if either operation fails, or if either item contains a `Rating` attribute.

```
aws dynamodb transact-write-items \  
  --transact-items file://transact-items.json \  
  --return-consumed-capacity TOTAL \  
  --return-item-collection-metrics SIZE
```

Contents of the `transact-items.json` file:

```
[  
  {  
    "Update": {  
      "Key": {  
        "Artist": {"S": "Acme Band"},  
        "SongTitle": {"S": "Happy Day"}  
      },  
      "UpdateExpression": "SET AlbumTitle = :newval",  
      "ExpressionAttributeValues": {  
        ":newval": {"S": "Updated Album Title"}  
      },  
      "TableName": "MusicCollection",  
      "ConditionExpression": "attribute_not_exists(Rating)"  
    },  
    {  
      "Delete": {  
        "Key": {  
          "Artist": {"S": "No One You Know"},
```



```

        "SongTitle": {"S": "Call Me Today"}
    },
    "TableName": "MusicCollection",
    "ConditionExpression": "attribute_not_exists(Rating)"
}
]

```

Output:

```

{
  "ConsumedCapacity": [
    {
      "TableName": "MusicCollection",
      "CapacityUnits": 10.0,
      "WriteCapacityUnits": 10.0
    }
  ],
  "ItemCollectionMetrics": {
    "MusicCollection": [
      {
        "ItemCollectionKey": {
          "Artist": {
            "S": "No One You Know"
          }
        },
        "SizeEstimateRangeGB": [
          0.0,
          1.0
        ]
      },
      {
        "ItemCollectionKey": {
          "Artist": {
            "S": "Acme Band"
          }
        },
        "SizeEstimateRangeGB": [
          0.0,
          1.0
        ]
      }
    ]
  }
}

```

```
}  
}
```

For more information, see [Managing Complex Workflows with DynamoDB Transactions](#) in the *Amazon DynamoDB Developer Guide*.

Example 2: To write items atomically using a client request token

The following command uses a client request token to make the call to `transact-write-items` idempotent, meaning that multiple calls have the same effect as one single call.

```
aws dynamodb transact-write-items \  
  --transact-items file://transact-items.json \  
  --client-request-token abc123
```

Contents of the `transact-items.json` file:

```
[  
  {  
    "Update": {  
      "Key": {  
        "Artist": {"S": "Acme Band"},  
        "SongTitle": {"S": "Happy Day"}  
      },  
      "UpdateExpression": "SET AlbumTitle = :newval",  
      "ExpressionAttributeValues": {  
        ":newval": {"S": "Updated Album Title"}  
      },  
      "TableName": "MusicCollection",  
      "ConditionExpression": "attribute_not_exists(Rating)"  
    },  
    {  
      "Delete": {  
        "Key": {  
          "Artist": {"S": "No One You Know"},  
          "SongTitle": {"S": "Call Me Today"}  
        },  
        "TableName": "MusicCollection",  
        "ConditionExpression": "attribute_not_exists(Rating)"  
      }  
    }  
  ]
```

```
] ]
```

This command produces no output.

For more information, see [Managing Complex Workflows with DynamoDB Transactions](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [TransactWriteItems](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag from a DynamoDB resource

The following `untag-resource` example removes the tag with the key `Owner` from the `MusicCollection` table.

```
aws dynamodb untag-resource \  
  --resource-arn arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection \  
  --tag-keys Owner
```

This command produces no output.

For more information, see [Tagging for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-continuous-backups

The following code example shows how to use `update-continuous-backups`.

AWS CLI

To update continuous backup settings for a DynamoDB table

The following `update-continuous-backups` example enables point-in-time recovery for the `MusicCollection` table.

```
aws dynamodb update-continuous-backups \  
  --table-name MusicCollection
```

```
--table-name MusicCollection \  
--point-in-time-recovery-specification PointInTimeRecoveryEnabled=true
```

Output:

```
{  
  "ContinuousBackupsDescription": {  
    "ContinuousBackupsStatus": "ENABLED",  
    "PointInTimeRecoveryDescription": {  
      "PointInTimeRecoveryStatus": "ENABLED",  
      "EarliestRestorableDateTime": 1576622404.0,  
      "LatestRestorableDateTime": 1576622404.0  
    }  
  }  
}
```

For more information, see [Point-in-Time Recovery for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [UpdateContinuousBackups](#) in *AWS CLI Command Reference*.

update-contributor-insights

The following code example shows how to use `update-contributor-insights`.

AWS CLI

To enable Contributor Insights on a table

The following `update-contributor-insights` example enables Contributor Insights on the `MusicCollection` table and the `AlbumTitle-index` global secondary index.

```
aws dynamodb update-contributor-insights \  
  --table-name MusicCollection \  
  --index-name AlbumTitle-index \  
  --contributor-insights-action ENABLE
```

Output:

```
{  
  "TableName": "MusicCollection",
```

```
"IndexName": "AlbumTitle-index",
"ContributorInsightsStatus": "ENABLING"
}
```

For more information, see [Analyzing Data Access Using CloudWatch Contributor Insights for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [UpdateContributorInsights](#) in *AWS CLI Command Reference*.

update-global-table-settings

The following code example shows how to use `update-global-table-settings`.

AWS CLI

To update provisioned write capacity settings on a DynamoDB global table

The following `update-global-table-settings` example sets the provisioned write capacity of the `MusicCollection` global table to 15.

```
aws dynamodb update-global-table-settings \
  --global-table-name MusicCollection \
  --global-table-provisioned-write-capacity-units 15
```

Output:

```
{
  "GlobalTableName": "MusicCollection",
  "ReplicaSettings": [
    {
      "RegionName": "eu-west-1",
      "ReplicaStatus": "UPDATING",
      "ReplicaProvisionedReadCapacityUnits": 10,
      "ReplicaProvisionedReadCapacityAutoScalingSettings": {
        "AutoScalingDisabled": true
      },
      "ReplicaProvisionedWriteCapacityUnits": 10,
      "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
        "AutoScalingDisabled": true
      }
    },
    {
```

```
    "RegionName": "us-east-1",
    "ReplicaStatus": "UPDATING",
    "ReplicaProvisionedReadCapacityUnits": 10,
    "ReplicaProvisionedReadCapacityAutoScalingSettings": {
      "AutoScalingDisabled": true
    },
    "ReplicaProvisionedWriteCapacityUnits": 10,
    "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
      "AutoScalingDisabled": true
    }
  },
  {
    "RegionName": "us-east-2",
    "ReplicaStatus": "UPDATING",
    "ReplicaProvisionedReadCapacityUnits": 10,
    "ReplicaProvisionedReadCapacityAutoScalingSettings": {
      "AutoScalingDisabled": true
    },
    "ReplicaProvisionedWriteCapacityUnits": 10,
    "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
      "AutoScalingDisabled": true
    }
  }
]
```

For more information, see [DynamoDB Global Tables](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [UpdateGlobalTableSettings](#) in *AWS CLI Command Reference*.

update-global-table

The following code example shows how to use `update-global-table`.

AWS CLI

To update a DynamoDB global table

The following `update-global-table` example adds a replica in the specified Region to the `MusicCollection` global table.

```
aws dynamodb update-global-table \  
  --global-table-name MusicCollection \  
  --region us-east-1
```

```
--replica-updates Create={RegionName=eu-west-1}
```

Output:

```
{
  "GlobalTableDescription": {
    "ReplicationGroup": [
      {
        "RegionName": "eu-west-1"
      },
      {
        "RegionName": "us-east-2"
      },
      {
        "RegionName": "us-east-1"
      }
    ],
    "GlobalTableArn": "arn:aws:dynamodb::123456789012:global-table/
MusicCollection",
    "CreationDateTime": 1576625818.532,
    "GlobalTableStatus": "ACTIVE",
    "GlobalTableName": "MusicCollection"
  }
}
```

For more information, see [DynamoDB Global Tables](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [UpdateGlobalTable](#) in *AWS CLI Command Reference*.

update-item

The following code example shows how to use `update-item`.

AWS CLI

Example 1: To update an item in a table

The following `update-item` example updates an item in the `MusicCollection` table. It adds a new attribute (`Year`) and modifies the `AlbumTitle` attribute. All of the attributes in the item, as they appear after the update, are returned in the response.

```
aws dynamodb update-item \
```

```
--table-name MusicCollection \  
--key file://key.json \  
--update-expression "SET #Y = :y, #AT = :t" \  
--expression-attribute-names file://expression-attribute-names.json \  
--expression-attribute-values file://expression-attribute-values.json \  
--return-values ALL_NEW \  
--return-consumed-capacity TOTAL \  
--return-item-collection-metrics SIZE
```

Contents of key.json:

```
{  
  "Artist": {"S": "Acme Band"},  
  "SongTitle": {"S": "Happy Day"}  
}
```

Contents of expression-attribute-names.json:

```
{  
  "#Y": "Year", "#AT": "AlbumTitle"  
}
```

Contents of expression-attribute-values.json:

```
{  
  ":y":{"N": "2015"},  
  ":t":{"S": "Louder Than Ever"}  
}
```

Output:

```
{  
  "Attributes": {  
    "AlbumTitle": {  
      "S": "Louder Than Ever"  
    },  
    "Awards": {  
      "N": "10"  
    },  
    "Artist": {  
      "S": "Acme Band"  
    }  
  }  
}
```



```

    },
    "Year": {
      "N": "2015"
    },
    "SongTitle": {
      "S": "Happy Day"
    }
  },
  "ConsumedCapacity": {
    "TableName": "MusicCollection",
    "CapacityUnits": 3.0
  },
  "ItemCollectionMetrics": {
    "ItemCollectionKey": {
      "Artist": {
        "S": "Acme Band"
      }
    },
    "SizeEstimateRangeGB": [
      0.0,
      1.0
    ]
  }
}

```

For more information, see [Writing an Item](#) in the *Amazon DynamoDB Developer Guide*.

Example 2: To update an item conditionally

The following example updates an item in the `MusicCollection` table, but only if the existing item does not already have a `Year` attribute.

```

aws dynamodb update-item \
  --table-name MusicCollection \
  --key file://key.json \
  --update-expression "SET #Y = :y, #AT = :t" \
  --expression-attribute-names file://expression-attribute-names.json \
  --expression-attribute-values file://expression-attribute-values.json \
  --condition-expression "attribute_not_exists(#Y)"

```

Contents of `key.json`:

```
{
```

```
"Artist": {"S": "Acme Band"},
"SongTitle": {"S": "Happy Day"}
}
```

Contents of `expression-attribute-names.json`:

```
{
  "#Y": "Year",
  "#AT": "AlbumTitle"
}
```

Contents of `expression-attribute-values.json`:

```
{
  ":y": {"N": "2015"},
  ":t": {"S": "Louder Than Ever"}
}
```

If the item already has a `Year` attribute, DynamoDB returns the following output.

```
An error occurred (ConditionalCheckFailedException) when calling the UpdateItem
operation: The conditional request failed
```

For more information, see [Writing an Item](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [UpdateItem](#) in *AWS CLI Command Reference*.

update-table-replica-auto-scaling

The following code example shows how to use `update-table-replica-auto-scaling`.

AWS CLI

To update auto scaling settings across replicas of a global table

The following `update-table-replica-auto-scaling` example updates write capacity auto scaling settings across replicas of the specified global table.

```
aws dynamodb update-table-replica-auto-scaling \
  --table-name MusicCollection \
  --provisioned-write-capacity-auto-scaling-update file://auto-scaling-policy.json
```

Contents of auto-scaling-policy.json:

```
{
  "MinimumUnits": 10,
  "MaximumUnits": 100,
  "AutoScalingDisabled": false,
  "ScalingPolicyUpdate": {
    "PolicyName": "DynamoDBWriteCapacityUtilization:table/MusicCollection",
    "TargetTrackingScalingPolicyConfiguration": {
      "TargetValue": 80
    }
  }
}
```

Output:

```
{
  "TableAutoScalingDescription": {
    "TableName": "MusicCollection",
    "TableStatus": "ACTIVE",
    "Replicas": [
      {
        "RegionName": "eu-central-1",
        "GlobalSecondaryIndexes": [],
        "ReplicaProvisionedReadCapacityAutoScalingSettings": {
          "MinimumUnits": 5,
          "MaximumUnits": 40000,
          "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",
          "ScalingPolicies": [
            {
              "PolicyName": "DynamoDBReadCapacityUtilization:table/
MusicCollection",
              "TargetTrackingScalingPolicyConfiguration": {
                "TargetValue": 70.0
              }
            }
          ]
        },
        "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
          "MinimumUnits": 10,
          "MaximumUnits": 100,
```

```

        "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",
        "ScalingPolicies": [
            {
                "PolicyName": "DynamoDBWriteCapacityUtilization:table/
MusicCollection",
                "TargetTrackingScalingPolicyConfiguration": {
                    "TargetValue": 80.0
                }
            }
        ],
        "ReplicaStatus": "ACTIVE"
    },
    {
        "RegionName": "us-east-1",
        "GlobalSecondaryIndexes": [],
        "ReplicaProvisionedReadCapacityAutoScalingSettings": {
            "MinimumUnits": 5,
            "MaximumUnits": 40000,
            "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",
            "ScalingPolicies": [
                {
                    "PolicyName": "DynamoDBReadCapacityUtilization:table/
MusicCollection",
                    "TargetTrackingScalingPolicyConfiguration": {
                        "TargetValue": 70.0
                    }
                }
            ]
        },
        "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
            "MinimumUnits": 10,
            "MaximumUnits": 100,
            "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",
            "ScalingPolicies": [
                {
                    "PolicyName": "DynamoDBWriteCapacityUtilization:table/
MusicCollection",

```

```

        "TargetTrackingScalingPolicyConfiguration": {
            "TargetValue": 80.0
        }
    ],
    },
    "ReplicaStatus": "ACTIVE"
},
{
    "RegionName": "us-east-2",
    "GlobalSecondaryIndexes": [],
    "ReplicaProvisionedReadCapacityAutoScalingSettings": {
        "MinimumUnits": 5,
        "MaximumUnits": 40000,
        "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",
        "ScalingPolicies": [
            {
                "PolicyName": "DynamoDBReadCapacityUtilization:table/
MusicCollection",
                "TargetTrackingScalingPolicyConfiguration": {
                    "TargetValue": 70.0
                }
            }
        ],
    },
    "ReplicaProvisionedWriteCapacityAutoScalingSettings": {
        "MinimumUnits": 10,
        "MaximumUnits": 100,
        "AutoScalingRoleArn": "arn:aws:iam::123456789012:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable",
        "ScalingPolicies": [
            {
                "PolicyName": "DynamoDBWriteCapacityUtilization:table/
MusicCollection",
                "TargetTrackingScalingPolicyConfiguration": {
                    "TargetValue": 80.0
                }
            }
        ],
    },
    "ReplicaStatus": "ACTIVE"
}

```

```

    }
  ]
}

```

For more information, see [DynamoDB Global Tables](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [UpdateTableReplicaAutoScaling](#) in *AWS CLI Command Reference*.

update-table

The following code example shows how to use `update-table`.

AWS CLI

Example 1: To modify a table's billing mode

The following `update-table` example increases the provisioned read and write capacity on the `MusicCollection` table.

```

aws dynamodb update-table \
  --table-name MusicCollection \
  --billing-mode PROVISIONED \
  --provisioned-throughput ReadCapacityUnits=15,WriteCapacityUnits=10

```

Output:

```

{
  "TableDescription": {
    "AttributeDefinitions": [
      {
        "AttributeName": "AlbumTitle",
        "AttributeType": "S"
      },
      {
        "AttributeName": "Artist",
        "AttributeType": "S"
      },
      {
        "AttributeName": "SongTitle",
        "AttributeType": "S"
      }
    ]
  }
}

```

```

    ],
    "TableName": "MusicCollection",
    "KeySchema": [
      {
        "AttributeName": "Artist",
        "KeyType": "HASH"
      },
      {
        "AttributeName": "SongTitle",
        "KeyType": "RANGE"
      }
    ],
    "TableStatus": "UPDATING",
    "CreationDateTime": "2020-05-26T15:59:49.473000-07:00",
    "ProvisionedThroughput": {
      "LastIncreaseDateTime": "2020-07-28T13:18:18.921000-07:00",
      "NumberOfDecreasesToday": 0,
      "ReadCapacityUnits": 15,
      "WriteCapacityUnits": 10
    },
    "TableSizeBytes": 182,
    "ItemCount": 2,
    "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection",
    "TableId": "abcd0123-01ab-23cd-0123-abcdef123456",
    "BillingModeSummary": {
      "BillingMode": "PROVISIONED",
      "LastUpdateToPayPerRequestDateTime": "2020-07-28T13:14:48.366000-07:00"
    }
  }
}

```

For more information, see [Updating a Table](#) in the *Amazon DynamoDB Developer Guide*.

Example 2: To create a global secondary index

The following example adds a global secondary index to the MusicCollection table.

```

aws dynamodb update-table \
  --table-name MusicCollection \
  --attribute-definitions AttributeName=AlbumTitle,AttributeType=S \
  --global-secondary-index-updates file://gsi-updates.json

```

Contents of `gsi-updates.json`:

```
[
  {
    "Create": {
      "IndexName": "AlbumTitle-index",
      "KeySchema": [
        {
          "AttributeName": "AlbumTitle",
          "KeyType": "HASH"
        }
      ],
      "ProvisionedThroughput": {
        "ReadCapacityUnits": 10,
        "WriteCapacityUnits": 10
      },
      "Projection": {
        "ProjectionType": "ALL"
      }
    }
  }
]
```

Output:

```
{
  "TableDescription": {
    "AttributeDefinitions": [
      {
        "AttributeName": "AlbumTitle",
        "AttributeType": "S"
      },
      {
        "AttributeName": "Artist",
        "AttributeType": "S"
      },
      {
        "AttributeName": "SongTitle",
        "AttributeType": "S"
      }
    ],
    "TableName": "MusicCollection",
    "KeySchema": [
      {
        "AttributeName": "Artist",
```



```
        "KeyType": "HASH"
      },
      {
        "AttributeName": "SongTitle",
        "KeyType": "RANGE"
      }
    ],
    "TableStatus": "UPDATING",
    "CreationDateTime": "2020-05-26T15:59:49.473000-07:00",
    "ProvisionedThroughput": {
      "LastIncreaseDateTime": "2020-07-28T12:59:17.537000-07:00",
      "NumberOfDecreasesToday": 0,
      "ReadCapacityUnits": 15,
      "WriteCapacityUnits": 10
    },
    "TableSizeBytes": 182,
    "ItemCount": 2,
    "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection",
    "TableId": "abcd0123-01ab-23cd-0123-abcdef123456",
    "BillingModeSummary": {
      "BillingMode": "PROVISIONED",
      "LastUpdateToPayPerRequestDateTime": "2020-07-28T13:14:48.366000-07:00"
    },
    "GlobalSecondaryIndexes": [
      {
        "IndexName": "AlbumTitle-index",
        "KeySchema": [
          {
            "AttributeName": "AlbumTitle",
            "KeyType": "HASH"
          }
        ],
        "Projection": {
          "ProjectionType": "ALL"
        },
        "IndexStatus": "CREATING",
        "Backfilling": false,
        "ProvisionedThroughput": {
          "NumberOfDecreasesToday": 0,
          "ReadCapacityUnits": 10,
          "WriteCapacityUnits": 10
        },
        "IndexSizeBytes": 0,
        "ItemCount": 0,
```

```

        "IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/index/AlbumTitle-index"
    }
]
}
}

```

For more information, see [Updating a Table](#) in the *Amazon DynamoDB Developer Guide*.

Example 3: To enable DynamoDB Streams on a table

The following command enables DynamoDB Streams on the MusicCollection table.

```

aws dynamodb update-table \
  --table-name MusicCollection \
  --stream-specification StreamEnabled=true,StreamViewType=NEW_IMAGE

```

Output:

```

{
  "TableDescription": {
    "AttributeDefinitions": [
      {
        "AttributeName": "AlbumTitle",
        "AttributeType": "S"
      },
      {
        "AttributeName": "Artist",
        "AttributeType": "S"
      },
      {
        "AttributeName": "SongTitle",
        "AttributeType": "S"
      }
    ],
    "TableName": "MusicCollection",
    "KeySchema": [
      {
        "AttributeName": "Artist",
        "KeyType": "HASH"
      },
      {
        "AttributeName": "SongTitle",

```

```
        "KeyType": "RANGE"
    }
],
"TableStatus": "UPDATING",
"CreationDateTime": "2020-05-26T15:59:49.473000-07:00",
"ProvisionedThroughput": {
    "LastIncreaseDateTime": "2020-07-28T12:59:17.537000-07:00",
    "NumberOfDecreasesToday": 0,
    "ReadCapacityUnits": 15,
    "WriteCapacityUnits": 10
},
"TableSizeBytes": 182,
"ItemCount": 2,
"TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection",
"TableId": "abcd0123-01ab-23cd-0123-abcdef123456",
"BillingModeSummary": {
    "BillingMode": "PROVISIONED",
    "LastUpdateToPayPerRequestDateTime": "2020-07-28T13:14:48.366000-07:00"
},
"LocalSecondaryIndexes": [
    {
        "IndexName": "AlbumTitleIndex",
        "KeySchema": [
            {
                "AttributeName": "Artist",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "AlbumTitle",
                "KeyType": "RANGE"
            }
        ],
        "Projection": {
            "ProjectionType": "INCLUDE",
            "NonKeyAttributes": [
                "Year",
                "Genre"
            ]
        },
        "IndexSizeBytes": 139,
        "ItemCount": 2,
        "IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/index/AlbumTitleIndex"
    }
]
```

```

    ],
    "GlobalSecondaryIndexes": [
      {
        "IndexName": "AlbumTitle-index",
        "KeySchema": [
          {
            "AttributeName": "AlbumTitle",
            "KeyType": "HASH"
          }
        ],
        "Projection": {
          "ProjectionType": "ALL"
        },
        "IndexStatus": "ACTIVE",
        "ProvisionedThroughput": {
          "NumberOfDecreasesToday": 0,
          "ReadCapacityUnits": 10,
          "WriteCapacityUnits": 10
        },
        "IndexSizeBytes": 0,
        "ItemCount": 0,
        "IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/index/AlbumTitle-index"
      }
    ],
    "StreamSpecification": {
      "StreamEnabled": true,
      "StreamViewType": "NEW_IMAGE"
    },
    "LatestStreamLabel": "2020-07-28T21:53:39.112",
    "LatestStreamArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/stream/2020-07-28T21:53:39.112"
  }
}

```

For more information, see [Updating a Table](#) in the *Amazon DynamoDB Developer Guide*.

Example 4: To enable server-side encryption

The following example enables server-side encryption on the MusicCollection table.

```

aws dynamodb update-table \
  --table-name MusicCollection \

```

```
--sse-specification Enabled=true,SSEType=KMS
```

Output:

```
{
  "TableDescription": {
    "AttributeDefinitions": [
      {
        "AttributeName": "AlbumTitle",
        "AttributeType": "S"
      },
      {
        "AttributeName": "Artist",
        "AttributeType": "S"
      },
      {
        "AttributeName": "SongTitle",
        "AttributeType": "S"
      }
    ],
    "TableName": "MusicCollection",
    "KeySchema": [
      {
        "AttributeName": "Artist",
        "KeyType": "HASH"
      },
      {
        "AttributeName": "SongTitle",
        "KeyType": "RANGE"
      }
    ],
    "TableStatus": "ACTIVE",
    "CreationDateTime": "2020-05-26T15:59:49.473000-07:00",
    "ProvisionedThroughput": {
      "LastIncreaseDateTime": "2020-07-28T12:59:17.537000-07:00",
      "NumberOfDecreasesToday": 0,
      "ReadCapacityUnits": 15,
      "WriteCapacityUnits": 10
    },
    "TableSizeBytes": 182,
    "ItemCount": 2,
    "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/MusicCollection",
    "TableId": "abcd0123-01ab-23cd-0123-abcdef123456",
  }
}
```

```
"BillingModeSummary": {
  "BillingMode": "PROVISIONED",
  "LastUpdateToPayPerRequestDateTime": "2020-07-28T13:14:48.366000-07:00"
},
"LocalSecondaryIndexes": [
  {
    "IndexName": "AlbumTitleIndex",
    "KeySchema": [
      {
        "AttributeName": "Artist",
        "KeyType": "HASH"
      },
      {
        "AttributeName": "AlbumTitle",
        "KeyType": "RANGE"
      }
    ],
    "Projection": {
      "ProjectionType": "INCLUDE",
      "NonKeyAttributes": [
        "Year",
        "Genre"
      ]
    },
    "IndexSizeBytes": 139,
    "ItemCount": 2,
    "IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/index/AlbumTitleIndex"
  }
],
"GlobalSecondaryIndexes": [
  {
    "IndexName": "AlbumTitle-index",
    "KeySchema": [
      {
        "AttributeName": "AlbumTitle",
        "KeyType": "HASH"
      }
    ],
    "Projection": {
      "ProjectionType": "ALL"
    },
    "IndexStatus": "ACTIVE",
    "ProvisionedThroughput": {
```

```

        "NumberOfDecreasesToday": 0,
        "ReadCapacityUnits": 10,
        "WriteCapacityUnits": 10
    },
    "IndexSizeBytes": 0,
    "ItemCount": 0,
    "IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/index/AlbumTitle-index"
    }
],
"StreamSpecification": {
    "StreamEnabled": true,
    "StreamViewType": "NEW_IMAGE"
},
"LatestStreamLabel": "2020-07-28T21:53:39.112",
"LatestStreamArn": "arn:aws:dynamodb:us-west-2:123456789012:table/
MusicCollection/stream/2020-07-28T21:53:39.112",
"SSEDescription": {
    "Status": "UPDATING"
}
}
}
}

```

For more information, see [Updating a Table](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [UpdateTable](#) in *AWS CLI Command Reference*.

update-time-to-live

The following code example shows how to use `update-time-to-live`.

AWS CLI

To update Time to Live settings on a table

The following `update-time-to-live` example enables Time to Live on the specified table.

```

aws dynamodb update-time-to-live \
  --table-name MusicCollection \
  --time-to-live-specification Enabled=true,AttributeName=ttl

```

Output:

```
{
  "TimeToLiveSpecification": {
    "Enabled": true,
    "AttributeName": "ttl"
  }
}
```

For more information, see [Time to Live](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [UpdateTimeToLive](#) in *AWS CLI Command Reference*.

DynamoDB Streams examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with DynamoDB Streams.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

describe-stream

The following code example shows how to use `describe-stream`.

AWS CLI

To get information about a DynamoDB stream

The following `describe-stream` command displays information about the specific DynamoDB stream.


```
aws dynamodbstreams describe-stream \  
  --stream-arn arn:aws:dynamodb:us-west-1:123456789012:table/Music/  
stream/2019-10-22T18:02:01.576
```

Output:

```
{  
  "StreamDescription": {  
    "StreamArn": "arn:aws:dynamodb:us-west-1:123456789012:table/Music/  
stream/2019-10-22T18:02:01.576",  
    "StreamLabel": "2019-10-22T18:02:01.576",  
    "StreamStatus": "ENABLED",  
    "StreamViewType": "NEW_AND_OLD_IMAGES",  
    "CreationRequestDateTime": 1571767321.571,  
    "TableName": "Music",  
    "KeySchema": [  
      {  
        "AttributeName": "Artist",  
        "KeyType": "HASH"  
      },  
      {  
        "AttributeName": "SongTitle",  
        "KeyType": "RANGE"  
      }  
    ],  
    "Shards": [  
      {  
        "ShardId": "shardId-00000001571767321804-697ce3d2",  
        "SequenceNumberRange": {  
          "StartingSequenceNumber": "40000000000000642977831",  
          "EndingSequenceNumber": "40000000000000642977831"  
        }  
      },  
      {  
        "ShardId": "shardId-00000001571780995058-40810d86",  
        "SequenceNumberRange": {  
          "StartingSequenceNumber": "757400000000005655171150"  
        },  
        "ParentShardId": "shardId-00000001571767321804-697ce3d2"  
      }  
    ]  
  }  
}
```

```
}

```

For more information, see [Capturing Table Activity with DynamoDB Streams](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [DescribeStream](#) in *AWS CLI Command Reference*.

get-records

The following code example shows how to use `get-records`.

AWS CLI

To get records from a Dynamodb stream

The following `get-records` command retrieves records using the specified Amazon DynamoDB shard iterator.

```
aws dynamodbstreams get-records \
  --shard-iterator "arn:aws:dynamodb:us-west-1:123456789012:table/Music/
stream/2019-10-22T18:02:01.576|1|
AAAAAAAAAAGgM3YZ89vLZZxjmoQeo33r9M4x3+zmmTLsiL86MfrF4+B4EbsByi52InVmi0Nmy6xVW4IRcIIbs1z07MNI
+CjNPlqQjnyRSAnf0wWmKhL1/KNParWSfz2odf780o00bIDIWRRMkt7+Hyzh9SD
+hFxFawR5C7QI10XPc8mRBfNIazfrVCjJK8/jsjCzsqNyXKzJbhh+GXCoxYN
+Kpmg4nyj1EAsYhbGL35muvHFohjcyuynbsczbWaxNfThDwRAYvoTmc8XhHKtAWUbJiaVd8ZPtQwDsThCrmDRPI dmTRG
+w/1EGS05ha1qNP+Vl4+tuHz2TRnhnJo/pny9GI/yGpce97mWvSPr5KPwy+DtcM5BHayBs
+PVYHITaTliInFlT
+LCwvaz1QH3MY3b8A05Z800wjpkM60iQqtMeDwN4NX6FrcxR34JoFKGsgR8XkHVJzz2xr1xqSJ12ycpNTyHnndusw==
```

Output:

```
{
  "Records": [
    {
      "eventID": "c3b5d798eef6215d42f8137b19a88e50",
      "eventName": "INSERT",
      "eventVersion": "1.1",
      "eventSource": "aws:dynamodb",
      "awsRegion": "us-west-1",
      "dynamodb": {
        "ApproximateCreationDateTime": 1571849028.0,
        "Keys": {
          "Artist": {
```

```

        "S": "No One You Know"
      },
      "SongTitle": {
        "S": "Call Me Today"
      }
    },
    "NewImage": {
      "AlbumTitle": {
        "S": "Somewhat Famous"
      },
      "Artist": {
        "S": "No One You Know"
      },
      "Awards": {
        "N": "1"
      },
      "SongTitle": {
        "S": "Call Me Today"
      }
    },
    "SequenceNumber": "700000000013256296913",
    "SizeBytes": 119,
    "StreamViewType": "NEW_AND_OLD_IMAGES"
  }
},
{
  "eventID": "878960a6967867e2da16b27380a27328",
  "eventName": "INSERT",
  "eventVersion": "1.1",
  "eventSource": "aws:dynamodb",
  "awsRegion": "us-west-1",
  "dynamodb": {
    "ApproximateCreationDateTime": 1571849029.0,
    "Keys": {
      "Artist": {
        "S": "Acme Band"
      },
      "SongTitle": {
        "S": "Happy Day"
      }
    },
    "NewImage": {
      "AlbumTitle": {
        "S": "Songs About Life"
      }
    }
  }
}

```

```
    },
    "Artist": {
      "S": "Acme Band"
    },
    "Awards": {
      "N": "10"
    },
    "SongTitle": {
      "S": "Happy Day"
    }
  },
  "SequenceNumber": "800000000013256297217",
  "SizeBytes": 100,
  "StreamViewType": "NEW_AND_OLD_IMAGES"
}
},
{
  "eventID": "520fabde080e159fc3710b15ee1d4daa",
  "eventName": "MODIFY",
  "eventVersion": "1.1",
  "eventSource": "aws:dynamodb",
  "awsRegion": "us-west-1",
  "dynamodb": {
    "ApproximateCreationDateTime": 1571849734.0,
    "Keys": {
      "Artist": {
        "S": "Acme Band"
      },
      "SongTitle": {
        "S": "Happy Day"
      }
    },
    "NewImage": {
      "AlbumTitle": {
        "S": "Updated Album Title"
      },
      "Artist": {
        "S": "Acme Band"
      },
      "Awards": {
        "N": "10"
      },
      "SongTitle": {
        "S": "Happy Day"
      }
    }
  }
}
```

```

    }
  },
  "OldImage": {
    "AlbumTitle": {
      "S": "Songs About Life"
    },
    "Artist": {
      "S": "Acme Band"
    },
    "Awards": {
      "N": "10"
    },
    "SongTitle": {
      "S": "Happy Day"
    }
  },
  "SequenceNumber": "900000000013256687845",
  "SizeBytes": 170,
  "StreamViewType": "NEW_AND_OLD_IMAGES"
}
],
"NextShardIterator": "arn:aws:dynamodb:us-west-1:123456789012:table/
Music/stream/2019-10-23T16:41:08.740|1|AAAAAAAAAAAEhEI04jkFLW
+LK0wivjT8d/IHEh3iExV2xK00aTxEzVy1C1C7Kbb5+Z0W6bT9VQ2n1/
mrs7+PRia0ZCHJu7JHJVW7zlsq0i/ges3fw8GYEymyL+piEk35cx67rQqwKKyq
+Q6w9JyjreIOj4F2lWLv261BwRTrIYC4IB7C3BZZK4715QwYdDxNdVHiSBRZX8UqoS6W0t0F87xZLNB9F/
NhYBLXi/wcGvAcBcC0TNI0H+N0Nqwt0B/
FGckNrf8YZ0xRoNN6RgGuVWHF3px0hxEJeFZoSoJTIKeG9YcYxzi5Ci/
mhdtm7tBXnbw5c6xmsGsBqTirNjldyJLcWl8Cl0U0LX63Ufo/5QliztcjEbKsQe28x8LM8o7VH1Is0fF/
ITt8awSA4igyJS0P87GN8Qri8kj8iaE35805jBHWf2wvwT6Iy2xGrR2r2HzYps9dwG0arVdEITaJfWzNoL4HajMhmREZ
+V04i1YIeHMXJfcwetNRuIbdQXfJht2NQZa4PVV6iknY6d19MrdbSTMKoqAuvp6g3Q2jH4t7GKCLWgodcPAn8g5+43Da
}

```

For more information, see [Capturing Table Activity with DynamoDB Streams](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [GetRecords](#) in *AWS CLI Command Reference*.

get-shard-iterator

The following code example shows how to use `get-shard-iterator`.

AWS CLI

To get a shard iterator

The following `get-shard-iterator` command retrieves a shard iterator for the specified shard.

```
aws dynamodbstreams get-shard-iterator \  
  --stream-arn arn:aws:dynamodb:us-west-1:12356789012:table/Music/  
stream/2019-10-22T18:02:01.576 \  
  --shard-id shardId-00000001571780995058-40810d86 \  
  --shard-iterator-type LATEST
```

Output:

```
{  
  "ShardIterator": "arn:aws:dynamodb:us-west-1:123456789012:table/Music/  
stream/2019-10-22T18:02:01.576|1|  
AAAAAAAAAAGgM3YZ89vLZZxjmoQeo33r9M4x3+zmmTLsiL86MfrF4+B4EbsByi52InVmi0Nmy6xVW4IRcIIbs1z07MNI  
+CjNPlqQjnyRSAnf0wWmKhL1/KNParWSfz2odf780o00bIDIWRRMkt7+Hyzh9SD  
+hFxFAWR5C7QI10XPc8mRBfNIazfrVCjJK8/jsjCzsqNyXKzJbhh+GXCoxYN  
+Kpmg4nyj1EAsYhbGL35muvHFoHjcyuynbsczbWaXNfThDwRAYvoTmc8XhHKtAWUbJiaVd8ZPtQwDsThCrmDRPI dmTRG  
+w/1EGS05ha1qNP+V14+tuHz2TRnhnJo/pny9GI/yGpce97mWvSPr5KPwy+DtcM5BHayBs  
+PVYHITaTliInFlT  
+LCwvaz1QH3MY3b8A05Z800wjpktm60iQqtMeDwN4NX6FrcxR34JoFKGsgR8XkHVJzz2xr1xqSJ12ycpNTyHnndusw=  
}
```

For more information, see [Capturing Table Activity with DynamoDB Streams](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [GetShardIterator](#) in *AWS CLI Command Reference*.

list-streams

The following code example shows how to use `list-streams`.

AWS CLI

To list DynamoDB streams

The following `list-streams` command lists all existing Amazon DynamoDB streams within the default AWS Region.

```
aws dynamodbstreams list-streams
```

Output:

```
{
  "Streams": [
    {
      "StreamArn": "arn:aws:dynamodb:us-west-1:123456789012:table/Music/stream/2019-10-22T18:02:01.576",
      "TableName": "Music",
      "StreamLabel": "2019-10-22T18:02:01.576"
    }
  ]
}
```

For more information, see [Capturing Table Activity with DynamoDB Streams](#) in the *Amazon DynamoDB Developer Guide*.

- For API details, see [ListStreams](#) in *AWS CLI Command Reference*.

Amazon EC2 examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon EC2.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

accept-address-transfer

The following code example shows how to use `accept-address-transfer`.

AWS CLI

To accept an Elastic IP address transferred to your account

The following `accept-address-transfer` example accepts the transfer of the specified Elastic IP address to your account.

```
aws ec2 accept-address-transfer \  
  --address 100.21.184.216
```

Output:

```
{  
  "AddressTransfer": {  
    "PublicIp": "100.21.184.216",  
    "AllocationId": "eipalloc-09ad461b0d03f6aaf",  
    "TransferAccountId": "123456789012",  
    "TransferOfferExpirationTimestamp": "2023-02-22T20:51:10.000Z",  
    "TransferOfferAcceptedTimestamp": "2023-02-22T22:52:54.000Z",  
    "AddressTransferStatus": "accepted"  
  }  
}
```

For more information, see [Transfer Elastic IP addresses](#) in the *Amazon VPC User Guide*.

- For API details, see [AcceptAddressTransfer](#) in *AWS CLI Command Reference*.

accept-reserved-instances-exchange-quote

The following code example shows how to use `accept-reserved-instances-exchange-quote`.

AWS CLI

To perform a Convertible Reserved Instance exchange

This example performs an exchange of the specified Convertible Reserved Instances.

Command:

```
aws ec2 accept-reserved-instances-exchange-quote --reserved-instance-ids 7b8750c3-397e-4da4-bbcb-a45ebexample --target-configurations OfferingId=b747b472-423c-48f3-8cee-679bcexample
```

Output:

```
{
  "ExchangeId": "riex-e68ed3c1-8bc8-4c17-af77-811afexample"
}
```

- For API details, see [AcceptReservedInstancesExchangeQuote](#) in *AWS CLI Command Reference*.

accept-transit-gateway-peering-attachment

The following code example shows how to use `accept-transit-gateway-peering-attachment`.

AWS CLI**To accept a transit gateway peering attachment**

The following `accept-transit-gateway-peering-attachment` example accepts the specified transit gateway peering attachment. The `--region` parameter specifies the Region that the accepter transit gateway is located in.

```
aws ec2 accept-transit-gateway-peering-attachment \
  --transit-gateway-attachment-id tgw-attach-4455667788aabbccd \
  --region us-east-2
```

Output:

```
{
  "TransitGatewayPeeringAttachment": {
    "TransitGatewayAttachmentId": "tgw-attach-4455667788aabbccd",
    "RequesterTgwInfo": {
      "TransitGatewayId": "tgw-123abc05e04123abc",
      "OwnerId": "123456789012",
      "Region": "us-west-2"
    }
  },
}
```

```

    "AcceptorTgwInfo": {
      "TransitGatewayId": "tgw-11223344aabbcc112",
      "OwnerId": "123456789012",
      "Region": "us-east-2"
    },
    "State": "pending",
    "CreationTime": "2019-12-09T11:38:31.000Z"
  }
}

```

For more information, see [Transit Gateway Peering Attachments](#) in the *Transit Gateways Guide*.

- For API details, see [AcceptTransitGatewayPeeringAttachment](#) in *AWS CLI Command Reference*.

accept-transit-gateway-vpc-attachment

The following code example shows how to use `accept-transit-gateway-vpc-attachment`.

AWS CLI

To accept a request to attach a VPC to a transit gateway.

The following `accept-transit-gateway-vpc-attachment` example accepts the request for the specified attachment.

```

aws ec2 accept-transit-gateway-vpc-attachment \
  --transit-gateway-attachment-id tgw-attach-0a34fe6b4fEXAMPLE

```

Output:

```

{
  "TransitGatewayVpcAttachment": {
    "TransitGatewayAttachmentId": "tgw-attach-0a34fe6b4fEXAMPLE",
    "TransitGatewayId": "tgw-0262a0e521EXAMPLE",
    "VpcId": "vpc-07e8ffd50fEXAMPLE",
    "VpcOwnerId": "123456789012",
    "State": "pending",
    "SubnetIds": [
      "subnet-0752213d59EXAMPLE"
    ],
    "CreationTime": "2019-07-10T17:33:46.000Z",
    "Options": {
      "DnsSupport": "enable",

```

```
        "Ipv6Support": "disable"
      }
    }
  }
```

For more information, see [Transit Gateway Attachments to a VPC](#) in the *Transit Gateways Guide*.

- For API details, see [AcceptTransitGatewayVpcAttachment](#) in *AWS CLI Command Reference*.

accept-vpc-endpoint-connections

The following code example shows how to use `accept-vpc-endpoint-connections`.

AWS CLI

To accept an interface endpoint connection request

This example accepts the specified endpoint connection request for the specified endpoint service.

Command:

```
aws ec2 accept-vpc-endpoint-connections --service-id vpce-svc-03d5ebb7d9579a2b3 --
vpc-endpoint-ids vpce-0c1308d7312217abc
```

Output:

```
{
  "Unsuccessful": []
}
```

- For API details, see [AcceptVpcEndpointConnections](#) in *AWS CLI Command Reference*.

accept-vpc-peering-connection

The following code example shows how to use `accept-vpc-peering-connection`.

AWS CLI

To accept a VPC peering connection

This example accepts the specified VPC peering connection request.

Command:

```
aws ec2 accept-vpc-peering-connection --vpc-peering-connection-id pcx-1a2b3c4d
```

Output:

```
{
  "VpcPeeringConnection": {
    "Status": {
      "Message": "Provisioning",
      "Code": "provisioning"
    },
    "Tags": [],
    "AccepterVpcInfo": {
      "OwnerId": "444455556666",
      "VpcId": "vpc-44455566",
      "CidrBlock": "10.0.1.0/28"
    },
    "VpcPeeringConnectionId": "pcx-1a2b3c4d",
    "RequesterVpcInfo": {
      "OwnerId": "444455556666",
      "VpcId": "vpc-111abc45",
      "CidrBlock": "10.0.0.0/28"
    }
  }
}
```

- For API details, see [AcceptVpcPeeringConnection](#) in *AWS CLI Command Reference*.

advertise-byoip-cidr

The following code example shows how to use `advertise-byoip-cidr`.

AWS CLI**To advertise an address range**

The following `advertise-byoip-cidr` example advertises the specified public IPv4 address range.

```
aws ec2 advertise-byoip-cidr \
```

```
--cidr 203.0.113.25/24
```

Output:

```
{
  "ByoipCidr": {
    "Cidr": "203.0.113.25/24",
    "StatusMessage": "ipv4pool-ec2-1234567890abcdef0",
    "State": "provisioned"
  }
}
```

- For API details, see [AdvertiseByoipCidr](#) in *AWS CLI Command Reference*.

allocate-address

The following code example shows how to use `allocate-address`.

AWS CLI**Example 1: To allocate an Elastic IP address from Amazon's address pool**

The following `allocate-address` example allocates an Elastic IP address. Amazon EC2 selects the address from Amazon's address pool.

```
aws ec2 allocate-address
```

Output:

```
{
  "PublicIp": "70.224.234.241",
  "AllocationId": "eipalloc-01435ba59eEXAMPLE",
  "PublicIpv4Pool": "amazon",
  "NetworkBorderGroup": "us-west-2",
  "Domain": "vpc"
}
```

For more information, see [Elastic IP addresses](#) in the *Amazon EC2 User Guide*.

Example 2: To allocate an Elastic IP address and associate it with a network border group

The following `allocate-address` example allocates an Elastic IP address and associates it with the specified network border group.

```
aws ec2 allocate-address \  
  --network-border-group us-west-2-lax-1
```

Output:

```
{  
  "PublicIp": "70.224.234.241",  
  "AllocationId": "eipalloc-e03dd489ceEXAMPLE",  
  "PublicIpv4Pool": "amazon",  
  "NetworkBorderGroup": "us-west-2-lax-1",  
  "Domain": "vpc"  
}
```

For more information, see [Elastic IP addresses](#) in the *Amazon EC2 User Guide*.

Example 3: To allocate an Elastic IP address from an address pool that you own

The following `allocate-address` example allocates an Elastic IP address from an address pool that you have brought to your Amazon Web Services account. Amazon EC2 selects the address from the address pool.

```
aws ec2 allocate-address \  
  --public-ipv4-pool ipv4pool-ec2-1234567890abcdef0
```

Output:

```
{  
  "AllocationId": "eipalloc-02463d08ceEXAMPLE",  
  "NetworkBorderGroup": "us-west-2",  
  "CustomerOwnedIp": "18.218.95.81",  
  "CustomerOwnedIpv4Pool": "ipv4pool-ec2-1234567890abcdef0",  
  "Domain": "vpc"  
  "NetworkBorderGroup": "us-west-2",  
}
```

For more information, see [Elastic IP addresses](#) in the *Amazon EC2 User Guide*.

- For API details, see [AllocateAddress](#) in *AWS CLI Command Reference*.

allocate-hosts

The following code example shows how to use `allocate-hosts`.

AWS CLI

Example 1: To allocate a Dedicated Host

The following `allocate-hosts` example allocates a single Dedicated Host in the `eu-west-1a` Availability Zone, onto which you can launch `m5.large` instances. By default, the Dedicated Host accepts only target instance launches, and does not support host recovery.

```
aws ec2 allocate-hosts \  
  --instance-type m5.large \  
  --availability-zone eu-west-1a \  
  --quantity 1
```

Output:

```
{  
  "HostIds": [  
    "h-07879acf49EXAMPLE"  
  ]  
}
```

Example 2: To allocate a Dedicated Host with auto-placement and host recovery enabled

The following `allocate-hosts` example allocates a single Dedicated Host in the `eu-west-1a` Availability Zone with auto-placement and host recovery enabled.

```
aws ec2 allocate-hosts \  
  --instance-type m5.large \  
  --availability-zone eu-west-1a \  
  --auto-placement on \  
  --host-recovery on \  
  --quantity 1
```

Output:

```
{  
  "HostIds": [  
    "h-07879acf49EXAMPLE"  
  ]  
}
```

```
        "h-07879acf49EXAMPLE"  
    ]  
}
```

Example 3: To allocate a Dedicated Host with tags

The following `allocate-hosts` example allocates a single Dedicated Host and applies a tag with a key named `purpose` and a value of `production`.

```
aws ec2 allocate-hosts \  
  --instance-type m5.large \  
  --availability-zone eu-west-1a \  
  --quantity 1 \  
  --tag-specifications 'ResourceType=dedicated-  
host,Tags={Key=purpose,Value=production}'
```

Output:

```
{  
  "HostIds": [  
    "h-07879acf49EXAMPLE"  
  ]  
}
```

For more information, see [Allocating Dedicated Hosts](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [AllocateHosts](#) in *AWS CLI Command Reference*.

allocate-ipam-pool-cidr

The following code example shows how to use `allocate-ipam-pool-cidr`.

AWS CLI

To allocate a CIDR from an IPAM pool

The following `allocate-ipam-pool-cidr` example allocates a CIDR from an IPAM pool.

(Linux):

```
aws ec2 allocate-ipam-pool-cidr \  

```



```
--ipam-pool-id ipam-pool-0533048da7d823723 \  
--netmask-length 24
```

(Windows):

```
aws ec2 allocate-ipam-pool-cidr ^  
--ipam-pool-id ipam-pool-0533048da7d823723 ^  
--netmask-length 24
```

Output:

```
{  
  "IpamPoolAllocation": {  
    "Cidr": "10.0.0.0/24",  
    "IpamPoolAllocationId": "ipam-pool-alloc-018ecc28043b54ba38e2cd99943cebfb",  
    "ResourceType": "custom",  
    "ResourceOwner": "123456789012"  
  }  
}
```

For more information, see [Manually allocate a CIDR to a pool to reserve IP address space](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [AllocateIpamPoolCidr](#) in *AWS CLI Command Reference*.

apply-security-groups-to-client-vpn-target-network

The following code example shows how to use `apply-security-groups-to-client-vpn-target-network`.

AWS CLI

To apply security groups to a target network for a Client VPN endpoint

The following `apply-security-groups-to-client-vpn-target-network` example applies security group `sg-01f6e627a89f4db32` to the association between the specified target network and Client VPN endpoint.

```
aws ec2 apply-security-groups-to-client-vpn-target-network \  
--security-group-ids sg-01f6e627a89f4db32 \  
--vpc-id vpc-0e2110c2f324332e0 \  

```

```
--client-vpn-endpoint-id cvpn-endpoint-123456789123abcde
```

Output:

```
{
  "SecurityGroupIds": [
    "sg-01f6e627a89f4db32"
  ]
}
```

For more information, see [Target Networks](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [ApplySecurityGroupsToClientVpnTargetNetwork](#) in *AWS CLI Command Reference*.

assign-ipv6-addresses

The following code example shows how to use `assign-ipv6-addresses`.

AWS CLI**To assign specific IPv6 addresses to a network interface**

This example assigns the specified IPv6 addresses to the specified network interface.

Command:

```
aws ec2 assign-ipv6-addresses --network-interface-id eni-38664473 --ipv6-addresses
2001:db8:1234:1a00:3304:8879:34cf:4071 2001:db8:1234:1a00:9691:9503:25ad:1761
```

Output:

```
{
  "AssignedIpv6Addresses": [
    "2001:db8:1234:1a00:3304:8879:34cf:4071",
    "2001:db8:1234:1a00:9691:9503:25ad:1761"
  ],
  "NetworkInterfaceId": "eni-38664473"
}
```

To assign IPv6 addresses that Amazon selects to a network interface

This example assigns two IPv6 addresses to the specified network interface. Amazon automatically assigns these IPv6 addresses from the available IPv6 addresses in the IPv6 CIDR block range of the subnet.

Command:

```
aws ec2 assign-ipv6-addresses --network-interface-id eni-38664473 --ipv6-address-count 2
```

Output:

```
{
  "AssignedIpv6Addresses": [
    "2001:db8:1234:1a00:3304:8879:34cf:4071",
    "2001:db8:1234:1a00:9691:9503:25ad:1761"
  ],
  "NetworkInterfaceId": "eni-38664473"
}
```

- For API details, see [AssignIpv6Addresses](#) in *AWS CLI Command Reference*.

assign-private-ip-addresses

The following code example shows how to use `assign-private-ip-addresses`.

AWS CLI

To assign a specific secondary private IP address a network interface

This example assigns the specified secondary private IP address to the specified network interface. If the command succeeds, no output is returned.

Command:

```
aws ec2 assign-private-ip-addresses --network-interface-id eni-e5aa89a3 --private-ip-addresses 10.0.0.82
```

To assign secondary private IP addresses that Amazon EC2 selects to a network interface

This example assigns two secondary private IP addresses to the specified network interface. Amazon EC2 automatically assigns these IP addresses from the available IP addresses in the

CIDR block range of the subnet the network interface is associated with. If the command succeeds, no output is returned.

Command:

```
aws ec2 assign-private-ip-addresses --network-interface-id eni-e5aa89a3 --secondary-private-ip-address-count 2
```

- For API details, see [AssignPrivateIpAddresses](#) in *AWS CLI Command Reference*.

assign-private-nat-gateway-address

The following code example shows how to use `assign-private-nat-gateway-address`.

AWS CLI

To assign private IP addresses to your private NAT gateway

The following `assign-private-nat-gateway-address` example assigns two private IP addresses to the specified private NAT gateway.

```
aws ec2 assign-private-nat-gateway-address \
  --nat-gateway-id nat-1234567890abcdef0 \
  --private-ip-address-count 2
```

Output:

```
{
  "NatGatewayId": "nat-1234567890abcdef0",
  "NatGatewayAddresses": [
    {
      "NetworkInterfaceId": "eni-0065a61b324d1897a",
      "IsPrimary": false,
      "Status": "assigning"
    },
    {
      "NetworkInterfaceId": "eni-0065a61b324d1897a",
      "IsPrimary": false,
      "Status": "assigning"
    }
  ]
}
```

```
}
```

For more information, see [NAT gateways](#) in the *Amazon VPC User Guide*.

- For API details, see [AssignPrivateNatGatewayAddress](#) in *AWS CLI Command Reference*.

associate-address

The following code example shows how to use `associate-address`.

AWS CLI

To associate an Elastic IP addresses in EC2-Classic

This example associates an Elastic IP address with an instance in EC2-Classic. If the command succeeds, no output is returned.

Command:

```
aws ec2 associate-address --instance-id i-07ffe74c7330ebf53 --public-ip 198.51.100.0
```

To associate an Elastic IP address in EC2-VPC

This example associates an Elastic IP address with an instance in a VPC.

Command:

```
aws ec2 associate-address --instance-id i-0b263919b6498b123 --allocation-id eipalloc-64d5890a
```

Output:

```
{
  "AssociationId": "eipassoc-2bebb745"
}
```

This example associates an Elastic IP address with a network interface.

Command:

```
aws ec2 associate-address --allocation-id eipalloc-64d5890a --network-interface-id eni-1a2b3c4d
```

This example associates an Elastic IP with a private IP address that's associated with a network interface.

Command:

```
aws ec2 associate-address --allocation-id eipalloc-64d5890a --network-interface-id
eni-1a2b3c4d --private-ip-address 10.0.0.85
```

- For API details, see [AssociateAddress](#) in *AWS CLI Command Reference*.

associate-client-vpn-target-network

The following code example shows how to use `associate-client-vpn-target-network`.

AWS CLI

To associate a target network with a Client VPN endpoint

The following `associate-client-vpn-target-network` example associates a subnet with the specified Client VPN endpoint.

```
aws ec2 associate-client-vpn-target-network \
  --subnet-id subnet-0123456789abcabca \
  --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde
```

Output:

```
{
  "AssociationId": "cvpn-assoc-12312312312312312",
  "Status": {
    "Code": "associating"
  }
}
```

For more information, see [Target Networks](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [AssociateClientVpnTargetNetwork](#) in *AWS CLI Command Reference*.

associate-dhcp-options

The following code example shows how to use `associate-dhcp-options`.

AWS CLI

To associate a DHCP options set with your VPC

This example associates the specified DHCP options set with the specified VPC. If the command succeeds, no output is returned.

Command:

```
aws ec2 associate-dhcp-options --dhcp-options-id dopt-d9070ebb --vpc-id vpc-a01106c2
```

To associate the default DHCP options set with your VPC

This example associates the default DHCP options set with the specified VPC. If the command succeeds, no output is returned.

Command:

```
aws ec2 associate-dhcp-options --dhcp-options-id default --vpc-id vpc-a01106c2
```

- For API details, see [AssociateDhcpOptions](#) in *AWS CLI Command Reference*.

associate-iam-instance-profile

The following code example shows how to use `associate-iam-instance-profile`.

AWS CLI

To associate an IAM instance profile with an instance

This example associates an IAM instance profile named `admin-role` with instance `i-123456789abcde123`.

Command:

```
aws ec2 associate-iam-instance-profile --instance-id i-123456789abcde123 --iam-instance-profile Name=admin-role
```

Output:

```
{
```

```

    "IamInstanceProfileAssociation": {
      "InstanceId": "i-123456789abcde123",
      "State": "associating",
      "AssociationId": "iip-assoc-0e7736511a163c209",
      "IamInstanceProfile": {
        "Id": "AIPAJBLK7RKJKWDXVHIEC",
        "Arn": "arn:aws:iam::123456789012:instance-profile/admin-role"
      }
    }
  }
}

```

- For API details, see [AssociateIamInstanceProfile](#) in *AWS CLI Command Reference*.

associate-instance-event-window

The following code example shows how to use `associate-instance-event-window`.

AWS CLI

Example 1: To associate one or more instances with an event window

The following `associate-instance-event-window` example associates one or more instances with an event window.

```

aws ec2 associate-instance-event-window \
  --region us-east-1 \
  --instance-event-window-id iew-0abcdef1234567890 \
  --association-target "InstanceIds=i-1234567890abcdef0,i-0598c7d356eba48d7"

```

Output:

```

{
  "InstanceEventWindow": {
    "InstanceEventWindowId": "iew-0abcdef1234567890",
    "Name": "myEventWindowName",
    "CronExpression": "* 21-23 * * 2,3",
    "AssociationTarget": {
      "InstanceIds": [
        "i-1234567890abcdef0",
        "i-0598c7d356eba48d7"
      ],
      "Tags": [],
    }
  }
}

```



```

        "DedicatedHostIds": []
      },
      "State": "creating"
    }
  }
}

```

For event window constraints, see [Considerations](#) in the Scheduled Events section of the *Amazon EC2 User Guide*.

Example 2: To associate instance tags with an event window

The following `associate-instance-event-window` example associates instance tags with an event window. Enter an `instance-event-window-id` parameter to specify the event window. To associate instance tags, specify the `association-target` parameter, and for the parameter value, specify one or more tags.

```

aws ec2 associate-instance-event-window \
  --region us-east-1 \
  --instance-event-window-id iew-0abcdef1234567890 \
  --association-target "InstanceTags=[{Key=k2,Value=v2},{Key=k1,Value=v1}]"

```

Output:

```

{
  "InstanceEventWindow": {
    "InstanceEventWindowId": "iew-0abcdef1234567890",
    "Name": "myEventWindowName",
    "CronExpression": "* 21-23 * * 2,3",
    "AssociationTarget": {
      "InstanceIds": [],
      "Tags": [
        {
          "Key": "k2",
          "Value": "v2"
        },
        {
          "Key": "k1",
          "Value": "v1"
        }
      ],
      "DedicatedHostIds": []
    }
  },
}

```

```

    "State": "creating"
  }
}

```

For event window constraints, see [Considerations](#) in the Scheduled Events section of the *Amazon EC2 User Guide*.

Example 3: To associate a Dedicated Host with an event window

The following `associate-instance-event-window` example associates a Dedicated Host with an event window. Enter an `instance-event-window-id` parameter to specify the event window. To associate a Dedicated Host, specify the `--association-target` parameter, and for the parameter values, specify one of more Dedicated Host IDs.

```

aws ec2 associate-instance-event-window \
  --region us-east-1 \
  --instance-event-window-id iew-0abcdef1234567890 \
  --association-target "DedicatedHostIds=h-029fa35a02b99801d"

```

Output:

```

{
  "InstanceEventWindow": {
    "InstanceEventWindowId": "iew-0abcdef1234567890",
    "Name": "myEventWindowName",
    "CronExpression": "* 21-23 * * 2,3",
    "AssociationTarget": {
      "InstanceIds": [],
      "Tags": [],
      "DedicatedHostIds": [
        "h-029fa35a02b99801d"
      ]
    },
    "State": "creating"
  }
}

```

For event window constraints, see [Considerations](#) in the Scheduled Events section of the *Amazon EC2 User Guide*.

- For API details, see [AssociateInstanceEventWindow](#) in *AWS CLI Command Reference*.

associate-ipam-resource-discovery

The following code example shows how to use `associate-ipam-resource-discovery`.

AWS CLI

To associate a resource discovery with an IPAM

In this example, you are an IPAM delegated admin and a resource discovery has been created and shared with you by another AWS account so that you can use IPAM to manage and monitor resource CIDRs owned by the other account.

Note

To complete this request, you'll need the resource discovery ID which you can get with [describe-ipam-resource-discoveries](#) and the IPAM ID which you can get with [describe-ipams](#). The resource discovery that you are associating must have first been shared with your account using AWS RAM. The `--region` you enter must match the home Region of the IPAM you are associating it with.

The following `associate-ipam-resource-discovery` example associates a resource discovery with an IPAM.

```
aws ec2 associate-ipam-resource-discovery \
  --ipam-id ipam-005f921c17ebd5107 \
  --ipam-resource-discovery-id ipam-res-disco-03e0406de76a044ee \
  --tag-specifications 'ResourceType=ipam-resource-discovery,Tags=[{Key=cost-
center,Value=cc123}]' \
  --region us-east-1
```

Output:

```
{
  {
    "IpamResourceDiscoveryAssociation": {
      "OwnerId": "320805250157",
      "IpamResourceDiscoveryAssociationId": "ipam-res-disco-
assoc-04382a6346357cf82",
      "IpamResourceDiscoveryAssociationArn": "arn:aws:ec2::320805250157:ipam-
resource-discovery-association/ipam-res-disco-assoc-04382a6346357cf82",
      "IpamResourceDiscoveryId": "ipam-res-disco-0365d2977fc1672fe",
```

```

    "IpamId": "ipam-005f921c17ebd5107",
    "IpamArn": "arn:aws:ec2::320805250157:ipam/ipam-005f921c17ebd5107",
    "IpamRegion": "us-east-1",
    "IsDefault": false,
    "ResourceDiscoveryStatus": "active",
    "State": "associate-in-progress",
    "Tags": []
  }
}

```

Once you associate a resource discovery, you can monitor and/or manage the IP addresses of resources created by the other accounts. For more information, see [Integrate IPAM with accounts outside of your organization](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [AssociateIpamResourceDiscovery](#) in *AWS CLI Command Reference*.

associate-nat-gateway-address

The following code example shows how to use `associate-nat-gateway-address`.

AWS CLI

To associate an Elastic IP address with a public NAT gateway

The following `associate-nat-gateway-address` example associates the specified Elastic IP address with the specified public NAT gateway. AWS automatically assigns a secondary private IPv4 address.

```

aws ec2 associate-nat-gateway-address \
  --nat-gateway-id nat-1234567890abcdef0 \
  --allocation-ids eipalloc-0be6ecac95EXAMPLE

```

Output:

```

{
  "NatGatewayId": "nat-1234567890abcdef0",
  "NatGatewayAddresses": [
    {
      "AllocationId": "eipalloc-0be6ecac95EXAMPLE",
      "NetworkInterfaceId": "eni-09cc4b2558794f7f9",
      "IsPrimary": false,

```

```
    "Status": "associating"
  }
]
}
```

For more information, see [NAT gateways](#) in the *Amazon VPC User Guide*.

- For API details, see [AssociateNatGatewayAddress](#) in *AWS CLI Command Reference*.

associate-route-table

The following code example shows how to use `associate-route-table`.

AWS CLI

To associate a route table with a subnet

This example associates the specified route table with the specified subnet.

Command:

```
aws ec2 associate-route-table --route-table-id rtb-22574640 --subnet-id
subnet-9d4a7b6c
```

Output:

```
{
  "AssociationId": "rtbassoc-781d0d1a"
}
```

- For API details, see [AssociateRouteTable](#) in *AWS CLI Command Reference*.

associate-subnet-cidr-block

The following code example shows how to use `associate-subnet-cidr-block`.

AWS CLI

To associate an IPv6 CIDR block with a subnet

This example associates an IPv6 CIDR block with the specified subnet.

Command:

```
aws ec2 associate-subnet-cidr-block --subnet-id subnet-5f46ec3b --ipv6-cidr-block
2001:db8:1234:1a00::/64
```

Output:

```
{
  "SubnetId": "subnet-5f46ec3b",
  "Ipv6CidrBlockAssociation": {
    "Ipv6CidrBlock": "2001:db8:1234:1a00::/64",
    "AssociationId": "subnet-cidr-assoc-3aa54053",
    "Ipv6CidrBlockState": {
      "State": "associating"
    }
  }
}
```

- For API details, see [AssociateSubnetCidrBlock](#) in *AWS CLI Command Reference*.

associate-transit-gateway-multicast-domain

The following code example shows how to use `associate-transit-gateway-multicast-domain`.

AWS CLI**To associate a transit gateway with a multicast domain**

The following `associate-transit-gateway-multicast-domain` example associates the specified subnet and attachment with the specified multicast domain.

```
aws ec2 associate-transit-gateway-multicast-domain \
  --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef79d6e597 \
  --transit-gateway-attachment-id tgw-attach-028c1dd0f8f5cbe8e \
  --subnet-ids subnet-000de86e3b49c932a \
  --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef7EXAMPLE
```

Output:

```
{
```

```

    "Associations": {
      "TransitGatewayMulticastDomainId": "tgw-mcast-domain-0c4905cef79d6e597",
      "TransitGatewayAttachmentId": "tgw-attach-028c1dd0f8f5cbe8e",
      "ResourceId": "vpc-01128d2c240c09bd5",
      "ResourceType": "vpc",
      "Subnets": [
        {
          "SubnetId": "subnet-000de86e3b49c932a",
          "State": "associating"
        }
      ]
    }
  }
}

```

For more information, see [Managing multicast domains](#) in the *Transit Gateways Guide*.

- For API details, see [AssociateTransitGatewayMulticastDomain](#) in *AWS CLI Command Reference*.

associate-transit-gateway-route-table

The following code example shows how to use `associate-transit-gateway-route-table`.

AWS CLI

To associate a transit gateway route table with a transit gateway attachment

The following example associates the specified transit gateway route table with the specified VPC attachment.

```

aws ec2 associate-transit-gateway-route-table \
  --transit-gateway-route-table-id tgw-rtb-002573ed1eEXAMPLE \
  --transit-gateway-attachment-id tgw-attach-0b5968d3b6EXAMPLE

```

Output:

```

{
  "Association": {
    "TransitGatewayRouteTableId": "tgw-rtb-002573ed1eEXAMPLE",
    "TransitGatewayAttachmentId": "tgw-attach-0b5968d3b6EXAMPLE",
    "ResourceId": "vpc-0065acced4EXAMPLE",
    "ResourceType": "vpc",
    "State": "associating"
  }
}

```

```
}
}
```

For more information, see [Associate a Transit Gateway Route Table](#) in the *AWS Transit Gateways Guide*.

- For API details, see [AssociateTransitGatewayRouteTable](#) in *AWS CLI Command Reference*.

associate-vpc-cidr-block

The following code example shows how to use `associate-vpc-cidr-block`.

AWS CLI

Example 1: To associate an Amazon-provided IPv6 CIDR block with a VPC

The following `associate-vpc-cidr-block` example associates an IPv6 CIDR block with the specified VPC.:

```
aws ec2 associate-vpc-cidr-block \
  --amazon-provided-ipv6-cidr-block \
  --ipv6-cidr-block-network-border-group us-west-2-lax-1 \
  --vpc-id vpc-8EXAMPLE
```

Output:

```
{
  "Ipv6CidrBlockAssociation": {
    "AssociationId": "vpc-cidr-assoc-0838ce7d9dEXAMPLE",
    "Ipv6CidrBlockState": {
      "State": "associating"
    },
    "NetworkBorderGroup": "us-west-2-lax-1"
  },
  "VpcId": "vpc-8EXAMPLE"
}
```

Example 2: To associate an additional IPv4 CIDR block with a VPC

The following `associate-vpc-cidr-block` example associates the IPv4 CIDR block `10.2.0.0/16` with the specified VPC.


```
aws ec2 associate-vpc-cidr-block \  
  --vpc-id vpc-1EXAMPLE \  
  --cidr-block 10.2.0.0/16
```

Output:

```
{  
  "CidrBlockAssociation": {  
    "AssociationId": "vpc-cidr-assoc-2EXAMPLE",  
    "CidrBlock": "10.2.0.0/16",  
    "CidrBlockState": {  
      "State": "associating"  
    }  
  },  
  "VpcId": "vpc-1EXAMPLE"  
}
```

- For API details, see [AssociateVpcCidrBlock](#) in *AWS CLI Command Reference*.

attach-classic-link-vpc

The following code example shows how to use `attach-classic-link-vpc`.

AWS CLI

To link (attach) an EC2-Classical instance to a VPC

This example links instance `i-1234567890abcdef0` to VPC `vpc-88888888` through the VPC security group `sg-12312312`.

Command:

```
aws ec2 attach-classic-link-vpc --instance-id i-1234567890abcdef0 --vpc-id  
vpc-88888888 --groups sg-12312312
```

Output:

```
{  
  "Return": true  
}
```

- For API details, see [AttachClassicLinkVpc](#) in *AWS CLI Command Reference*.

attach-internet-gateway

The following code example shows how to use `attach-internet-gateway`.

AWS CLI

To attach an internet gateway to your VPC

The following `attach-internet-gateway` example attaches the specified internet gateway to the specific VPC.

```
aws ec2 attach-internet-gateway \  
  --internet-gateway-id igw-0d0fb496b3EXAMPLE \  
  --vpc-id vpc-0a60eb65b4EXAMPLE
```

This command produces no output.

For more information, see [Internet gateways](#) in the *Amazon VPC User Guide*.

- For API details, see [AttachInternetGateway](#) in *AWS CLI Command Reference*.

attach-network-interface

The following code example shows how to use `attach-network-interface`.

AWS CLI

Example 1: To attach a network interface to an instance

The following `attach-network-interface` example attaches the specified network interface to the specified instance.

```
aws ec2 attach-network-interface \  
  --network-interface-id eni-0dc56a8d4640ad10a \  
  --instance-id i-1234567890abcdef0 \  
  --device-index 1
```

Output:

```
{
  "AttachmentId": "eni-attach-01a8fc87363f07cf9"
}
```

For more information, see [Elastic network interfaces](#) in the *Amazon EC2 User Guide*.

Example 2: To attach a network interface to an instance with multiple network cards

The following `attach-network-interface` example attaches the specified network interface to the specified instance and network card.

```
aws ec2 attach-network-interface \
  --network-interface-id eni-07483b1897541ad83 \
  --instance-id i-01234567890abcdef \
  --network-card-index 1 \
  --device-index 1
```

Output:

```
{
  "AttachmentId": "eni-attach-0fbd7ee87a88cd06c"
}
```

For more information, see [Elastic network interfaces](#) in the *Amazon EC2 User Guide*.

- For API details, see [AttachNetworkInterface](#) in *AWS CLI Command Reference*.

attach-verified-access-trust-provider

The following code example shows how to use `attach-verified-access-trust-provider`.

AWS CLI

To attach a trust provider to an instance

The following `attach-verified-access-trust-provider` example attaches the specified Verified Access trust provider to the specified Verified Access instance.

```
aws ec2 attach-verified-access-trust-provider \
  --verified-access-instance-id vai-0ce000c0b7643abea \
```

```
--verified-access-trust-provider-id vatp-0bb32de759a3e19e7
```

Output:

```
{
  "VerifiedAccessTrustProvider": {
    "VerifiedAccessTrustProviderId": "vatp-0bb32de759a3e19e7",
    "Description": "",
    "TrustProviderType": "user",
    "UserTrustProviderType": "iam-identity-center",
    "PolicyReferenceName": "idc",
    "CreationTime": "2023-08-25T19:00:38",
    "LastUpdatedTime": "2023-08-25T19:00:38"
  },
  "VerifiedAccessInstance": {
    "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
    "Description": "",
    "VerifiedAccessTrustProviders": [
      {
        "VerifiedAccessTrustProviderId": "vatp-0bb32de759a3e19e7",
        "TrustProviderType": "user",
        "UserTrustProviderType": "iam-identity-center"
      }
    ],
    "CreationTime": "2023-08-25T18:27:56",
    "LastUpdatedTime": "2023-08-25T18:27:56"
  }
}
```

For more information, see [Verified Access instances](#) in the *AWS Verified Access User Guide*.

- For API details, see [AttachVerifiedAccessTrustProvider](#) in *AWS CLI Command Reference*.

attach-volume

The following code example shows how to use `attach-volume`.

AWS CLI

To attach a volume to an instance

This example command attaches a volume (`vol-1234567890abcdef0`) to an instance (`i-01474ef662b89480`) as `/dev/sdf`.

Command:

```
aws ec2 attach-volume --volume-id vol-1234567890abcdef0 --instance-id
i-01474ef662b89480 --device /dev/sdf
```

Output:

```
{
  "AttachTime": "YYYY-MM-DDTHH:MM:SS.000Z",
  "InstanceId": "i-01474ef662b89480",
  "VolumeId": "vol-1234567890abcdef0",
  "State": "attaching",
  "Device": "/dev/sdf"
}
```

- For API details, see [AttachVolume](#) in *AWS CLI Command Reference*.

attach-vpn-gateway

The following code example shows how to use `attach-vpn-gateway`.

AWS CLI**To attach a virtual private gateway to your VPC**

The following `attach-vpn-gateway` example attaches the specified virtual private gateway to the specified VPC.

```
aws ec2 attach-vpn-gateway \
  --vpn-gateway-id vgw-9a4cacf3 \
  --vpc-id vpc-a01106c2
```

Output:

```
{
  "VpcAttachment": {
    "State": "attaching",
    "VpcId": "vpc-a01106c2"
  }
}
```

- For API details, see [AttachVpnGateway](#) in *AWS CLI Command Reference*.

authorize-client-vpn-ingress

The following code example shows how to use `authorize-client-vpn-ingress`.

AWS CLI

To add an authorization rule for a Client VPN endpoint

The following `authorize-client-vpn-ingress` example adds an ingress authorization rule that permits all clients to access the internet (`0.0.0.0/0`).

```
aws ec2 authorize-client-vpn-ingress \
  --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde \
  --target-network-cidr 0.0.0.0/0 \
  --authorize-all-groups
```

Output:

```
{
  "Status": {
    "Code": "authorizing"
  }
}
```

For more information, see [Authorization Rules](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [AuthorizeClientVpnIngress](#) in *AWS CLI Command Reference*.

authorize-security-group-egress

The following code example shows how to use `authorize-security-group-egress`.

AWS CLI

To add a rule that allows outbound traffic to a specific address range

This example command adds a rule that grants access to the specified address ranges on TCP port 80.

Command (Linux):

```
aws ec2 authorize-security-group-egress --group-id sg-1a2b3c4d --ip-permissions
IpProtocol=tcp,FromPort=80,ToPort=80,IpRanges='[{"CidrIp=10.0.0.0/16}]'
```

Command (Windows):

```
aws ec2 authorize-security-group-egress --group-id sg-1a2b3c4d --ip-permissions
IpProtocol=tcp,FromPort=80,ToPort=80,IpRanges=[{"CidrIp=10.0.0.0/16}]
```

To add a rule that allows outbound traffic to a specific security group

This example command adds a rule that grants access to the specified security group on TCP port 80.

Command (Linux):

```
aws ec2 authorize-security-group-egress --group-id sg-1a2b3c4d --ip-permissions
IpProtocol=tcp,FromPort=80,ToPort=80,UserIdGroupPairs='[{"GroupId=sg-4b51a32f}]'
```

Command (Windows):

```
aws ec2 authorize-security-group-egress --group-id sg-1a2b3c4d --ip-permissions
IpProtocol=tcp,FromPort=80,ToPort=80,UserIdGroupPairs=[{"GroupId=sg-4b51a32f}]
```

- For API details, see [AuthorizeSecurityGroupEgress](#) in *AWS CLI Command Reference*.

authorize-security-group-ingress

The following code example shows how to use `authorize-security-group-ingress`.

AWS CLI

Example 1: To add a rule that allows inbound SSH traffic

The following `authorize-security-group-ingress` example adds a rule that allows inbound traffic on TCP port 22 (SSH).

```
aws ec2 authorize-security-group-ingress \  
  --group-id sg-1234567890abcdef0 \  
  --protocol tcp \  
  --port 22 \  
  --source-security-group-id sg-1234567890abcdef0
```

```
--cidr 203.0.113.0/24
```

Output:

```
{
  "Return": true,
  "SecurityGroupRules": [
    {
      "SecurityGroupId": "sgr-01afa97ef3e1bedfc",
      "GroupId": "sg-1234567890abcdef0",
      "GroupOwnerId": "123456789012",
      "IsEgress": false,
      "IpProtocol": "tcp",
      "FromPort": 22,
      "ToPort": 22,
      "CidrIpv4": "203.0.113.0/24"
    }
  ]
}
```

Example 2: To add a rule that allows inbound HTTP traffic from another security group

The following `authorize-security-group-ingress` example adds a rule that allows inbound access on TCP port 80 from the source security group `sg-1a2b3c4d`. The source group must be in the same VPC or in a peer VPC (requires a VPC peering connection). Incoming traffic is allowed based on the private IP addresses of instances that are associated with the source security group (not the public IP address or Elastic IP address).

```
aws ec2 authorize-security-group-ingress \
  --group-id sg-1234567890abcdef0 \
  --protocol tcp \
  --port 80 \
  --source-group sg-1a2b3c4d
```

Output:

```
{
  "Return": true,
  "SecurityGroupRules": [
    {
      "SecurityGroupId": "sgr-01f4be99110f638a7",
```



```

    "GroupId": "sg-1234567890abcdef0",
    "GroupOwnerId": "123456789012",
    "IsEgress": false,
    "IpProtocol": "tcp",
    "FromPort": 80,
    "ToPort": 80,
    "ReferencedGroupInfo": {
      "GroupId": "sg-1a2b3c4d",
      "UserId": "123456789012"
    }
  }
]
}

```

Example 3: To add multiple rules in the same call

The following `authorize-security-group-ingress` example uses the `ip-permissions` parameter to add two inbound rules, one that enables inbound access on TCP port 3389 (RDP) and the other that enables ping/ICMP.

```
aws ec2 authorize-security-group-ingress --group-id sg-1234567890abcdef0 --ip-permissions
IpProtocol=tcp,FromPort=3389,ToPort=3389,IpRanges="[{CidrIp=172.31.0.0/16}]"
IpProtocol=icmp,FromPort=-1,ToPort=-1,IpRanges="[{CidrIp=172.31.0.0/16}]"
```

Output:

```

{
  "Return": true,
  "SecurityGroupRules": [
    {
      "SecurityGroupId": "sgr-00e06e5d3690f29f3",
      "GroupId": "sg-1234567890abcdef0",
      "GroupOwnerId": "123456789012",
      "IsEgress": false,
      "IpProtocol": "tcp",
      "FromPort": 3389,
      "ToPort": 3389,
      "CidrIpv4": "172.31.0.0/16"
    },
    {
      "SecurityGroupId": "sgr-0a133dd4493944b87",
      "GroupId": "sg-1234567890abcdef0",
      "GroupOwnerId": "123456789012",

```

```

        "IsEgress": false,
        "IpProtocol": "tcp",
        "FromPort": -1,
        "ToPort": -1,
        "CidrIpv4": "172.31.0.0/16"
    }
]
}

```

Example 4: To add a rule for ICMP traffic

The following `authorize-security-group-ingress` example uses the `ip-permissions` parameter to add an inbound rule that allows the ICMP message Destination Unreachable: Fragmentation Needed and Don't Fragment was Set (Type 3, Code 4) from anywhere.

```
aws ec2 authorize-security-group-ingress --group-id sg-1234567890abcdef0 --ip-permissions IpProtocol=icmp,FromPort=3,ToPort=4,IpRanges="[[CidrIp=0.0.0.0/0]]"
```

Output:

```

{
  "Return": true,
  "SecurityGroupRules": [
    {
      "SecurityGroupRuleId": "sgr-0de3811019069b787",
      "GroupId": "sg-1234567890abcdef0",
      "GroupOwnerId": "123456789012",
      "IsEgress": false,
      "IpProtocol": "icmp",
      "FromPort": 3,
      "ToPort": 4,
      "CidrIpv4": "0.0.0.0/0"
    }
  ]
}

```

Example 5: To add a rule for IPv6 traffic

The following `authorize-security-group-ingress` example uses the `ip-permissions` parameter to add an inbound rule that allows SSH access (port 22) from the IPv6 range `2001:db8:1234:1a00::/64`.

```
aws ec2 authorize-security-group-ingress --group-id sg-1234567890abcdef0 --ip-permissions IpProtocol=tcp,FromPort=22,ToPort=22,Ipv6Ranges="[{CidrIpv6=2001:db8:1234:1a00::/64}]"
```

Output:

```
{
  "Return": true,
  "SecurityGroupRules": [
    {
      "SecurityGroupRuleId": "sgr-0455bc68b60805563",
      "GroupId": "sg-1234567890abcdef0",
      "GroupOwnerId": "123456789012",
      "IsEgress": false,
      "IpProtocol": "tcp",
      "FromPort": 22,
      "ToPort": 22,
      "CidrIpv6": "2001:db8:1234:1a00::/64"
    }
  ]
}
```

Example 6: To add a rule for ICMPv6 traffic

The following `authorize-security-group-ingress` example uses the `ip-permissions` parameter to add an inbound rule that allows ICMPv6 traffic from anywhere.

```
aws ec2 authorize-security-group-ingress --group-id sg-1234567890abcdef0 --ip-permissions IpProtocol=icmpv6,Ipv6Ranges="[{CidrIpv6>::/0}]"
```

Output:

```
{
  "Return": true,
  "SecurityGroupRules": [
    {
      "SecurityGroupRuleId": "sgr-04b612d9363ab6327",
      "GroupId": "sg-1234567890abcdef0",
      "GroupOwnerId": "123456789012",
      "IsEgress": false,
      "IpProtocol": "icmpv6",
      "FromPort": -1,
      "ToPort": -1,
      "CidrIpv6": "::/0"
    }
  ]
}
```

```

    }
  ]
}

```

Example 7: Add a rule with a description

The following `authorize-security-group-ingress` example uses the `ip-permissions` parameter to add an inbound rule that allows RDP traffic from the specified IPv4 address range. The rule includes a description to help you identify it later.

```
aws ec2 authorize-security-group-ingress --group-id sg-1234567890abcdef0 --ip-permissions
IpProtocol=tcp,FromPort=3389,ToPort=3389,IpRanges="[{CidrIp=203.0.113.0/24,Description='RDP
access from NY office'}]"
```

Output:

```

{
  "Return": true,
  "SecurityGroupRules": [
    {
      "SecurityGroupRuleId": "sgr-0397bbcc01e974db3",
      "GroupId": "sg-1234567890abcdef0",
      "GroupOwnerId": "123456789012",
      "IsEgress": false,
      "IpProtocol": "tcp",
      "FromPort": 3389,
      "ToPort": 3389,
      "CidrIpv4": "203.0.113.0/24",
      "Description": "RDP access from NY office"
    }
  ]
}

```

Example 8: To add an inbound rule that uses a prefix list

The following `authorize-security-group-ingress` example uses the `ip-permissions` parameter to add an inbound rule that allows all traffic for the CIDR ranges in the specified prefix list.

```
aws ec2 authorize-security-group-ingress --group-id sg-04a351bfe432d4e71 --ip-permissions
IpProtocol=all,PrefixListIds="[{PrefixListId=pl-002dc3ec097de1514}]"
```

Output:

```
{
  "Return": true,
  "SecurityGroupRules": [
    {
      "SecurityGroupRuleId": "sgr-09c74b32f677c6c7c",
      "GroupId": "sg-1234567890abcdef0",
      "GroupOwnerId": "123456789012",
      "IsEgress": false,
      "IpProtocol": "-1",
      "FromPort": -1,
      "ToPort": -1,
      "PrefixListId": "pl-0721453c7ac4ec009"
    }
  ]
}
```

For more information, see [Security groups](#) in the *Amazon VPC User Guide*.

- For API details, see [AuthorizeSecurityGroupIngress](#) in *AWS CLI Command Reference*.

bundle-instance

The following code example shows how to use `bundle-instance`.

AWS CLI**To bundle an instance**

This example bundles instance `i-1234567890abcdef0` to a bucket called `bundletasks`. Before you specify values for your access key IDs, review and follow the guidance in [Best Practices for Managing AWS Access Keys](#).

Command:

```
aws ec2 bundle-instance --instance-id i-1234567890abcdef0 --bucket bundletasks --
prefix winami --owner-akid AK12AJEXAMPLE --owner-sak example123example
```

Output:

```
{
```

```
"BundleTask": {
  "UpdateTime": "2015-09-15T13:30:35.000Z",
  "InstanceId": "i-1234567890abcdef0",
  "Storage": {
    "S3": {
      "Prefix": "winami",
      "Bucket": "bundletasks"
    }
  },
  "State": "pending",
  "StartTime": "2015-09-15T13:30:35.000Z",
  "BundleId": "bun-294e041f"
}
}
```

- For API details, see [BundleInstance](#) in *AWS CLI Command Reference*.

cancel-bundle-task

The following code example shows how to use `cancel-bundle-task`.

AWS CLI

To cancel a bundle task

This example cancels bundle task `bun-2a4e041c`.

Command:

```
aws ec2 cancel-bundle-task --bundle-id bun-2a4e041c
```

Output:

```
{
  "BundleTask": {
    "UpdateTime": "2015-09-15T13:27:40.000Z",
    "InstanceId": "i-1234567890abcdef0",
    "Storage": {
      "S3": {
        "Prefix": "winami",
        "Bucket": "bundletasks"
      }
    },
  },
}
```

```
"State": "cancelling",
"StartTime": "2015-09-15T13:24:35.000Z",
"BundleId": "bun-2a4e041c"
}
}
```

- For API details, see [CancelBundleTask](#) in *AWS CLI Command Reference*.

cancel-capacity-reservation-fleets

The following code example shows how to use `cancel-capacity-reservation-fleets`.

AWS CLI

To cancel a Capacity Reservation Fleet

The following `cancel-capacity-reservation-fleets` example cancels the specified Capacity Reservation Fleet and the capacity it reserves. When you cancel a Fleet, its status changes to `cancelled`, and it can no longer create new Capacity Reservations. Additionally, all of the individual Capacity Reservations in the Fleet are cancelled, and the instances that were previously running in the reserved capacity continue to run normally in shared capacity.

```
aws ec2 cancel-capacity-reservation-fleets \
  --capacity-reservation-fleet-ids crf-abcdef01234567890
```

Output:

```
{
  "SuccessfulFleetCancellations": [
    {
      "CurrentFleetState": "cancelling",
      "PreviousFleetState": "active",
      "CapacityReservationFleetId": "crf-abcdef01234567890"
    }
  ],
  "FailedFleetCancellations": []
}
```

For more information about Capacity Reservation Fleets, see [Capacity Reservation Fleets](#) in the *Amazon EC2 User Guide*.

- For API details, see [CancelCapacityReservationFleets](#) in *AWS CLI Command Reference*.

cancel-capacity-reservation

The following code example shows how to use `cancel-capacity-reservation`.

AWS CLI

To cancel a capacity reservation

The following `cancel-capacity-reservation` example cancels the specified capacity reservation.

```
aws ec2 cancel-capacity-reservation \  
  --capacity-reservation-id cr-1234abcd56EXAMPLE
```

Output:

```
{  
  "Return": true  
}
```

For more information, see [Canceling a Capacity Reservation](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [CancelCapacityReservation](#) in *AWS CLI Command Reference*.

cancel-conversion-task

The following code example shows how to use `cancel-conversion-task`.

AWS CLI

To cancel an active conversion of an instance or a volume

This example cancels the upload associated with the task ID `import-i-fh95npoc`. If the command succeeds, no output is returned.

Command:

```
aws ec2 cancel-conversion-task --conversion-task-id import-i-fh95npoc
```

- For API details, see [CancelConversionTask](#) in *AWS CLI Command Reference*.

cancel-export-task

The following code example shows how to use `cancel-export-task`.

AWS CLI

To cancel an active export task

This example cancels an active export task with the task ID `export-i-fgelt0i7`. If the command succeeds, no output is returned.

Command:

```
aws ec2 cancel-export-task --export-task-id export-i-fgelt0i7
```

- For API details, see [CancelExportTask](#) in *AWS CLI Command Reference*.

cancel-image-launch-permission

The following code example shows how to use `cancel-image-launch-permission`.

AWS CLI

To cancel having an AMI shared with your Amazon Web Services account

The following `cancel-image-launch-permission` example removes your account from the specified AMI's launch permissions.

```
aws ec2 cancel-image-launch-permission \  
  --image-id ami-0123456789example \  
  --region us-east-1
```

Output:

```
{  
  "Return": true  
}
```

For more information, see [Cancel having an AMI shared with your Amazon Web Services account](#) in the *Amazon EC2 User Guide*.

- For API details, see [CancelImageLaunchPermission](#) in *AWS CLI Command Reference*.

cancel-import-task

The following code example shows how to use `cancel-import-task`.

AWS CLI

To cancel an import task

The following `cancel-import-task` example cancels the specified import image task.

```
aws ec2 cancel-import-task \  
  --import-task-id import-ami-1234567890abcdef0
```

Output:

```
{  
  "ImportTaskId": "import-ami-1234567890abcdef0",  
  "PreviousState": "active",  
  "State": "deleting"  
}
```

- For API details, see [CancelImportTask](#) in *AWS CLI Command Reference*.

cancel-reserved-instances-listing

The following code example shows how to use `cancel-reserved-instances-listing`.

AWS CLI

To cancel a Reserved Instance listing

The following `cancel-reserved-instances-listing` example cancels the specified Reserved Instance listing.

```
aws ec2 cancel-reserved-instances-listing \  
  --reserved-instances-listing-id 5ec28771-05ff-4b9b-aa31-9e57dexample
```

- For API details, see [CancelReservedInstancesListing](#) in *AWS CLI Command Reference*.

cancel-spot-fleet-requests

The following code example shows how to use `cancel-spot-fleet-requests`.

AWS CLI

Example 1: To cancel a Spot fleet request and terminate the associated instances

The following `cancel-spot-fleet-requests` example cancels a Spot Fleet request and terminates the associated On-Demand Instances and Spot Instances.

```
aws ec2 cancel-spot-fleet-requests \  
  --spot-fleet-request-ids sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \  
  --terminate-instances
```

Output:

```
{  
  "SuccessfulFleetRequests": [  
    {  
      "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",  
      "CurrentSpotFleetRequestState": "cancelled_terminating",  
      "PreviousSpotFleetRequestState": "active"  
    }  
  ],  
  "UnsuccessfulFleetRequests": []  
}
```

For more information, see [Cancel a Spot Fleet request](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

Example 2: To cancel a Spot fleet request without terminating the associated instances

The following `cancel-spot-fleet-requests` example cancels a Spot Fleet request without terminating the associated On-Demand Instances and Spot Instances.

```
aws ec2 cancel-spot-fleet-requests \  
  --spot-fleet-request-ids sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \  
  --no-terminate-instances
```

Output:

```
{  
  "SuccessfulFleetRequests": [  
    {  
      "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
```

```
        "CurrentSpotFleetRequestState": "cancelled_running",
        "PreviousSpotFleetRequestState": "active"
    }
],
"UnsuccessfulFleetRequests": []
}
```

For more information, see [Cancel a Spot Fleet request](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [CancelSpotFleetRequests](#) in *AWS CLI Command Reference*.

cancel-spot-instance-requests

The following code example shows how to use `cancel-spot-instance-requests`.

AWS CLI

To cancel Spot Instance requests

This example command cancels a Spot Instance request.

Command:

```
aws ec2 cancel-spot-instance-requests --spot-instance-request-ids sir-08b93456
```

Output:

```
{
  "CancelledSpotInstanceRequests": [
    {
      "State": "cancelled",
      "SpotInstanceRequestId": "sir-08b93456"
    }
  ]
}
```

- For API details, see [CancelSpotInstanceRequests](#) in *AWS CLI Command Reference*.

confirm-product-instance

The following code example shows how to use `confirm-product-instance`.

AWS CLI

To confirm the product instance

This example determines whether the specified product code is associated with the specified instance.

Command:

```
aws ec2 confirm-product-instance --product-code 774F4FF8 --instance-id
i-1234567890abcdef0
```

Output:

```
{
  "OwnerId": "123456789012"
}
```

- For API details, see [ConfirmProductInstance](#) in *AWS CLI Command Reference*.

copy-fpga-image

The following code example shows how to use `copy-fpga-image`.

AWS CLI

To copy an Amazon FPGA image

This example copies the specified AFI from the `us-east-1` region to the current region (`eu-west-1`).

Command:

```
aws ec2 copy-fpga-image --name copy-afi --source-fpga-image-id afi-0d123e123bfc85abc
--source-region us-east-1 --region eu-west-1
```

Output:

```
{
  "FpgaImageId": "afi-06b12350a123fbabc"
}
```

- For API details, see [CopyFpgaImage](#) in *AWS CLI Command Reference*.

copy-image

The following code example shows how to use `copy-image`.

AWS CLI

Example 1: To copy an AMI to another Region

The following `copy-image` example command copies the specified AMI from the `us-west-2` Region to the `us-east-1` Region and adds a short description.

```
aws ec2 copy-image \  
  --region us-east-1 \  
  --name ami-name \  
  --source-region us-west-2 \  
  --source-image-id ami-066877671789bd71b \  
  --description "This is my copied image."
```

Output:

```
{  
  "ImageId": "ami-0123456789abcdefg"  
}
```

For more information, see [Copy an AMI](#) in the *Amazon EC2 User Guide*.

Example 2: To copy an AMI to another Region and encrypt the backing snapshot

The following `copy-image` command copies the specified AMI from the `us-west-2` Region to the current Region and encrypts the backing snapshot using the specified KMS key.

```
aws ec2 copy-image \  
  --source-region us-west-2 \  
  --name ami-name \  
  --source-image-id ami-066877671789bd71b \  
  --encrypted \  
  --kms-key-id alias/my-kms-key
```

Output:

```
{
  "ImageId": "ami-0123456789abcdefg"
}
```

For more information, see [Copy an AMI](#) in the *Amazon EC2 User Guide*.

Example 3: To include your user-defined AMI tags when copying an AMI

The following `copy-image` command uses the `--copy-image-tags` parameter to copy your user-defined AMI tags when copying the AMI.

```
aws ec2 copy-image \
  --region us-east-1 \
  --name ami-name \
  --source-region us-west-2 \
  --source-image-id ami-066877671789bd71b \
  --description "This is my copied image."
  --copy-image-tags
```

Output:

```
{
  "ImageId": "ami-0123456789abcdefg"
}
```

For more information, see [Copy an AMI](#) in the *Amazon EC2 User Guide*.

- For API details, see [CopyImage](#) in *AWS CLI Command Reference*.

copy-snapshot

The following code example shows how to use `copy-snapshot`.

AWS CLI

Example 1: To copy a snapshot to another Region

The following `copy-snapshot` example command copies the specified snapshot from the `us-west-2` Region to the `us-east-1` Region and adds a short description.

```
aws ec2 copy-snapshot \
```

```
--region us-east-1 \  
--source-region us-west-2 \  
--source-snapshot-id snap-066877671789bd71b \  
--description "This is my copied snapshot."
```

Output:

```
{  
  "SnapshotId": "snap-066877671789bd71b"  
}
```

For more information, see [Copy an Amazon EBS snapshot](#) in the *Amazon EC2 User Guide*.

Example 2: To copy an unencrypted snapshot and encrypt the new snapshot

The following `copy-snapshot` command copies the specified unencrypted snapshot from the `us-west-2` Region to the current Region and encrypts the new snapshot using the specified KMS key.

```
aws ec2 copy-snapshot \  
  --source-region us-west-2 \  
  --source-snapshot-id snap-066877671789bd71b \  
  --encrypted \  
  --kms-key-id alias/my-kms-key
```

Output:

```
{  
  "SnapshotId": "snap-066877671789bd71b"  
}
```

For more information, see [Copy an Amazon EBS snapshot](#) in the *Amazon EC2 User Guide*.

- For API details, see [CopySnapshot](#) in *AWS CLI Command Reference*.

create-capacity-reservation-fleet

The following code example shows how to use `create-capacity-reservation-fleet`.

AWS CLI

To create a Capacity Reservation Fleet

The following `create-capacity-reservation-fleet` example creates a Capacity Reservation Fleet for the instance type specified in the request, up to the specified total target capacity. The number of instances for which the Capacity Reservation Fleet reserves capacity depends on the total target capacity and instance type weights that you specify in the request. Specify the instance types to use and a priority for each of the designated instance types.

```
aws ec2 create-capacity-reservation-fleet \  
--total-target-capacity 24 \  
--allocation-strategy prioritized \  
--instance-match-criteria open \  
--tenancy default \  
--end-date 2022-12-31T23:59:59.000Z \  
--instance-type-specifications file://instanceTypeSpecification.json
```

Contents of `instanceTypeSpecification.json`:

```
[  
  {  
    "InstanceType": "m5.xlarge",  
    "InstancePlatform": "Linux/UNIX",  
    "Weight": 3.0,  
    "AvailabilityZone": "us-east-1a",  
    "EbsOptimized": true,  
    "Priority" : 1  
  }  
]
```

Output:

```
{  
  "Status": "submitted",  
  "TotalFulfilledCapacity": 0.0,  
  "CapacityReservationFleetId": "crf-abcdef01234567890",  
  "TotalTargetCapacity": 24  
}
```

For more information about Capacity Reservation Fleets, see [Capacity Reservation Fleets](#) in the *Amazon EC2 User Guide*.

For more information about instance type weight and total target capacity, see [Instance type weight](#) and [Total target capacity](#) in the *Amazon EC2 User Guide*.

For more information about designating priority for specified instance types, see [Allocation strategy](#) and [Instance type priority](#) in the *Amazon EC2 User Guide*.

- For API details, see [CreateCapacityReservationFleet](#) in *AWS CLI Command Reference*.

create-capacity-reservation

The following code example shows how to use create-capacity-reservation.

AWS CLI

Example 1: To create a Capacity Reservation

The following create-capacity-reservation example creates a capacity reservation in the eu-west-1a Availability Zone, into which you can launch three t2.medium instances running a Linux/Unix operating system. By default, the capacity reservation is created with open instance matching criteria and no support for ephemeral storage, and it remains active until you manually cancel it.

```
aws ec2 create-capacity-reservation \  
  --availability-zone eu-west-1a \  
  --instance-type t2.medium \  
  --instance-platform Linux/UNIX \  
  --instance-count 3
```

Output:

```
{  
  "CapacityReservation": {  
    "CapacityReservationId": "cr-1234abcd56EXAMPLE ",  
    "EndDateType": "unlimited",  
    "AvailabilityZone": "eu-west-1a",  
    "InstanceMatchCriteria": "open",  
    "EphemeralStorage": false,  
    "CreateDate": "2019-08-16T09:27:35.000Z",  
    "AvailableInstanceCount": 3,  
    "InstancePlatform": "Linux/UNIX",  
    "TotalInstanceCount": 3,  
    "State": "active",  
    "Tenancy": "default",  
    "EbsOptimized": false,  
    "InstanceType": "t2.medium"
```

```
}  
}
```

Example 2: To create a Capacity Reservation that automatically ends at a specified date/time

The following `create-capacity-reservation` example creates a capacity reservation in the `eu-west-1a` Availability Zone, into which you can launch three `m5.large` instances running a Linux/Unix operating system. This capacity reservation automatically ends on 08/31/2019 at 23:59:59.

```
aws ec2 create-capacity-reservation \  
  --availability-zone eu-west-1a \  
  --instance-type m5.large \  
  --instance-platform Linux/UNIX \  
  --instance-count 3 \  
  --end-date-type limited \  
  --end-date 2019-08-31T23:59:59Z
```

Output:

```
{  
  "CapacityReservation": {  
    "CapacityReservationId": "cr-1234abcd56EXAMPLE ",  
    "EndDateType": "limited",  
    "AvailabilityZone": "eu-west-1a",  
    "EndDate": "2019-08-31T23:59:59.000Z",  
    "InstanceMatchCriteria": "open",  
    "EphemeralStorage": false,  
    "CreateDate": "2019-08-16T10:15:53.000Z",  
    "AvailableInstanceCount": 3,  
    "InstancePlatform": "Linux/UNIX",  
    "TotalInstanceCount": 3,  
    "State": "active",  
    "Tenancy": "default",  
    "EbsOptimized": false,  
    "InstanceType": "m5.large"  
  }  
}
```

Example 3: To create a Capacity Reservation that accepts only targeted instance launches

The following `create-capacity-reservation` example creates a capacity reservation that accepts only targeted instance launches.

```
aws ec2 create-capacity-reservation \  
  --availability-zone eu-west-1a \  
  --instance-type m5.large \  
  --instance-platform Linux/UNIX \  
  --instance-count 3 \  
  --instance-match-criteria targeted
```

Output:

```
{  
  "CapacityReservation": {  
    "CapacityReservationId": "cr-1234abcd56EXAMPLE ",  
    "EndDateType": "unlimited",  
    "AvailabilityZone": "eu-west-1a",  
    "InstanceMatchCriteria": "targeted",  
    "EphemeralStorage": false,  
    "CreateDate": "2019-08-16T10:21:57.000Z",  
    "AvailableInstanceCount": 3,  
    "InstancePlatform": "Linux/UNIX",  
    "TotalInstanceCount": 3,  
    "State": "active",  
    "Tenancy": "default",  
    "EbsOptimized": false,  
    "InstanceType": "m5.large"  
  }  
}
```

For more information, see [Creating a Capacity Reservation](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [CreateCapacityReservation](#) in *AWS CLI Command Reference*.

create-carrier-gateway

The following code example shows how to use `create-carrier-gateway`.

AWS CLI

To create a carrier gateway

The following `create-carrier-gateway` example creates a carrier gateway for the specified VPC.

```
aws ec2 create-carrier-gateway \  
  --vpc-id vpc-0c529aEXAMPLE1111
```

Output:

```
{  
  "CarrierGateway": {  
    "CarrierGatewayId": "cagw-0465cdEXAMPLE1111",  
    "VpcId": "vpc-0c529aEXAMPLE1111",  
    "State": "pending",  
    "OwnerId": "123456789012"  
  }  
}
```

For more information, see [Carrier gateways](#) in the *AWS Wavelength User Guide*.

- For API details, see [CreateCarrierGateway](#) in *AWS CLI Command Reference*.

create-client-vpn-endpoint

The following code example shows how to use `create-client-vpn-endpoint`.

AWS CLI

To create a Client VPN endpoint

The following `create-client-vpn-endpoint` example creates a Client VPN endpoint that uses mutual authentication and specifies a value for the client CIDR block.

```
aws ec2 create-client-vpn-endpoint \  
  --client-cidr-block "172.31.0.0/16" \  
  --server-certificate-arn arn:aws:acm:ap-south-1:123456789012:certificate/  
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \  
  --authentication-options Type=certificate-  
authentication,MutualAuthentication={ClientRootCertificateChainArn=arn:aws:acm:ap-  
south-1:123456789012:certificate/a1b2c3d4-5678-90ab-cdef-22222EXAMPLE} \  
  --connection-log-options Enabled=false
```

Output:

```
{
  "ClientVpnEndpointId": "cvpn-endpoint-123456789123abcde",
  "Status": {
    "Code": "pending-associate"
  },
  "DnsName": "cvpn-endpoint-123456789123abcde.prod.clientvpn.ap-
south-1.amazonaws.com"
}
```

For more information, see [Client VPN Endpoints](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [CreateClientVpnEndpoint](#) in *AWS CLI Command Reference*.

create-client-vpn-route

The following code example shows how to use `create-client-vpn-route`.

AWS CLI**To create a route for a Client VPN endpoint**

The following `create-client-vpn-route` example adds a route to the internet (`0.0.0.0/0`) for the specified subnet of the Client VPN endpoint.

```
aws ec2 create-client-vpn-route \
  --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde \
  --destination-cidr-block 0.0.0.0/0 \
  --target-vpc-subnet-id subnet-0123456789abcabca
```

Output:

```
{
  "Status": {
    "Code": "creating"
  }
}
```

For more information, see [Routes](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [CreateClientVpnRoute](#) in *AWS CLI Command Reference*.

create-coip-cidr

The following code example shows how to use `create-coip-cidr`.

AWS CLI

To create a range of customer-owned IP (CoIP) addresses

The following `create-coip-cidr` example creates the specified range of CoIP addresses in the specified CoIP pool.

```
aws ec2 create-coip-cidr \  
  --cidr 15.0.0.0/24 \  
  --coip-pool-id ipv4pool-coip-1234567890abcdefg
```

Output:

```
{  
  "CoipCidr": {  
    "Cidr": "15.0.0.0/24",  
    "CoipPoolId": "ipv4pool-coip-1234567890abcdefg",  
    "LocalGatewayRouteTableId": "lgw-rtb-abcdefg1234567890"  
  }  
}
```

For more information, see [Customer-owned IP addresses](#) in the *AWS Outposts User Guide*.

- For API details, see [CreateCoipCidr](#) in *AWS CLI Command Reference*.

create-coip-pool

The following code example shows how to use `create-coip-pool`.

AWS CLI

To create a pool of customer-owned IP (CoIP) addresses

The following `create-coip-pool` example creates a CoIP pool for CoIP addresses in the specified local gateway route table.

```
aws ec2 create-coip-pool \  
  --local-gateway-route-table-id lgw-rtb-abcdefg1234567890
```

Output:

```
{
  "CoipPool": {
    "PoolId": "ipv4pool-coip-1234567890abcdefg",
    "LocalGatewayRouteTableId": "lgw-rtb-abcdefg1234567890",
    "PoolArn": "arn:aws:ec2:us-west-2:123456789012:coip-pool/ipv4pool-
coip-1234567890abcdefg"
  }
}
```

For more information, see [Customer-owned IP addresses](#) in the *AWS Outposts User Guide*.

- For API details, see [CreateCoipPool](#) in *AWS CLI Command Reference*.

create-customer-gateway

The following code example shows how to use `create-customer-gateway`.

AWS CLI**To create a customer gateway**

This example creates a customer gateway with the specified IP address for its outside interface.

Command:

```
aws ec2 create-customer-gateway --type ipsec.1 --public-ip 12.1.2.3 --bgp-asn 65534
```

Output:

```
{
  "CustomerGateway": {
    "CustomerGatewayId": "cgw-0e11f167",
    "IpAddress": "12.1.2.3",
    "State": "available",
    "Type": "ipsec.1",
    "BgpAsn": "65534"
  }
}
```

- For API details, see [CreateCustomerGateway](#) in *AWS CLI Command Reference*.

create-default-subnet

The following code example shows how to use `create-default-subnet`.

AWS CLI

To create a default subnet

This example creates a default subnet in Availability Zone `us-east-2a`.

Command:

```
aws ec2 create-default-subnet --availability-zone us-east-2a
```

```
{
  "Subnet": {
    "AvailabilityZone": "us-east-2a",
    "Tags": [],
    "AvailableIpAddressCount": 4091,
    "DefaultForAz": true,
    "Ipv6CidrBlockAssociationSet": [],
    "VpcId": "vpc-1a2b3c4d",
    "State": "available",
    "MapPublicIpOnLaunch": true,
    "SubnetId": "subnet-1122aabb",
    "CidrBlock": "172.31.32.0/20",
    "AssignIpv6AddressOnCreation": false
  }
}
```

- For API details, see [CreateDefaultSubnet](#) in *AWS CLI Command Reference*.

create-default-vpc

The following code example shows how to use `create-default-vpc`.

AWS CLI

To create a default VPC

This example creates a default VPC.

Command:

```
aws ec2 create-default-vpc
```

Output:

```
{
  "Vpc": {
    "VpcId": "vpc-8eaae5ea",
    "InstanceTenancy": "default",
    "Tags": [],
    "Ipv6CidrBlockAssociationSet": [],
    "State": "pending",
    "DhcpOptionsId": "dopt-af0c32c6",
    "CidrBlock": "172.31.0.0/16",
    "IsDefault": true
  }
}
```

- For API details, see [CreateDefaultVpc](#) in *AWS CLI Command Reference*.

create-dhcp-options

The following code example shows how to use create-dhcp-options.

AWS CLI

To create a set of DHCP options

The following create-dhcp-options example creates a set of DHCP options that specifies the domain name, the domain name servers, and the NetBIOS node type.

```
aws ec2 create-dhcp-options \
  --dhcp-configuration \
    "Key=domain-name-servers,Values=10.2.5.1,10.2.5.2" \
    "Key=domain-name,Values=example.com" \
    "Key=netbios-node-type,Values=2"
```

Output:

```
{
  "DhcpOptions": {
    "DhcpConfigurations": [
```

```
{
  "Key": "domain-name",
  "Values": [
    {
      "Value": "example.com"
    }
  ]
},
{
  "Key": "domain-name-servers",
  "Values": [
    {
      "Value": "10.2.5.1"
    },
    {
      "Value": "10.2.5.2"
    }
  ]
},
{
  "Key": "netbios-node-type",
  "Values": [
    {
      "Value": "2"
    }
  ]
}
],
"DhcpOptionsId": "dopt-06d52773eff4c55f3"
}
```

- For API details, see [CreateDhcpOptions](#) in *AWS CLI Command Reference*.

create-egress-only-internet-gateway

The following code example shows how to use `create-egress-only-internet-gateway`.

AWS CLI

To create an egress-only Internet gateway

This example creates an egress-only Internet gateway for the specified VPC.

Command:

```
aws ec2 create-egress-only-internet-gateway --vpc-id vpc-0c62a468
```

Output:

```
{
  "EgressOnlyInternetGateway": {
    "EgressOnlyInternetGatewayId": "eigw-015e0e244e24dfe8a",
    "Attachments": [
      {
        "State": "attached",
        "VpcId": "vpc-0c62a468"
      }
    ]
  }
}
```

- For API details, see [CreateEgressOnlyInternetGateway](#) in *AWS CLI Command Reference*.

create-fleet

The following code example shows how to use `create-fleet`.

AWS CLI**Example 1: To create an EC2 Fleet that launches Spot Instances as the default purchasing model**

The following `create-fleet` example creates an EC2 Fleet using the minimum parameters required to launch a fleet: a launch template, target capacity, and default purchasing model. The launch template is identified by its launch template ID and version number. The target capacity for the fleet is 2 instances, and the default purchasing model is `spot`, which results in the fleet launching 2 Spot Instances.

When you create an EC2 Fleet, use a JSON file to specify information about the instances to launch.

```
aws ec2 create-fleet \  
  --cli-input-json file://file_name.json
```

Contents of file_name.json:

```
{
  "LaunchTemplateConfigs": [
    {
      "LaunchTemplateSpecification": {
        "LaunchTemplateId": "lt-0e8c754449b27161c",
        "Version": "1"
      }
    }
  ],
  "TargetCapacitySpecification": {
    "TotalTargetCapacity": 2,
    "DefaultTargetCapacityType": "spot"
  }
}
```

Output:

```
{
  "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE"
}
```

Example 2: To create an EC2 Fleet that launches On-Demand Instances as the default purchasing model

The following `create-fleet` example creates an EC2 Fleet using the minimum parameters required to launch a fleet: a launch template, target capacity, and default purchasing model. The launch template is identified by its launch template ID and version number. The target capacity for the fleet is 2 instances, and the default purchasing model is on-demand, which results in the fleet launching 2 On-Demand Instances.

When you create an EC2 Fleet, use a JSON file to specify information about the instances to launch.

```
aws ec2 create-fleet \
  --cli-input-json file://file_name.json
```

Contents of file_name.json:

```
{
```

```
"LaunchTemplateConfigs": [  
  {  
    "LaunchTemplateSpecification": {  
      "LaunchTemplateId": "lt-0e8c754449b27161c",  
      "Version": "1"  
    }  
  }  
],  
"TargetCapacitySpecification": {  
  "TotalTargetCapacity": 2,  
  "DefaultTargetCapacityType": "on-demand"  
}  
}
```

Output:

```
{  
  "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE"  
}
```

Example 3: To create an EC2 Fleet that launches On-Demand Instances as the primary capacity

The following `create-fleet` example creates an EC2 Fleet that specifies the total target capacity of 2 instances for the fleet, and a target capacity of 1 On-Demand Instance. The default purchasing model is spot. The fleet launches 1 On-Demand Instance as specified, but needs to launch one more instance to fulfil the total target capacity. The purchasing model for the difference is calculated as `TotalTargetCapacity - OnDemandTargetCapacity = DefaultTargetCapacityType`, which results in the fleet launching 1 Spot Instance.

When you create an EC2 Fleet, use a JSON file to specify information about the instances to launch.

```
aws ec2 create-fleet \  
  --cli-input-json file:///file_name.json
```

Contents of file_name.json:

```
{  
  "LaunchTemplateConfigs": [  
    {
```

```
    "LaunchTemplateSpecification": {
      "LaunchTemplateId": "lt-0e8c754449b27161c",
      "Version": "1"
    }
  ],
  "TargetCapacitySpecification": {
    "TotalTargetCapacity": 2,
    "OnDemandTargetCapacity": 1,
    "DefaultTargetCapacityType": "spot"
  }
}
```

Output:

```
{
  "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE"
}
```

Example 4: To create an EC2 Fleet that launches Spot Instances using the lowest-price allocation strategy

If the allocation strategy for Spot Instances is not specified, the default allocation strategy, which is lowest-price, is used. The following `create-fleet` example creates an EC2 Fleet using the lowest-price allocation strategy. The three launch specifications, which override the launch template, have different instance types but the same weighted capacity and subnet. The total target capacity is 2 instances and the default purchasing model is spot. The EC2 Fleet launches 2 Spot Instances using the instance type of the launch specification with the lowest price.

When you create an EC2 Fleet, use a JSON file to specify information about the instances to launch.

```
aws ec2 create-fleet \
  --cli-input-json file:///file_name.jsonContents of file_name.json::

{
  "LaunchTemplateConfigs": [
    {
      "LaunchTemplateSpecification": {
        "LaunchTemplateId": "lt-0e8c754449b27161c",
```

```
    "Version": "1"
  },
  "Overrides": [
    {
      "InstanceType": "c4.large",
      "WeightedCapacity": 1,
      "SubnetId": "subnet-a4f6c5d3"
    },
    {
      "InstanceType": "c3.large",
      "WeightedCapacity": 1,
      "SubnetId": "subnet-a4f6c5d3"
    },
    {
      "InstanceType": "c5.large",
      "WeightedCapacity": 1,
      "SubnetId": "subnet-a4f6c5d3"
    }
  ]
}
],
"TargetCapacitySpecification": {
  "TotalTargetCapacity": 2,
  "DefaultTargetCapacityType": "spot"
}
}
```

Output:

```
{
  "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE"
}
```

- For API details, see [CreateFleet](#) in *AWS CLI Command Reference*.

create-flow-logs

The following code example shows how to use create-flow-logs.

AWS CLI

Example 1: To create a flow log

The following `create-flow-logs` example creates a flow log that captures all rejected traffic for the specified network interface. The flow logs are delivered to a log group in CloudWatch Logs using the permissions in the specified IAM role.

```
aws ec2 create-flow-logs \  
  --resource-type NetworkInterface \  
  --resource-ids eni-11223344556677889 \  
  --traffic-type REJECT \  
  --log-group-name my-flow-logs \  
  --deliver-logs-permission-arn arn:aws:iam::123456789101:role/publishFlowLogs
```

Output:

```
{  
  "ClientToken": "so0eNA2uSHUN1HI0S2cJ305GuIX1CezaRdGtexample",  
  "FlowLogIds": [  
    "fl-12345678901234567"  
  ],  
  "Unsuccessful": []  
}
```

For more information, see [VPC Flow Logs](#) in the *Amazon VPC User Guide*.

Example 2: To create a flow log with a custom format

The following `create-flow-logs` example creates a flow log that captures all traffic for the specified VPC and delivers the flow logs to an Amazon S3 bucket. The `--log-format` parameter specifies a custom format for the flow log records. To run this command on Windows, change the single quotes (') to double quotes (").

```
aws ec2 create-flow-logs \  
  --resource-type VPC \  
  --resource-ids vpc-00112233344556677 \  
  --traffic-type ALL \  
  --log-destination-type s3 \  
  --log-destination arn:aws:s3:::flow-log-bucket/my-custom-flow-logs/ \  
  --log-format '${version} ${vpc-id} ${subnet-id} ${instance-id} ${srcaddr}  
  ${dstaddr} ${srcport} ${dstport} ${protocol} ${tcp-flags} ${type} ${pkt-srcaddr}  
  ${pkt-dstaddr}'
```

For more information, see [VPC Flow Logs](#) in the *Amazon VPC User Guide*.

Example 3: To create a flow log with a one-minute maximum aggregation interval

The following `create-flow-logs` example creates a flow log that captures all traffic for the specified VPC and delivers the flow logs to an Amazon S3 bucket. The `--max-aggregation-interval` parameter specifies a maximum aggregation interval of 60 seconds (1 minute).

```
aws ec2 create-flow-logs \  
  --resource-type VPC \  
  --resource-ids vpc-00112233344556677 \  
  --traffic-type ALL \  
  --log-destination-type s3 \  
  --log-destination arn:aws:s3:::flow-log-bucket/my-custom-flow-logs/ \  
  --max-aggregation-interval 60
```

For more information, see [VPC Flow Logs](#) in the *Amazon VPC User Guide*.

- For API details, see [CreateFlowLogs](#) in *AWS CLI Command Reference*.

create-fpga-image

The following code example shows how to use `create-fpga-image`.

AWS CLI

To create an Amazon FPGA image

This example creates an AFI from the specified tarball in the specified bucket.

Command:

```
aws ec2 create-fpga-image --name my-afi --description test-afi --input-storage-  
location Bucket=my-fpga-bucket,Key=dcp/17_12_22-103226.Developer_CL.tar --logs-  
storage-location Bucket=my-fpga-bucket,Key=logs
```

Output:

```
{  
  "FpgaImageId": "afi-0d123e123bfc85abc",  
  "FpgaImageGlobalId": "agfi-123cb27b5e84a0abc"  
}
```

- For API details, see [CreateFpgaImage](#) in *AWS CLI Command Reference*.

create-image

The following code example shows how to use `create-image`.

AWS CLI

Example 1: To create an AMI from an Amazon EBS-backed instance

The following `create-image` example creates an AMI from the specified instance.

```
aws ec2 create-image \  
  --instance-id i-1234567890abcdef0 \  
  --name "My server" \  
  --description "An AMI for my server"
```

Output:

```
{  
  "ImageId": "ami-abcdef01234567890"  
}
```

For more information about specifying a block device mapping for your AMI, see [Specifying a block device mapping for an AMI](#) in the *Amazon EC2 User Guide*.

Example 2: To create an AMI from an Amazon EBS-backed instance without reboot

The following `create-image` example creates an AMI and sets the `--no-reboot` parameter, so that the instance is not rebooted before the image is created.

```
aws ec2 create-image \  
  --instance-id i-1234567890abcdef0 \  
  --name "My server" \  
  --no-reboot
```

Output:

```
{  
  "ImageId": "ami-abcdef01234567890"  
}
```

For more information about specifying a block device mapping for your AMI, see [Specifying a block device mapping for an AMI](#) in the *Amazon EC2 User Guide*.

Example 3: To tag an AMI and snapshots on creation

The following `create-image` example creates an AMI, and tags the AMI and the snapshots with the same tag `cost-center=cc123`

```
aws ec2 create-image \  
  --instance-id i-1234567890abcdef0 \  
  --name "My server" \  
  --tag-specifications "ResourceType=image,Tags=[{Key=cost-center,Value=cc123}]" \  
  "ResourceType=snapshot,Tags=[{Key=cost-center,Value=cc123}]"
```

Output:

```
{  
  "ImageId": "ami-abcdef01234567890"  
}
```

For more information about tagging your resources on creation, see [Add tags on resource creation](#) in the *Amazon EC2 User Guide*.

- For API details, see [CreateImage](#) in *AWS CLI Command Reference*.

create-instance-connect-endpoint

The following code example shows how to use `create-instance-connect-endpoint`.

AWS CLI

To create an EC2 Instance Connect Endpoint

The following `create-instance-connect-endpoint` example creates an EC2 Instance Connect Endpoint in the specified subnet.

```
aws ec2 create-instance-connect-endpoint \  
  --region us-east-1 \  
  --subnet-id subnet-0123456789example
```

Output:

```
{
  "VpcId": "vpc-0123abcd",
  "InstanceConnectEndpointArn": "arn:aws:ec2:us-east-1:111111111111:instance-
connect-endpoint/eice-0123456789example",
  "AvailabilityZone": "us-east-1a",
  "NetworkInterfaceIds": [
    "eni-0123abcd"
  ],
  "PreserveClientIp": true,
  "Tags": [],
  "FipsDnsName": "eice-0123456789example.0123abcd.fips.ec2-instance-connect-
endpoint.us-east-1.amazonaws.com",
  "StateMessage": "",
  "State": "create-complete",
  "DnsName": "eice-0123456789example.0123abcd.ec2-instance-connect-endpoint.us-
east-1.amazonaws.com",
  "SubnetId": "subnet-0123abcd",
  "OwnerId": "111111111111",
  "SecurityGroupIds": [
    "sg-0123abcd"
  ],
  "InstanceConnectEndpointId": "eice-0123456789example",
  "CreatedAt": "2023-04-07T15:43:53.000Z"
}
```

For more information, see [Create an EC2 Instance Connect Endpoint](#) in the *Amazon EC2 User Guide*.

- For API details, see [CreateInstanceConnectEndpoint](#) in *AWS CLI Command Reference*.

create-instance-event-window

The following code example shows how to use create-instance-event-window.

AWS CLI

Example 1: To create an event window with a time range

The following create-instance-event-window example creates an event window with a time range. You can't also specify the cron-expression parameter.

```
aws ec2 create-instance-event-window \
```

```

--region us-east-1 \
--time-range StartWeekDay=monday,StartHour=2,EndWeekDay=wednesday,EndHour=8 \
--tag-specifications "ResourceType=instance-event-
window,Tags=[{Key=K1,Value=V1}]" \
--name myEventWindowName

```

Output:

```

{
  "InstanceEventWindow": {
    "InstanceEventWindowId": "iew-0abcdef1234567890",
    "TimeRanges": [
      {
        "StartWeekDay": "monday",
        "StartHour": 2,
        "EndWeekDay": "wednesday",
        "EndHour": 8
      }
    ],
    "Name": "myEventWindowName",
    "State": "creating",
    "Tags": [
      {
        "Key": "K1",
        "Value": "V1"
      }
    ]
  }
}

```

For event window constraints, see [Considerations](#) in the Scheduled Events section of the *Amazon EC2 User Guide*.

Example 2: To create an event window with a cron expression

The following `create-instance-event-window` example creates an event window with a cron expression. You can't also specify the `time-range` parameter.

```

aws ec2 create-instance-event-window \
--region us-east-1 \
--cron-expression "*" 21-23 * * 2,3" \
--tag-specifications "ResourceType=instance-event-
window,Tags=[{Key=K1,Value=V1}]" \

```

```
--name myEventWindowName
```

Output:

```
{
  "InstanceEventWindow": {
    "InstanceEventWindowId": "iew-0abcdef1234567890",
    "Name": "myEventWindowName",
    "CronExpression": "* 21-23 * * 2,3",
    "State": "creating",
    "Tags": [
      {
        "Key": "K1",
        "Value": "V1"
      }
    ]
  }
}
```

For event window constraints, see [Considerations](#) in the Scheduled Events section of the *Amazon EC2 User Guide*.

- For API details, see [CreateInstanceEventWindow](#) in *AWS CLI Command Reference*.

create-instance-export-task

The following code example shows how to use `create-instance-export-task`.

AWS CLI

To export an instance

This example command creates a task to export the instance `i-1234567890abcdef0` to the Amazon S3 bucket `myexportbucket`.

Command:

```
aws ec2 create-instance-export-task --description "RHEL5 instance" --instance-id i-1234567890abcdef0 --target-environment vmware --export-to-s3-task DiskImageFormat=vmdk,ContainerFormat=ova,S3Bucket=myexportbucket,S3Prefix=RHEL5
```

Output:

```
{
  "ExportTask": {
    "State": "active",
    "InstanceExportDetails": {
      "InstanceId": "i-1234567890abcdef0",
      "TargetEnvironment": "vmware"
    },
    "ExportToS3Task": {
      "S3Bucket": "myexportbucket",
      "S3Key": "RHEL5export-i-fh8sjjsq.ova",
      "DiskImageFormat": "vmdk",
      "ContainerFormat": "ova"
    },
    "Description": "RHEL5 instance",
    "ExportTaskId": "export-i-fh8sjjsq"
  }
}
```

- For API details, see [CreateInstanceExportTask](#) in *AWS CLI Command Reference*.

create-internet-gateway

The following code example shows how to use `create-internet-gateway`.

AWS CLI

To create an internet gateway

The following `create-internet-gateway` example creates an internet gateway with the tag `Name=my-igw`.

```
aws ec2 create-internet-gateway \
  --tag-specifications ResourceType=internet-gateway,Tags=[{Key=Name,Value=my-igw}]
```

Output:

```
{
  "InternetGateway": {
    "Attachments": [],
    "InternetGatewayId": "igw-0d0fb496b3994d755",
```



```
    "OwnerId": "123456789012",
    "Tags": [
      {
        "Key": "Name",
        "Value": "my-igw"
      }
    ]
  }
}
```

For more information, see [Internet gateways](#) in the *Amazon VPC User Guide*.

- For API details, see [CreateInternetGateway](#) in *AWS CLI Command Reference*.

create-ipam-pool

The following code example shows how to use `create-ipam-pool`.

AWS CLI

To create an IPAM pool

The following `create-ipam-pool` example creates an IPAM pool.

(Linux):

```
aws ec2 create-ipam-pool \
  --ipam-scope-id ipam-scope-02fc38cd4c48e7d38 \
  --address-family ipv4 \
  --auto-import \
  --allocation-min-netmask-length 16 \
  --allocation-max-netmask-length 26 \
  --allocation-default-netmask-length 24 \
  --allocation-resource-tags "Key=Environment,Value=Preprod" \
  --tag-specifications 'ResourceType=ipam-pool,Tags=[{Key=Name,Value="Preprod
pool"}]'
```

(Windows):

```
aws ec2 create-ipam-pool ^
  --ipam-scope-id ipam-scope-02fc38cd4c48e7d38 ^
  --address-family ipv4 ^
```

```
--auto-import ^
--allocation-min-netmask-length 16 ^
--allocation-max-netmask-length 26 ^
--allocation-default-netmask-length 24 ^
--allocation-resource-tags "Key=Environment,Value=Preprod" ^
--tag-specifications ResourceType=ipam-pool,Tags=[{Key=Name,Value="Preprod
pool"}]
```

Output:

```
{
  "IpamPool": {
    "OwnerId": "123456789012",
    "IpamPoolId": "ipam-pool-0533048da7d823723",
    "IpamPoolArn": "arn:aws:ec2::123456789012:ipam-pool/ipam-
pool-0533048da7d823723",
    "IpamScopeArn": "arn:aws:ec2::123456789012:ipam-scope/ipam-
scope-02fc38cd4c48e7d38",
    "IpamScopeType": "private",
    "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",
    "IpamRegion": "us-east-1",
    "Locale": "None",
    "PoolDepth": 1,
    "State": "create-in-progress",
    "AutoImport": true,
    "AddressFamily": "ipv4",
    "AllocationMinNetmaskLength": 16,
    "AllocationMaxNetmaskLength": 26,
    "AllocationDefaultNetmaskLength": 24,
    "AllocationResourceTags": [
      {
        "Key": "Environment",
        "Value": "Preprod"
      }
    ],
    "Tags": [
      {
        "Key": "Name",
        "Value": "Preprod pool"
      }
    ]
  }
}
```

For more information, see [Plan for IP address provisioning](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [CreateIpamPool](#) in *AWS CLI Command Reference*.

create-ipam-resource-discovery

The following code example shows how to use `create-ipam-resource-discovery`.

AWS CLI

To create a resource discovery

In this example, you're a delegated IPAM admin who wants to create and share a resource discovery with the IPAM admin in another AWS Organization so that the admin in the other organization can manage and monitor the IP addresses of resources in your organization.

Important

This example includes both the `--region` and `--operating-regions` options because, while they are optional, they must be configured in a particular way to successfully integrate a resource discovery with an IPAM. * `--operating-regions` must match the Regions where you have resources that you want IPAM to discover. If there are Regions where you do not want IPAM to manage the IP addresses (for example for compliance reasons), do not include them. * `--region` must match the home Region of the IPAM you want to associate it with. You must create the resource discovery in the same Region that the IPAM was created in. For example, if the IPAM you are associating with was created in `us-east-1`, include `--region us-east-1` in the request. Both the `--region` and `--operating-regions` options default to the Region you're running the command in if you don't specify them.

In this example, the operating Regions of the IPAM we're integrating with include `us-west-1`, `us-west-2`, and `ap-south-1`. When we create the resource discovery, we want IPAM to discover the resource IP addresses in `us-west-1` and `us-west-2` but not `ap-south-1`. So we are including only `--operating-regions RegionName='us-west-1' RegionName='us-west-2'` in the request.

The following `create-ipam-resource-discovery` example creates an IPAM resource discovery.

```
aws ec2 create-ipam-resource-discovery \  
  --description 'Example-resource-discovery' \  
  --region us-east-1 \  
  --operating-regions RegionName='us-west-1' RegionName='us-west-2'
```

```
--tag-specifications 'ResourceType=ipam-resource-discovery,Tags=[{Key=cost-
center,Value=cc123}]' \
--operating-regions RegionName='us-west-1' RegionName='us-west-2' \
--region us-east-1
```

Output:

```
{
  "IpamResourceDiscovery":{
    "OwnerId": "149977607591",
    "IpamResourceDiscoveryId": "ipam-res-disco-0257046d8aa78b8bc",
    "IpamResourceDiscoveryArn": "arn:aws:ec2::149977607591:ipam-resource-
discovery/ipam-res-disco-0257046d8aa78b8bc",
    "IpamResourceDiscoveryRegion": "us-east-1",
    "Description": "'Example-resource-discovery'",
    "OperatingRegions":[
      {"RegionName": "us-west-1"},
      {"RegionName": "us-west-2"},
      {"RegionName": "us-east-1"}
    ],
    "IsDefault": false,
    "State": "create-in-progress",
    "Tags": [
      {
        "Key": "cost-center",
        "Value": "cc123"
      }
    ]
  }
}
```

Once you create a resource discovery, you may want to share it with another IPAM delegated admin, which you can do with [create-resource-share](#). For more information, see [Integrate IPAM with accounts outside of your organization](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [CreateIpamResourceDiscovery](#) in *AWS CLI Command Reference*.

create-ipam-scope

The following code example shows how to use `create-ipam-scope`.

AWS CLI

To create an IPAM scope

The following `create-ipam-scope` example creates an IPAM scope.

(Linux):

```
aws ec2 create-ipam-scope \  
  --ipam-id ipam-08440e7a3acde3908 \  
  --description "Example description" \  
  --tag-specifications 'ResourceType=ipam-scope,Tags=[{Key=Name,Value="Example  
name value"}]'
```

(Windows):

```
aws ec2 create-ipam-scope ^  
  --ipam-id ipam-08440e7a3acde3908 ^  
  --description "Example description" ^  
  --tag-specifications ResourceType=ipam-scope,Tags=[{Key=Name,Value="Example name  
value"}]
```

Output:

```
{  
  "IpamScope": {  
    "OwnerId": "123456789012",  
    "IpamScopeId": "ipam-scope-01c1ebab2b63bd7e4",  
    "IpamScopeArn": "arn:aws:ec2::123456789012:ipam-scope/ipam-  
scope-01c1ebab2b63bd7e4",  
    "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",  
    "IpamRegion": "us-east-1",  
    "IpamScopeType": "private",  
    "IsDefault": false,  
    "Description": "Example description",  
    "PoolCount": 0,  
    "State": "create-in-progress",  
    "Tags": [  
      {  
        "Key": "Name",  
        "Value": "Example name value"  
      }  
    ]  
  }  
}
```

For more information, see [Create additional scopes](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [CreateIpamScope](#) in *AWS CLI Command Reference*.

create-ipam

The following code example shows how to use `create-ipam`.

AWS CLI

To create an IPAM

The following `create-ipam` example creates an IPAM.

(Linux):

```
aws ec2 create-ipam \  
  --description "Example description" \  
  --operating-regions "RegionName=us-east-2" "RegionName=us-west-1" \  
  --tag-specifications 'ResourceType=ipam,Tags=[{Key=Name,Value=ExampleIPAM}]'
```

(Windows):

```
aws ec2 create-ipam ^  
  --description "Example description" ^  
  --operating-regions "RegionName=us-east-2" "RegionName=us-west-1" ^  
  --tag-specifications ResourceType=ipam,Tags=[{Key=Name,Value=ExampleIPAM}]
```

Output:

```
{  
  "Ipam": {  
    "OwnerId": "123456789012",  
    "IpamId": "ipam-036486dfa6af58ee0",  
    "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-036486dfa6af58ee0",  
    "IpamRegion": "us-east-1",  
    "PublicDefaultScopeId": "ipam-scope-071b8042b0195c183",  
    "PrivateDefaultScopeId": "ipam-scope-0807405dece705a30",  
    "ScopeCount": 2,  
    "OperatingRegions": [  
      {  
        "RegionName": "us-east-2"      }  
    ]  
  }  
}
```

```
    },
    {
      "RegionName": "us-west-1"
    },
    {
      "RegionName": "us-east-1"
    }
  ],
  "State": "create-in-progress",
  "Tags": [
    {
      "Key": "Name",
      "Value": "ExampleIPAM"
    }
  ]
}
```

For more information, see [Create an IPAM](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [CreateIpam](#) in *AWS CLI Command Reference*.

create-key-pair

The following code example shows how to use `create-key-pair`.

AWS CLI

To create a key pair

This example creates a key pair named `MyKeyPair`.

Command:

```
aws ec2 create-key-pair --key-name MyKeyPair
```

The output is an ASCII version of the private key and key fingerprint. You need to save the key to a file.

For more information, see *Using Key Pairs* in the *AWS Command Line Interface User Guide*.

- For API details, see [CreateKeyPair](#) in *AWS CLI Command Reference*.

create-launch-template-version

The following code example shows how to use `create-launch-template-version`.

AWS CLI

To create a launch template version

This example creates a new launch template version based on version 1 of the launch template and specifies a different AMI ID.

Command:

```
aws ec2 create-launch-template-version --launch-template-id lt-0abcd290751193123
--version-description WebVersion2 --source-version 1 --launch-template-data
'{"ImageId":"ami-c998b6b2"}'
```

Output:

```
{
  "LaunchTemplateVersion": {
    "VersionDescription": "WebVersion2",
    "LaunchTemplateId": "lt-0abcd290751193123",
    "LaunchTemplateName": "WebServers",
    "VersionNumber": 2,
    "CreatedBy": "arn:aws:iam::123456789012:root",
    "LaunchTemplateData": {
      "ImageId": "ami-c998b6b2",
      "InstanceType": "t2.micro",
      "NetworkInterfaces": [
        {
          "Ipv6Addresses": [
            {
              "Ipv6Address": "2001:db8:1234:1a00::123"
            }
          ],
          "DeviceIndex": 0,
          "SubnetId": "subnet-7b16de0c",
          "AssociatePublicIpAddress": true
        }
      ]
    },
    "DefaultVersion": false,
    "CreateTime": "2017-12-01T13:35:46.000Z"
  }
}
```



```
}
}
```

- For API details, see [CreateLaunchTemplateVersion](#) in *AWS CLI Command Reference*.

create-launch-template

The following code example shows how to use `create-launch-template`.

AWS CLI

Example 1: To create a launch template

The following `create-launch-template` example creates a launch template that specifies the subnet in which to launch the instance, assigns a public IP address and an IPv6 address to the instance, and creates a tag for the instance.

```
aws ec2 create-launch-template \
  --launch-template-name TemplateForWebServer \
  --version-description WebVersion1 \
  --launch-template-data '{"NetworkInterfaces":
[{"AssociatePublicIpAddress":true,"DeviceIndex":0,"Ipv6AddressCount":1,"SubnetId":"subnet-7b
[{"ResourceType":"instance","Tags":[{"Key":"purpose","Value":"webserver"}]}'
```

Output:

```
{
  "LaunchTemplate": {
    "LatestVersionNumber": 1,
    "LaunchTemplateId": "lt-01238c059e3466abc",
    "LaunchTemplateName": "TemplateForWebServer",
    "DefaultVersionNumber": 1,
    "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
    "CreateTime": "2019-01-27T09:13:24.000Z"
  }
}
```

For more information, see [Launching an Instance from a Launch Template](#) in the *Amazon Elastic Compute Cloud User Guide*. For information about quoting JSON-formatted parameters, see [Quoting Strings](#) in the *AWS Command Line Interface User Guide*.

Example 2: To create a launch template for Amazon EC2 Auto Scaling

The following `create-launch-template` example creates a launch template with multiple tags and a block device mapping to specify an additional EBS volume when an instance launches. Specify a value for `Groups` that corresponds to security groups for the VPC that your Auto Scaling group will launch instances into. Specify the VPC and subnets as properties of the Auto Scaling group.

```
aws ec2 create-launch-template \
  --launch-template-name TemplateForAutoScaling \
  --version-description AutoScalingVersion1 \
  --launch-template-data '{"NetworkInterfaces":
[{"DeviceIndex":0,"AssociatePublicIpAddress":true,"Groups":
["sg-7c227019,sg-903004f8"],"DeleteOnTermination":true}],"ImageId":"ami-
b42209de","InstanceType":"m4.large","TagSpecifications":
[{"ResourceType":"instance","Tags":[{"Key":"environment","Value":"production"},
{"Key":"purpose","Value":"webserver"}]},{"ResourceType":"volume","Tags":
[{"Key":"environment","Value":"production"},{"Key":"cost-
center","Value":"cc123"}]}],"BlockDeviceMappings":[{"DeviceName":"/dev/sda1","Ebs":
{"VolumeSize":100}}]}' --region us-east-1
```

Output:

```
{
  "LaunchTemplate": {
    "LatestVersionNumber": 1,
    "LaunchTemplateId": "lt-0123c79c33a54e0abc",
    "LaunchTemplateName": "TemplateForAutoScaling",
    "DefaultVersionNumber": 1,
    "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
    "CreateTime": "2019-04-30T18:16:06.000Z"
  }
}
```

For more information, see [Creating a Launch Template for an Auto Scaling Group in the *Amazon EC2 Auto Scaling User Guide*](#). For information about quoting JSON-formatted parameters, see [Quoting Strings in the *AWS Command Line Interface User Guide*](#).

Example 3: To create a launch template that specifies encryption of EBS volumes

The following `create-launch-template` example creates a launch template that includes encrypted EBS volumes created from an unencrypted snapshot. It also tags the volumes during creation. If encryption by default is disabled, you must specify the `"Encrypted"` option as

shown in the following example. If you use the "KmsKeyId" option to specify a customer managed CMK, you also must specify the "Encrypted" option even if encryption by default is enabled.

```
aws ec2 create-launch-template \  
  --launch-template-name TemplateForEncryption \  
  --launch-template-data file://config.json
```

Contents of config.json:

```
{  
  "BlockDeviceMappings":[  
    {  
      "DeviceName":"/dev/sda1",  
      "Ebs":{  
        "VolumeType":"gp2",  
        "DeleteOnTermination":true,  
        "SnapshotId":"snap-066877671789bd71b",  
        "Encrypted":true,  
        "KmsKeyId":"arn:aws:kms:us-east-1:012345678910:key/abcd1234-  
a123-456a-a12b-a123b4cd56ef"  
      }  
    }  
  ],  
  "ImageId":"ami-00068cd7555f543d5",  
  "InstanceType":"c5.large",  
  "TagSpecifications":[  
    {  
      "ResourceType":"volume",  
      "Tags":[  
        {  
          "Key":"encrypted",  
          "Value":"yes"  
        }  
      ]  
    }  
  ]  
}
```

Output:

```
{
```

```

    "LaunchTemplate": {
      "LatestVersionNumber": 1,
      "LaunchTemplateId": "lt-0d5bd51bcf8530abc",
      "LaunchTemplateName": "TemplateForEncryption",
      "DefaultVersionNumber": 1,
      "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
      "CreateTime": "2020-01-07T19:08:36.000Z"
    }
  }
}

```

For more information, see [Restoring an Amazon EBS Volume from a Snapshot and Encryption by Default](#) in the *Amazon Elastic Compute Cloud User Guide*.

- For API details, see [CreateLaunchTemplate](#) in *AWS CLI Command Reference*.

create-local-gateway-route-table-virtual-interface-group-association

The following code example shows how to use `create-local-gateway-route-table-virtual-interface-group-association`.

AWS CLI

To associate a local gateway route table with a virtual interfaces (VIFs) group

The following `create-local-gateway-route-table-virtual-interface-group-association` example creates an association between the specified local gateway route table and VIF group.

```

aws ec2 create-local-gateway-route-table-virtual-interface-group-association \
  --local-gateway-route-table-id lgw-rtb-exampleidabcd1234 \
  --local-gateway-virtual-interface-group-id lgw-vif-grp-exampleid0123abcd

```

Output:

```

{
  "LocalGatewayRouteTableVirtualInterfaceGroupAssociation": {
    "LocalGatewayRouteTableVirtualInterfaceGroupAssociationId": "lgw-vif-grp-
assoc-exampleid12345678",
    "LocalGatewayVirtualInterfaceGroupId": "lgw-vif-grp-exampleid0123abcd",
    "LocalGatewayId": "lgw-exampleid11223344",
    "LocalGatewayRouteTableId": "lgw-rtb-exampleidabcd1234",

```

```
    "LocalGatewayRouteTableArn": "arn:aws:ec2:us-west-2:111122223333:local-
gateway-route-table/lgw-rtb-exampleidabcd1234",
    "OwnerId": "111122223333",
    "State": "pending",
    "Tags": []
  }
}
```

For more information, see [VIF group associations](#) in the *AWS Outposts User Guide*.

- For API details, see [CreateLocalGatewayRouteTableVirtualInterfaceGroupAssociation](#) in *AWS CLI Command Reference*.

create-local-gateway-route-table-vpc-association

The following code example shows how to use `create-local-gateway-route-table-vpc-association`.

AWS CLI

To associate a VPC with a route table

The following `create-local-gateway-route-table-vpc-association` example associates the specified VPC with the specified local gateway route table.

```
aws ec2 create-local-gateway-route-table-vpc-association \
  --local-gateway-route-table-id lgw-rtb-059615ef7dEXAMPLE \
  --vpc-id vpc-07ef66ac71EXAMPLE
```

Output:

```
{
  "LocalGatewayRouteTableVpcAssociation": {
    "LocalGatewayRouteTableVpcAssociationId": "lgw-vpc-assoc-0ee765bcc8EXAMPLE",
    "LocalGatewayRouteTableId": "lgw-rtb-059615ef7dEXAMPLE",
    "LocalGatewayId": "lgw-09b493aa7cEXAMPLE",
    "VpcId": "vpc-07ef66ac71EXAMPLE",
    "State": "associated"
  }
}
```

- For API details, see [CreateLocalGatewayRouteTableVpcAssociation](#) in *AWS CLI Command Reference*.

create-local-gateway-route-table

The following code example shows how to use `create-local-gateway-route-table`.

AWS CLI

To create a local gateway route table

The following `create-local-gateway-route-table` example creates a local gateway route table with the direct VPC routing mode.

```
aws ec2 create-local-gateway-route-table \  
  --local-gateway-id lgw-1a2b3c4d5e6f7g8h9 \  
  --mode direct-vpc-routing
```

Output:

```
{  
  "LocalGatewayRouteTable": {  
    "LocalGatewayRouteTableId": "lgw-rtb-abcdefg1234567890",  
    "LocalGatewayRouteTableArn": "arn:aws:ec2:us-west-2:111122223333:local-gateway-route-table/lgw-rtb-abcdefg1234567890",  
    "LocalGatewayId": "lgw-1a2b3c4d5e6f7g8h9",  
    "OutpostArn": "arn:aws:outposts:us-west-2:111122223333:outpost/op-021345abcdef67890",  
    "OwnerId": "111122223333",  
    "State": "pending",  
    "Tags": [],  
    "Mode": "direct-vpc-routing"  
  }  
}
```

For more information, see [Local gateway route tables](#) in the *AWS Outposts User Guide*.

- For API details, see [CreateLocalGatewayRouteTable](#) in *AWS CLI Command Reference*.

create-local-gateway-route

The following code example shows how to use `create-local-gateway-route`.

AWS CLI

To create a static route for a local gateway route table

The following `create-local-gateway-route` example creates the specified route in the specified local gateway route table.

```
aws ec2 create-local-gateway-route \  
  --destination-cidr-block 0.0.0.0/0 \  
  --local-gateway-route-table-id lgw-rtb-059615ef7dEXAMPLE
```

Output:

```
{  
  "Route": {  
    "DestinationCidrBlock": "0.0.0.0/0",  
    "LocalGatewayVirtualInterfaceGroupId": "lgw-vif-grp-07145b276bEXAMPLE",  
    "Type": "static",  
    "State": "deleted",  
    "LocalGatewayRouteTableId": "lgw-rtb-059615ef7dEXAMPLE"  
  }  
}
```

- For API details, see [CreateLocalGatewayRoute](#) in *AWS CLI Command Reference*.

create-managed-prefix-list

The following code example shows how to use `create-managed-prefix-list`.

AWS CLI

To create a prefix list

The following `create-managed-prefix-list` example creates an IPv4 prefix list with a maximum of 10 entries, and creates 2 entries in the prefix list.

```
aws ec2 create-managed-prefix-list \  
  --address-family IPv4 \  
  --max-entries 10 \  
  --entries Cidr=10.0.0.0/16,Description=vpc-a Cidr=10.2.0.0/16,Description=vpc-b \  
  --prefix-list-name vpc-cidrs
```

Output:

```
{
  "PrefixList": {
    "PrefixListId": "pl-0123456abcabcabc1",
    "AddressFamily": "IPv4",
    "State": "create-in-progress",
    "PrefixListArn": "arn:aws:ec2:us-west-2:123456789012:prefix-list/
pl-0123456abcabcabc1",
    "PrefixListName": "vpc-cidrs",
    "MaxEntries": 10,
    "Version": 1,
    "Tags": [],
    "OwnerId": "123456789012"
  }
}
```

For more information, see [Managed prefix lists](#) in the *Amazon VPC User Guide*.

- For API details, see [CreateManagedPrefixList](#) in *AWS CLI Command Reference*.

create-nat-gateway

The following code example shows how to use `create-nat-gateway`.

AWS CLI**Example 1: To create a public NAT gateway**

The following `create-nat-gateway` example creates a public NAT gateway in the specified subnet and associates the Elastic IP address with the specified allocation ID. When you create a public NAT gateway, you must associate an Elastic IP address.

```
aws ec2 create-nat-gateway \
  --subnet-id subnet-0250c25a1fEXAMPLE \
  --allocation-id eipalloc-09ad461b0dEXAMPLE
```

Output:

```
{
  "NatGateway": {
    "CreateTime": "2021-12-01T22:22:38.000Z",
```



```

    "NatGatewayAddresses": [
      {
        "AllocationId": "eipalloc-09ad461b0dEXAMPLE"
      }
    ],
    "NatGatewayId": "nat-0c61bf8a12EXAMPLE",
    "State": "pending",
    "SubnetId": "subnet-0250c25a1fEXAMPLE",
    "VpcId": "vpc-0a60eb65b4EXAMPLE",
    "ConnectivityType": "public"
  }
}

```

For more information, see [NAT gateways](#) in the *Amazon VPC User Guide*.

Example 2: To create a private NAT gateway

The following `create-nat-gateway` example creates a private NAT gateway in the specified subnet. A private NAT gateway does not have an associated Elastic IP address.

```

aws ec2 create-nat-gateway \
  --subnet-id subnet-0250c25a1fEXAMPLE \
  --connectivity-type private

```

Output:

```

{
  "NatGateway": {
    "CreateTime": "2021-12-01T22:26:00.000Z",
    "NatGatewayAddresses": [
      {}
    ],
    "NatGatewayId": "nat-011b568379EXAMPLE",
    "State": "pending",
    "SubnetId": "subnet-0250c25a1fEXAMPLE",
    "VpcId": "vpc-0a60eb65b4EXAMPLE",
    "ConnectivityType": "private"
  }
}

```

For more information, see [NAT gateways](#) in the *Amazon VPC User Guide*.

- For API details, see [CreateNatGateway](#) in *AWS CLI Command Reference*.

create-network-acl-entry

The following code example shows how to use `create-network-acl-entry`.

AWS CLI

To create a network ACL entry

This example creates an entry for the specified network ACL. The rule allows ingress traffic from any IPv4 address (0.0.0.0/0) on UDP port 53 (DNS) into any associated subnet. If the command succeeds, no output is returned.

Command:

```
aws ec2 create-network-acl-entry --network-acl-id acl-5fb85d36 --ingress --rule-number 100 --protocol udp --port-range From=53,To=53 --cidr-block 0.0.0.0/0 --rule-action allow
```

This example creates a rule for the specified network ACL that allows ingress traffic from any IPv6 address (::/0) on TCP port 80 (HTTP).

Command:

```
aws ec2 create-network-acl-entry --network-acl-id acl-5fb85d36 --ingress --rule-number 120 --protocol tcp --port-range From=80,To=80 --ipv6-cidr-block ::/0 --rule-action allow
```

- For API details, see [CreateNetworkAclEntry](#) in *AWS CLI Command Reference*.

create-network-acl

The following code example shows how to use `create-network-acl`.

AWS CLI

To create a network ACL

This example creates a network ACL for the specified VPC.

Command:

```
aws ec2 create-network-acl --vpc-id vpc-a01106c2
```

Output:

```
{
  "NetworkAcl": {
    "Associations": [],
    "NetworkAclId": "acl-5fb85d36",
    "VpcId": "vpc-a01106c2",
    "Tags": [],
    "Entries": [
      {
        "CidrBlock": "0.0.0.0/0",
        "RuleNumber": 32767,
        "Protocol": "-1",
        "Egress": true,
        "RuleAction": "deny"
      },
      {
        "CidrBlock": "0.0.0.0/0",
        "RuleNumber": 32767,
        "Protocol": "-1",
        "Egress": false,
        "RuleAction": "deny"
      }
    ],
    "IsDefault": false
  }
}
```

- For API details, see [CreateNetworkAcl](#) in *AWS CLI Command Reference*.

create-network-insights-access-scope

The following code example shows how to use `create-network-insights-access-scope`.

AWS CLI

To create a Network Access Scope

The following `create-network-insights-access-scope` example creates a Network Access Scope.

```
aws ec2 create-network-insights-access-scope \  
  --cli-input-json file://access-scope-file.json
```

Contents of `access-scope-file.json`:

```
{  
  "MatchPaths": [  
    {  
      "Source": {  
        "ResourceStatement": {  
          "Resources": [  
            "vpc-abcd12e3"  
          ]  
        }  
      }  
    },  
  ],  
  "ExcludePaths": [  
    {  
      "Source": {  
        "ResourceStatement": {  
          "ResourceTypes": [  
            "AWS::EC2::InternetGateway"  
          ]  
        }  
      }  
    }  
  ]  
}
```

Output:

```
{  
  "NetworkInsightsAccessScope": {  
    "NetworkInsightsAccessScopeId": "nis-123456789abc01234",  
    "NetworkInsightsAccessScopeArn": "arn:aws:ec2:us-  
east-1:123456789012:network-insights-access-scope/nis-123456789abc01234",  
    "CreateDate": "2022-01-25T19:20:28.796000+00:00",  
    "UpdatedDate": "2022-01-25T19:20:28.797000+00:00"  
  },  
  "NetworkInsightsAccessScopeContent": {  
    "NetworkInsightsAccessScopeId": "nis-123456789abc01234",
```

```
    "MatchPaths": [
      {
        "Source": {
          "ResourceStatement": {
            "Resources": [
              "vpc-abcd12e3"
            ]
          }
        }
      },
      {
        "ExcludePaths": [
          {
            "Source": {
              "ResourceStatement": {
                "ResourceTypes": [
                  "AWS::EC2::InternetGateway"
                ]
              }
            }
          }
        ]
      }
    ]
  }
}
```

For more information, see [Getting started with Network Access Analyzer using the AWS CLI](#) in the *Network Access Analyzer Guide*.

- For API details, see [CreateNetworkInsightsAccessScope](#) in *AWS CLI Command Reference*.

create-network-insights-path

The following code example shows how to use `create-network-insights-path`.

AWS CLI

To create a path

The following `create-network-insights-path` example creates a path. The source is the specified internet gateway and the destination is the specified EC2 instance. To determine whether the destination is reachable using the specified protocol and port, analyze the path using the `start-network-insights-analysis` command.

```
aws ec2 create-network-insights-path \  
  --source igw-0797cccdc9d73b0e5 \  
  --destination i-0495d385ad28331c7 \  
  --destination-port 22 \  
  --protocol TCP
```

Output:

```
{  
  "NetworkInsightsPaths": {  
    "NetworkInsightsPathId": "nip-0b26f224f1d131fa8",  
    "NetworkInsightsPathArn": "arn:aws:ec2:us-east-1:123456789012:network-  
insights-path/nip-0b26f224f1d131fa8",  
    "CreateDate": "2021-01-20T22:43:46.933Z",  
    "Source": "igw-0797cccdc9d73b0e5",  
    "Destination": "i-0495d385ad28331c7",  
    "Protocol": "tcp"  
  }  
}
```

For more information, see [Getting started using the AWS CLI](#) in the *Reachability Analyzer Guide*.

- For API details, see [CreateNetworkInsightsPath](#) in *AWS CLI Command Reference*.

create-network-interface-permission

The following code example shows how to use `create-network-interface-permission`.

AWS CLI

To create a network interface permission

This example grants permission to account 123456789012 to attach network interface `eni-1a2b3c4d` to an instance.

Command:

```
aws ec2 create-network-interface-permission --network-interface-id eni-1a2b3c4d --  
aws-account-id 123456789012 --permission INSTANCE-ATTACH
```

Output:

```
{
  "InterfacePermission": {
    "PermissionState": {
      "State": "GRANTED"
    },
    "NetworkInterfacePermissionId": "eni-perm-06fd19020ede149ea",
    "NetworkInterfaceId": "eni-1a2b3c4d",
    "Permission": "INSTANCE-ATTACH",
    "AwsAccountId": "123456789012"
  }
}
```

- For API details, see [CreateNetworkInterfacePermission](#) in *AWS CLI Command Reference*.

create-network-interface

The following code example shows how to use `create-network-interface`.

AWS CLI

Example 1: To specify an IPv4 address for a network interface

The following `create-network-interface` example creates a network interface for the specified subnet with the specified primary IPv4 address.

```
aws ec2 create-network-interface \
  --subnet-id subnet-00a24d0d67acf6333 \
  --description "my network interface" \
  --groups sg-09dfba7ed20cda78b \
  --private-ip-address 10.0.8.17
```

Output:

```
{
  "NetworkInterface": {
    "AvailabilityZone": "us-west-2a",
    "Description": "my network interface",
    "Groups": [
      {
        "GroupName": "my-security-group",
        "GroupId": "sg-09dfba7ed20cda78b"
      }
    ]
  }
}
```

```

    }
  ],
  "InterfaceType": "interface",
  "Ipv6Addresses": [],
  "MacAddress": "06:6a:0f:9a:49:37",
  "NetworkInterfaceId": "eni-0492b355f0cf3b3f8",
  "OwnerId": "123456789012",
  "PrivateDnsName": "ip-10-0-8-18.us-west-2.compute.internal",
  "PrivateIpAddress": "10.0.8.17",
  "PrivateIpAddresses": [
    {
      "Primary": true,
      "PrivateDnsName": "ip-10-0-8-17.us-west-2.compute.internal",
      "PrivateIpAddress": "10.0.8.17"
    }
  ],
  "RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",
  "RequesterManaged": false,
  "SourceDestCheck": true,
  "Status": "pending",
  "SubnetId": "subnet-00a24d0d67acf6333",
  "TagSet": [],
  "VpcId": "vpc-02723a0feeeb9d57b"
}
}

```

Example 2: To create a network interface with an IPv4 address and an IPv6 address

The following `create-network-interface` example creates a network interface for the specified subnet with an IPv4 address and an IPv6 address that are selected by Amazon EC2.

```

aws ec2 create-network-interface \
  --subnet-id subnet-00a24d0d67acf6333 \
  --description "my dual stack network interface" \
  --ipv6-address-count 1 \
  --groups sg-09dfba7ed20cda78b

```

Output:

```

{
  "NetworkInterface": {
    "AvailabilityZone": "us-west-2a",

```



```
"Description": "my dual stack network interface",
"Groups": [
  {
    "GroupName": "my-security-group",
    "GroupId": "sg-09dfba7ed20cda78b"
  }
],
"InterfaceType": "interface",
"Ipv6Addresses": [
  {
    "Ipv6Address": "2600:1f13:cfe:3650:a1dc:237c:393a:4ba7",
    "IsPrimaryIpv6": false
  }
],
"MacAddress": "06:b8:68:d2:b2:2d",
"NetworkInterfaceId": "eni-05da417453f9a84bf",
"OwnerId": "123456789012",
"PrivateDnsName": "ip-10-0-8-18.us-west-2.compute.internal",
"PrivateIpAddress": "10.0.8.18",
"PrivateIpAddresses": [
  {
    "Primary": true,
    "PrivateDnsName": "ip-10-0-8-18.us-west-2.compute.internal",
    "PrivateIpAddress": "10.0.8.18"
  }
],
"RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",
"RequesterManaged": false,
"SourceDestCheck": true,
"Status": "pending",
"SubnetId": "subnet-00a24d0d67acf6333",
"TagSet": [],
"VpcId": "vpc-02723a0feeb9d57b",
"Ipv6Address": "2600:1f13:cfe:3650:a1dc:237c:393a:4ba7"
}
}
```

Example 3: To create a network interface with connection tracking configuration options

The following `create-network-interface` example creates a network interface and configures the idle connection tracking timeouts.

```
aws ec2 create-network-interface \
```

```
--subnet-id subnet-00a24d0d67acf6333 \  
--groups sg-02e57dbcfe0331c1b \  
--connection-tracking-specification TcpEstablishedTimeout=86400,UdpTimeout=60
```

Output:

```
{  
  "NetworkInterface": {  
    "AvailabilityZone": "us-west-2a",  
    "ConnectionTrackingConfiguration": {  
      "TcpEstablishedTimeout": 86400,  
      "UdpTimeout": 60  
    },  
    "Description": "",  
    "Groups": [  
      {  
        "GroupName": "my-security-group",  
        "GroupId": "sg-02e57dbcfe0331c1b"  
      }  
    ],  
    "InterfaceType": "interface",  
    "Ipv6Addresses": [],  
    "MacAddress": "06:4c:53:de:6d:91",  
    "NetworkInterfaceId": "eni-0c133586e08903d0b",  
    "OwnerId": "123456789012",  
    "PrivateDnsName": "ip-10-0-8-94.us-west-2.compute.internal",  
    "PrivateIpAddress": "10.0.8.94",  
    "PrivateIpAddresses": [  
      {  
        "Primary": true,  
        "PrivateDnsName": "ip-10-0-8-94.us-west-2.compute.internal",  
        "PrivateIpAddress": "10.0.8.94"  
      }  
    ],  
    "RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",  
    "RequesterManaged": false,  
    "SourceDestCheck": true,  
    "Status": "pending",  
    "SubnetId": "subnet-00a24d0d67acf6333",  
    "TagSet": [],  
    "VpcId": "vpc-02723a0feeeb9d57b"  
  }  
}
```

Example 4: To create an Elastic Fabric Adapter

The following `create-network-interface` example creates an EFA.

```
aws ec2 create-network-interface \  
  --interface-type efa \  
  --subnet-id subnet-00a24d0d67acf6333 \  
  --description "my efa" \  
  --groups sg-02e57dbcf0331c1b
```

Output:

```
{  
  "NetworkInterface": {  
    "AvailabilityZone": "us-west-2a",  
    "Description": "my efa",  
    "Groups": [  
      {  
        "GroupName": "my-efa-sg",  
        "GroupId": "sg-02e57dbcf0331c1b"  
      }  
    ],  
    "InterfaceType": "efa",  
    "Ipv6Addresses": [],  
    "MacAddress": "06:d7:a4:f7:4d:57",  
    "NetworkInterfaceId": "eni-034acc2885e862b65",  
    "OwnerId": "123456789012",  
    "PrivateDnsName": "ip-10-0-8-180.us-west-2.compute.internal",  
    "PrivateIpAddress": "10.0.8.180",  
    "PrivateIpAddresses": [  
      {  
        "Primary": true,  
        "PrivateDnsName": "ip-10-0-8-180.us-west-2.compute.internal",  
        "PrivateIpAddress": "10.0.8.180"  
      }  
    ],  
    "RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",  
    "RequesterManaged": false,  
    "SourceDestCheck": true,  
    "Status": "pending",  
    "SubnetId": "subnet-00a24d0d67acf6333",  
    "TagSet": [],  
    "VpcId": "vpc-02723a0feeb9d57b"
```

```
}  
}
```

For more information, see [Elastic network interfaces](#) in the *Amazon EC2 User Guide*.

- For API details, see [CreateNetworkInterface](#) in *AWS CLI Command Reference*.

create-placement-group

The following code example shows how to use `create-placement-group`.

AWS CLI

To create a placement group

This example command creates a placement group with the specified name.

Command:

```
aws ec2 create-placement-group --group-name my-cluster --strategy cluster
```

To create a partition placement group

This example command creates a partition placement group named `HDFS-Group-A` with five partitions.

Command:

```
aws ec2 create-placement-group --group-name HDFS-Group-A --strategy partition --  
partition-count 5
```

- For API details, see [CreatePlacementGroup](#) in *AWS CLI Command Reference*.

create-replace-root-volume-task

The following code example shows how to use `create-replace-root-volume-task`.

AWS CLI

Example 1: To restore a root volume to its initial launch state

The following `create-replace-root-volume-task` example restores the root volume of instance `i-0123456789abcdefa` to its initial launch state.

```
aws ec2 create-replace-root-volume-task \  
  --instance-id i-0123456789abcdefa
```

Output:

```
{  
  "ReplaceRootVolumeTask":  
  {  
    "InstanceId": "i-0123456789abcdefa",  
    "ReplaceRootVolumeTaskId": "replacevol-0111122223333abcd",  
    "TaskState": "pending",  
    "StartTime": "2022-03-14T15:06:38Z",  
    "Tags": []  
  }  
}
```

For more information, see [Replace a root volume](#) in the *Amazon Elastic Compute Cloud User Guide*.

Example 2: To restore a root volume to a specific snapshot

The following `create-replace-root-volume-task` example restores the root volume of instance `i-0123456789abcdefa` to snapshot `snap-0abcdef1234567890`.

```
aws ec2 create-replace-root-volume-task \  
  --instance-id i-0123456789abcdefa \  
  --snapshot-id snap-0abcdef1234567890
```

Output:

```
{  
  "ReplaceRootVolumeTask":  
  {  
    "InstanceId": "i-0123456789abcdefa",  
    "ReplaceRootVolumeTaskId": "replacevol-0555566667777abcd",  
    "TaskState": "pending",  
    "StartTime": "2022-03-14T15:16:28Z",  
    "Tags": []  
  }  
}
```

```
}
```

For more information, see [Replace a root volume](#) in the *Amazon Elastic Compute Cloud User Guide*.

- For API details, see [CreateReplaceRootVolumeTask](#) in *AWS CLI Command Reference*.

create-reserved-instances-listing

The following code example shows how to use `create-reserved-instances-listing`.

AWS CLI

To list a Reserved Instance in the Reserved Instance Marketplace

The following `create-reserved-instances-listing` example creates a listing for the specified Reserved Instance in the Reserved Instance Marketplace.

```
aws ec2 create-reserved-instances-listing \  
  --reserved-instances-id 5ec28771-05ff-4b9b-aa31-9e57dexample \  
  --instance-count 3 \  
  --price-schedules CurrencyCode=USD,Price=25.50 \  
  --client-token 550e8400-e29b-41d4-a716-446655440000
```

- For API details, see [CreateReservedInstancesListing](#) in *AWS CLI Command Reference*.

create-restore-image-task

The following code example shows how to use `create-restore-image-task`.

AWS CLI

To restore an AMI from an S3 bucket

The following `create-restore-image-task` example restores an AMI from an S3 bucket. Use the values for `S3ObjectKey` and `Bucket` from the `describe-store-image-tasks` output, specify the object key of the AMI and the name of the S3 bucket to which the AMI was copied, and specify the name for the restored AMI. The name must be unique for AMIs in the Region for this account. The restored AMI will receive a new AMI ID.

```
aws ec2 create-restore-image-task \  
  --image-name my-ami
```

```
--object-key ami-1234567890abcdef0.bin \  
--bucket my-ami-bucket \  
--name "New AMI Name"
```

Output:

```
{  
  "ImageId": "ami-0eab20fe36f83e1a8"  
}
```

For more information about storing and restoring an AMI using S3, see [Store and restore an AMI using S3](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ami-store-restore.html) <<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ami-store-restore.html>> in the *Amazon EC2 User Guide*.

- For API details, see [CreateRestoreImageTask](#) in *AWS CLI Command Reference*.

create-route-table

The following code example shows how to use `create-route-table`.

AWS CLI

To create a route table

This example creates a route table for the specified VPC.

Command:

```
aws ec2 create-route-table --vpc-id vpc-a01106c2
```

Output:

```
{  
  "RouteTable": {  
    "Associations": [],  
    "RouteTableId": "rtb-22574640",  
    "VpcId": "vpc-a01106c2",  
    "PropagatingVgws": [],  
    "Tags": [],  
    "Routes": [  
      {
```

```
        "GatewayId": "local",
        "DestinationCidrBlock": "10.0.0.0/16",
        "State": "active"
    }
]
}
}
```

- For API details, see [CreateRouteTable](#) in *AWS CLI Command Reference*.

create-route

The following code example shows how to use `create-route`.

AWS CLI

To create a route

This example creates a route for the specified route table. The route matches all IPv4 traffic (`0.0.0.0/0`) and routes it to the specified Internet gateway. If the command succeeds, no output is returned.

Command:

```
aws ec2 create-route --route-table-id rtb-22574640 --destination-cidr-block
0.0.0.0/0 --gateway-id igw-c0a643a9
```

This example command creates a route in route table `rtb-g8ff4ea2`. The route matches traffic for the IPv4 CIDR block `10.0.0.0/16` and routes it to VPC peering connection, `pcx-111aaa22`. This route enables traffic to be directed to the peer VPC in the VPC peering connection. If the command succeeds, no output is returned.

Command:

```
aws ec2 create-route --route-table-id rtb-g8ff4ea2 --destination-cidr-block
10.0.0.0/16 --vpc-peering-connection-id pcx-1a2b3c4d
```

This example creates a route in the specified route table that matches all IPv6 traffic (`::/0`) and routes it to the specified egress-only Internet gateway.

Command:


```
aws ec2 create-route --route-table-id rtb-dce620b8 --destination-ipv6-cidr-block ::/0 --egress-only-internet-gateway-id eigw-01eadbd45ecd7943f
```

- For API details, see [CreateRoute](#) in *AWS CLI Command Reference*.

create-security-group

The following code example shows how to use `create-security-group`.

AWS CLI

To create a security group for EC2-Classical

This example creates a security group named `MySecurityGroup`.

Command:

```
aws ec2 create-security-group --group-name MySecurityGroup --description "My security group"
```

Output:

```
{
  "GroupId": "sg-903004f8"
}
```

To create a security group for EC2-VPC

This example creates a security group named `MySecurityGroup` for the specified VPC.

Command:

```
aws ec2 create-security-group --group-name MySecurityGroup --description "My security group" --vpc-id vpc-1a2b3c4d
```

Output:

```
{
  "GroupId": "sg-903004f8"
}
```

For more information, see *Using Security Groups in the AWS Command Line Interface User Guide*.

- For API details, see [CreateSecurityGroup](#) in *AWS CLI Command Reference*.

create-snapshot

The following code example shows how to use `create-snapshot`.

AWS CLI

To create a snapshot

This example command creates a snapshot of the volume with a volume ID of `vol-1234567890abcdef0` and a short description to identify the snapshot.

Command:

```
aws ec2 create-snapshot --volume-id vol-1234567890abcdef0 --description "This is my root volume snapshot"
```

Output:

```
{
  "Description": "This is my root volume snapshot",
  "Tags": [],
  "Encrypted": false,
  "VolumeId": "vol-1234567890abcdef0",
  "State": "pending",
  "VolumeSize": 8,
  "StartTime": "2018-02-28T21:06:01.000Z",
  "Progress": "",
  "OwnerId": "012345678910",
  "SnapshotId": "snap-066877671789bd71b"
}
```

To create a snapshot with tags

This example command creates a snapshot and applies two tags: `purpose=prod` and `costcenter=123`.

Command:

```
aws ec2 create-snapshot --volume-id vol-1234567890abcdef0 --description 'Prod
backup' --tag-specifications 'ResourceType=snapshot,Tags=[{Key=purpose,Value=prod},
{Key=costcenter,Value=123}]'
```

Output:

```
{
  "Description": "Prod backup",
  "Tags": [
    {
      "Value": "prod",
      "Key": "purpose"
    },
    {
      "Value": "123",
      "Key": "costcenter"
    }
  ],
  "Encrypted": false,
  "VolumeId": "vol-1234567890abcdef0",
  "State": "pending",
  "VolumeSize": 8,
  "StartTime": "2018-02-28T21:06:06.000Z",
  "Progress": "",
  "OwnerId": "012345678910",
  "SnapshotId": "snap-09ed24a70bc19bbe4"
}
```

- For API details, see [CreateSnapshot](#) in *AWS CLI Command Reference*.

create-snapshots

The following code example shows how to use create-snapshots.

AWS CLI

Example 1: To create a multi-volume snapshot

The following create-snapshots example creates snapshots of all volumes attached to the specified instance.

```
aws ec2 create-snapshots \
```

```
--instance-specification InstanceId=i-1234567890abcdef0 \  
--description "This is snapshot of a volume from my-instance"
```

Output:

```
{  
  "Snapshots": [  
    {  
      "Description": "This is a snapshot of a volume from my-instance",  
      "Tags": [],  
      "Encrypted": false,  
      "VolumeId": "vol-0a01d2d5a34697479",  
      "State": "pending",  
      "VolumeSize": 16,  
      "StartTime": "2019-08-05T16:58:19.000Z",  
      "Progress": "",  
      "OwnerId": "123456789012",  
      "SnapshotId": "snap-07f30e3909aa0045e"  
    },  
    {  
      "Description": "This is a snapshot of a volume from my-instance",  
      "Tags": [],  
      "Encrypted": false,  
      "VolumeId": "vol-02d0d4947008cb1a2",  
      "State": "pending",  
      "VolumeSize": 20,  
      "StartTime": "2019-08-05T16:58:19.000Z",  
      "Progress": "",  
      "OwnerId": "123456789012",  
      "SnapshotId": "snap-0ec20b602264aad48"  
    },  
    ...  
  ]  
}
```

Example 2: To create a multi-volume snapshot with tags from the source volume

The following `create-snapshots` example creates snapshots of all volumes attached to the specified instance and copies the tags from each volume to its corresponding snapshot.

```
aws ec2 create-snapshots \  
  --instance-specification InstanceId=i-1234567890abcdef0 \  
  --copy-tags-from-source volume \  
  --instance-specification InstanceId=i-1234567890abcdef0 \  
  --copy-tags-from-source volume \  
  --description "This is snapshot of a volume from my-instance"
```

```
--description "This is snapshot of a volume from my-instance"
```

Output:

```
{
  "Snapshots": [
    {
      "Description": "This is a snapshot of a volume from my-instance",
      "Tags": [
        {
          "Key": "Name",
          "Value": "my-volume"
        }
      ],
      "Encrypted": false,
      "VolumeId": "vol-02d0d4947008cb1a2",
      "State": "pending",
      "VolumeSize": 20,
      "StartTime": "2019-08-05T16:53:04.000Z",
      "Progress": "",
      "OwnerId": "123456789012",
      "SnapshotId": "snap-053bfaeb821a458dd"
    }
    ...
  ]
}
```

Example 3: To create a multi-volume snapshot not including the root volume

The following `create-snapshots` example creates a snapshot of all volumes attached to the specified instance except for the root volume.

```
aws ec2 create-snapshots \
  --instance-specification InstanceId=i-1234567890abcdef0,ExcludeBootVolume=true
```

See example 1 for sample output.

Example 4: To create a multi-volume snapshot and add tags

The following `create-snapshots` example creates snapshots of all volumes attached to the specified instance and adds two tags to each snapshot.

```
aws ec2 create-snapshots \  
  --instance-specification InstanceId=i-1234567890abcdef0 \  
  --tag-specifications 'ResourceType=snapshot,Tags=[{Key=Name,Value=backup},  
{Key=costcenter,Value=123}]'
```

See example 1 for sample output.

- For API details, see [CreateSnapshots](#) in *AWS CLI Command Reference*.

create-spot-datafeed-subscription

The following code example shows how to use `create-spot-datafeed-subscription`.

AWS CLI

To create a Spot Instance data feed

The following `create-spot-datafeed-subscription` example creates a Spot Instance data feed.

```
aws ec2 create-spot-datafeed-subscription \  
  --bucket my-bucket \  
  --prefix spot-data-feed
```

Output:

```
{  
  "SpotDatafeedSubscription": {  
    "Bucket": "my-bucket",  
    "OwnerId": "123456789012",  
    "Prefix": "spot-data-feed",  
    "State": "Active"  
  }  
}
```

The data feed is stored in the Amazon S3 bucket that you specified. The file names for this data feed have the following format.

```
my-bucket.s3.amazonaws.com/spot-data-feed/123456789012.YYYY-MM-DD-HH.n.abcd1234.gz
```

For more information, see [Spot Instance data feed](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [CreateSpotDatafeedSubscription](#) in *AWS CLI Command Reference*.

create-store-image-task

The following code example shows how to use `create-store-image-task`.

AWS CLI

To store an AMI in an S3 bucket

The following `create-store-image-task` example stores an AMI in an S3 bucket. Specify the ID of the AMI and the name of the S3 bucket in which to store the AMI.

```
aws ec2 create-store-image-task \  
  --image-id ami-1234567890abcdef0 \  
  --bucket my-ami-bucket
```

Output:

```
{  
  "ObjectKey": "ami-1234567890abcdef0.bin"  
}
```

For more information, see [Store and restore an AMI using S3](#) in the *Amazon EC2 User Guide*.

- For API details, see [CreateStoreImageTask](#) in *AWS CLI Command Reference*.

create-subnet-cidr-reservation

The following code example shows how to use `create-subnet-cidr-reservation`.

AWS CLI

To create a subnet CIDR reservation

The following `create-subnet-cidr-reservation` example creates a subnet CIDR reservation for the specified subnet and CIDR range.

```
aws ec2 create-subnet-cidr-reservation \  
  --subnet-id subnet-1234567890abcdef0 \  
  --cidr-range 10.0.0.0/24
```

```
--subnet-id subnet-03c51e2eEXAMPLE \  
--reservation-type prefix \  
--cidr 10.1.0.20/26
```

Output:

```
{  
  "SubnetCidrReservation": {  
    "SubnetCidrReservationId": "scr-044f977c4eEXAMPLE",  
    "SubnetId": "subnet-03c51e2e6cEXAMPLE",  
    "Cidr": "10.1.0.16/28",  
    "ReservationType": "prefix",  
    "OwnerId": "123456789012"  
  }  
}
```

For more information, see [Subnet CIDR reservations](#) in the *Amazon VPC User Guide*.

- For API details, see [CreateSubnetCidrReservation](#) in *AWS CLI Command Reference*.

create-subnet

The following code example shows how to use create-subnet.

AWS CLI

Example 1: To create a subnet with an IPv4 CIDR block only

The following create-subnet example creates a subnet in the specified VPC with the specified IPv4 CIDR block.

```
aws ec2 create-subnet \  
  --vpc-id vpc-081ec835f3EXAMPLE \  
  --cidr-block 10.0.0.0/24 \  
  --tag-specifications ResourceType=subnet,Tags=[{Key=Name,Value=my-ipv4-only-  
subnet}]
```

Output:

```
{  
  "Subnet": {  
    "AvailabilityZone": "us-west-2a",
```



```

    "AvailabilityZoneId": "usw2-az2",
    "AvailableIpAddressCount": 251,
    "CidrBlock": "10.0.0.0/24",
    "DefaultForAz": false,
    "MapPublicIpOnLaunch": false,
    "State": "available",
    "SubnetId": "subnet-0e99b93155EXAMPLE",
    "VpcId": "vpc-081ec835f3EXAMPLE",
    "OwnerId": "123456789012",
    "AssignIpv6AddressOnCreation": false,
    "Ipv6CidrBlockAssociationSet": [],
    "Tags": [
      {
        "Key": "Name",
        "Value": "my-ipv4-only-subnet"
      }
    ],
    "SubnetArn": "arn:aws:ec2:us-west-2:123456789012:subnet/
subnet-0e99b93155EXAMPLE"
  }
}

```

Example 2: To create a subnet with both IPv4 and IPv6 CIDR blocks

The following `create-subnet` example creates a subnet in the specified VPC with the specified IPv4 and IPv6 CIDR blocks.

```

aws ec2 create-subnet \
  --vpc-id vpc-081ec835f3EXAMPLE \
  --cidr-block 10.0.0.0/24 \
  --ipv6-cidr-block 2600:1f16:cfe:3660::/64 \
  --tag-specifications ResourceType=subnet,Tags=[{Key=Name,Value=my-ipv4-ipv6-
subnet}]

```

Output:

```

{
  "Subnet": {
    "AvailabilityZone": "us-west-2a",
    "AvailabilityZoneId": "usw2-az2",
    "AvailableIpAddressCount": 251,
    "CidrBlock": "10.0.0.0/24",
    "DefaultForAz": false,

```

```

    "MapPublicIpOnLaunch": false,
    "State": "available",
    "SubnetId": "subnet-0736441d38EXAMPLE",
    "VpcId": "vpc-081ec835f3EXAMPLE",
    "OwnerId": "123456789012",
    "AssignIpv6AddressOnCreation": false,
    "Ipv6CidrBlockAssociationSet": [
      {
        "AssociationId": "subnet-cidr-assoc-06c5f904499fcc623",
        "Ipv6CidrBlock": "2600:1f13:cfe:3660::/64",
        "Ipv6CidrBlockState": {
          "State": "associating"
        }
      }
    ],
    "Tags": [
      {
        "Key": "Name",
        "Value": "my-ipv4-ipv6-subnet"
      }
    ],
    "SubnetArn": "arn:aws:ec2:us-west-2:123456789012:subnet/
subnet-0736441d38EXAMPLE"
  }
}

```

Example 3: To create a subnet with an IPv6 CIDR block only

The following `create-subnet` example creates a subnet in the specified VPC, with the specified IPv6 CIDR block.

```

aws ec2 create-subnet \
  --vpc-id vpc-081ec835f3EXAMPLE \
  --ipv6-native \
  --ipv6-cidr-block 2600:1f16:115:200::/64 \
  --tag-specifications ResourceType=subnet,Tags=[{Key=Name,Value=my-ipv6-only-
subnet}]

```

Output:

```

{
  "Subnet": {
    "AvailabilityZone": "us-west-2a",

```

```
"AvailabilityZoneId": "usw2-az2",
"AvailableIpAddressCount": 0,
"DefaultForAz": false,
"MapPublicIpOnLaunch": false,
"State": "available",
"SubnetId": "subnet-03f720e7deEXAMPLE",
"VpcId": "vpc-081ec835f3EXAMPLE",
"OwnerId": "123456789012",
"AssignIpv6AddressOnCreation": true,
"Ipv6CidrBlockAssociationSet": [
  {
    "AssociationId": "subnet-cidr-assoc-01ef639edde556709",
    "Ipv6CidrBlock": "2600:1f13:cfe:3660::/64",
    "Ipv6CidrBlockState": {
      "State": "associating"
    }
  }
],
"Tags": [
  {
    "Key": "Name",
    "Value": "my-ipv6-only-subnet"
  }
],
"SubnetArn": "arn:aws:ec2:us-west-2:123456789012:subnet/
subnet-03f720e7deEXAMPLE"
}
```

For more information, see [VPCs and subnets](#) in the *Amazon VPC User Guide*.

- For API details, see [CreateSubnet](#) in *AWS CLI Command Reference*.

create-tags

The following code example shows how to use `create-tags`.

AWS CLI

Example 1: To add a tag to a resource

The following `create-tags` example adds the tag `Stack=production` to the specified image, or overwrites an existing tag for the AMI where the tag key is `Stack`.

```
aws ec2 create-tags \  
  --resources ami-1234567890abcdef0 \  
  --tags Key=Stack,Value=production
```

For more information, see [This is the topic title](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

Example 2: To add tags to multiple resources

The following `create-tags` example adds (or overwrites) two tags for an AMI and an instance. One of the tags has a key (`webserver`) but no value (value is set to an empty string). The other tag has a key (`stack`) and a value (`Production`).

```
aws ec2 create-tags \  
  --resources ami-1a2b3c4d i-1234567890abcdef0 \  
  --tags Key=webserver,Value=   Key=stack,Value=Production
```

For more information, see [This is the topic title](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

Example 3: To add tags containing special characters

The following `create-tags` example adds the tag `[Group]=test` for an instance. The square brackets (`[` and `]`) are special characters, and must be escaped. The following examples also use the line continuation character appropriate for each environment.

If you are using Windows, surround the element that has special characters with double quotes (`"`), and then precede each double quote character with a backslash (`\`) as follows:

```
aws ec2 create-tags ^  
  --resources i-1234567890abcdef0 ^  
  --tags Key=\"[Group]\",Value=test
```

If you are using Windows PowerShell, surround the element the value that has special characters with double quotes (`"`), precede each double quote character with a backslash (`\`), and then surround the entire key and value structure with single quotes (`'`) as follows:

```
aws ec2 create-tags `  
  --resources i-1234567890abcdef0 `  
  --tags 'Key=\"[Group]\",Value=test'
```

If you are using Linux or OS X, surround the element that has special characters with double quotes ("), and then surround the entire key and value structure with single quotes (') as follows:

```
aws ec2 create-tags \  
  --resources i-1234567890abcdef0 \  
  --tags 'Key="[Group]",Value=test'
```

For more information, see [This is the topic title](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [CreateTags](#) in *AWS CLI Command Reference*.

create-traffic-mirror-filter-rule

The following code example shows how to use `create-traffic-mirror-filter-rule`.

AWS CLI

To create a filter rule for incoming TCP traffic

The following `create-traffic-mirror-filter-rule` example creates a rule that you can use to mirror all incoming TCP traffic. Before you run this command, use `create-traffic-mirror-filter` to create the the Traffic Mirror filter.

```
aws ec2 create-traffic-mirror-filter-rule \  
  --description "TCP Rule" \  
  --destination-cidr-block 0.0.0.0/0 \  
  --protocol 6 \  
  --rule-action accept \  
  --rule-number 1 \  
  --source-cidr-block 0.0.0.0/0 \  
  --traffic-direction ingress \  
  --traffic-mirror-filter-id tmf-04812ff784b25ae67
```

Output:

```
{  
  "TrafficMirrorFilterRule": {  
    "DestinationCidrBlock": "0.0.0.0/0",  
    "TrafficMirrorFilterId": "tmf-04812ff784b25ae67",  
    "TrafficMirrorFilterRuleId": "tmfr-02d20d996673f3732",  
    "SourceCidrBlock": "0.0.0.0/0",
```

```

    "TrafficDirection": "ingress",
    "Description": "TCP Rule",
    "RuleNumber": 1,
    "RuleAction": "accept",
    "Protocol": 6
  },
  "ClientToken": "4752b573-40a6-4eac-a8a4-a72058761219"
}

```

For more information, see [Create a Traffic Mirror Filter](#) in the *AWS Traffic Mirroring Guide*.

- For API details, see [CreateTrafficMirrorFilterRule](#) in *AWS CLI Command Reference*.

create-traffic-mirror-filter

The following code example shows how to use `create-traffic-mirror-filter`.

AWS CLI

To create a Traffic Mirror Filter

The following `create-traffic-mirror-filter` example creates a Traffic Mirror filter. After you create the filter, use `create-traffic-mirror-filter-rule` to add rules to the filter.

```

aws ec2 create-traffic-mirror-filter \
  --description "TCP Filter"

```

Output:

```

{
  "ClientToken": "28908518-100b-4987-8233-8c744EXAMPLE",
  "TrafficMirrorFilter": {
    "TrafficMirrorFilterId": "tmf-04812ff784EXAMPLE",
    "Description": "TCP Filter",
    "EgressFilterRules": [],
    "IngressFilterRules": [],
    "Tags": [],
    "NetworkServices": []
  }
}

```

For more information, see [Create a Traffic Mirror Filter](#) in the *AWS Traffic Mirroring Guide*.

- For API details, see [CreateTrafficMirrorFilter](#) in *AWS CLI Command Reference*.

create-traffic-mirror-session

The following code example shows how to use `create-traffic-mirror-session`.

AWS CLI

To create a Traffic Mirror Session

The following `create-traffic-mirror-session` command creates a traffic mirror sessions for the specified source and target for 25 bytes of the packet.

```
aws ec2 create-traffic-mirror-session \  
  --description "example session" \  
  --traffic-mirror-target-id tmt-07f75d8feeEXAMPLE \  
  --network-interface-id eni-070203f901EXAMPLE \  
  --session-number 1 \  
  --packet-length 25 \  
  --traffic-mirror-filter-id tmf-04812ff784EXAMPLE
```

Output:

```
{  
  "TrafficMirrorSession": {  
    "TrafficMirrorSessionId": "tms-08a33b1214EXAMPLE",  
    "TrafficMirrorTargetId": "tmt-07f75d8feeEXAMPLE",  
    "TrafficMirrorFilterId": "tmf-04812ff784EXAMPLE",  
    "NetworkInterfaceId": "eni-070203f901EXAMPLE",  
    "OwnerId": "111122223333",  
    "PacketLength": 25,  
    "SessionNumber": 1,  
    "VirtualNetworkId": 7159709,  
    "Description": "example session",  
    "Tags": []  
  },  
  "ClientToken": "5236cffc-ee13-4a32-bb5b-388d9da09d96"  
}
```

For more information, see [Create a Traffic Mirror Session](#) in the *AWS Traffic Mirroring Guide*.

- For API details, see [CreateTrafficMirrorSession](#) in *AWS CLI Command Reference*.

create-traffic-mirror-target

The following code example shows how to use create-traffic-mirror-target.

AWS CLI

To create a a Network Load Balancer Traffic Mirror target

The following create-traffic-mirror-target example creates a Network Load Balancer Traffic Mirror target.

```
aws ec2 create-traffic-mirror-target \  
  --description "Example Network Load Balancer Target" \  
  --network-load-balancer-arn arn:aws:elasticloadbalancing:us-  
east-1:111122223333:loadbalancer/net/NLB/7cdec873EXAMPLE
```

Output:

```
{  
  "TrafficMirrorTarget": {  
    "Type": "network-load-balancer",  
    "Tags": [],  
    "Description": "Example Network Load Balancer Target",  
    "OwnerId": "111122223333",  
    "NetworkLoadBalancerArn": "arn:aws:elasticloadbalancing:us-  
east-1:724145273726:loadbalancer/net/NLB/7cdec873EXAMPLE",  
    "TrafficMirrorTargetId": "tmt-0dabe9b0a6EXAMPLE"  
  },  
  "ClientToken": "d5c090f5-8a0f-49c7-8281-72c796a21f72"  
}
```

To create a network Traffic Mirror target

The following create-traffic-mirror-target example creates a network interface Traffic Mirror target.

```
aws ec2 create-traffic-mirror-target --description "Network interface target" --network-  
interface-id eni-eni-01f6f631eEXAMPLE
```

Output:

```
{
```



```

"ClientToken": "5289a345-0358-4e62-93d5-47ef3061d65e",
"TrafficMirrorTarget": {
  "Description": "Network interface target",
  "NetworkInterfaceId": "eni-01f6f631eEXAMPLE",
  "TrafficMirrorTargetId": "tmt-02dcdb2abEXAMPLE",
  "OwnerId": "111122223333",
  "Type": "network-interface",
  "Tags": []
}
}

```

For more information, see [Create a Traffic Mirror Target](#) in the *AWS Traffic Mirroring Guide*.

- For API details, see [CreateTrafficMirrorTarget](#) in *AWS CLI Command Reference*.

create-transit-gateway-connect-peer

The following code example shows how to use `create-transit-gateway-connect-peer`.

AWS CLI

To create a Transit Gateway Connect peer

The following `create-transit-gateway-connect-peer` example creates a Connect peer.

```

aws ec2 create-transit-gateway-connect-peer \
  --transit-gateway-attachment-id tgw-attach-0f0927767cEXAMPLE \
  --peer-address 172.31.1.11 \
  --inside-cidr-blocks 169.254.6.0/29

```

Output:

```

{
  "TransitGatewayConnectPeer": {
    "TransitGatewayAttachmentId": "tgw-attach-0f0927767cEXAMPLE",
    "TransitGatewayConnectPeerId": "tgw-connect-peer-0666adbac4EXAMPLE",
    "State": "pending",
    "CreationTime": "2021-10-13T03:35:17.000Z",
    "ConnectPeerConfiguration": {
      "TransitGatewayAddress": "10.0.0.234",
      "PeerAddress": "172.31.1.11",
      "InsideCidrBlocks": [

```

```

        "169.254.6.0/29"
    ],
    "Protocol": "gre",
    "BgpConfigurations": [
        {
            "TransitGatewayAsn": 64512,
            "PeerAsn": 64512,
            "TransitGatewayAddress": "169.254.6.2",
            "PeerAddress": "169.254.6.1",
            "BgpStatus": "down"
        },
        {
            "TransitGatewayAsn": 64512,
            "PeerAsn": 64512,
            "TransitGatewayAddress": "169.254.6.3",
            "PeerAddress": "169.254.6.1",
            "BgpStatus": "down"
        }
    ]
}
}
}
}
}

```

For more information, see [Transit gateway Connect attachments and Transit Gateway Connect peers](#) in the *Transit Gateways Guide*.

- For API details, see [CreateTransitGatewayConnectPeer](#) in *AWS CLI Command Reference*.

create-transit-gateway-connect

The following code example shows how to use `create-transit-gateway-connect`.

AWS CLI

To create a transit gateway Connect attachment

The following `create-transit-gateway-connect` example creates a Connect attachment, with the "gre" protocol, for the specified attachment.

```

aws ec2 create-transit-gateway-connect \
  --transport-transit-gateway-attachment-id tgw-attach-0a89069f57EXAMPLE \
  --options "Protocol=gre"

```

Output:

```
{
  "TransitGatewayConnect": {
    "TransitGatewayAttachmentId": "tgw-attach-037012e5dcEXAMPLE",
    "TransportTransitGatewayAttachmentId": "tgw-attach-0a89069f57EXAMPLE",
    "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",
    "State": "pending",
    "CreationTime": "2021-03-09T19:59:17+00:00",
    "Options": {
      "Protocol": "gre"
    }
  }
}
```

For more information, see [Transit gateway Connect attachments and Transit Gateway Connect peers](#) in the *Transit Gateways Guide*.

- For API details, see [CreateTransitGatewayConnect](#) in *AWS CLI Command Reference*.

create-transit-gateway-multicast-domain

The following code example shows how to use `create-transit-gateway-multicast-domain`.

AWS CLI**Example 1: To create an IGMP multicast domain**

The following `create-transit-gateway-multicast-domain` example creates a multicast domain for the specified transit gateway. With static sources disabled, any instances in subnets associated with the multicast domain can send multicast traffic. If at least one member uses the IGMP protocol, you must enable IGMPv2 support.

```
aws ec2 create-transit-gateway-multicast-domain \
  --transit-gateway-id tgw-0bf0bffefaEXAMPLE \
  --options StaticSourcesSupport=disable,Igmpv2Support=enable
```

Output:

```
{
  "TransitGatewayMulticastDomain": {
```

```

    "TransitGatewayMulticastDomainId": "tgw-mcast-domain-0c9e29e2a7EXAMPLE",
    "TransitGatewayId": "tgw-0bf0bfffefaEXAMPLE",
    "TransitGatewayMulticastDomainArn": "arn:aws:ec2:us-
west-2:123456789012:transit-gateway-multicast-domain/tgw-mcast-
domain-0c9e29e2a7EXAMPLE",
    "OwnerId": "123456789012",
    "Options": {
      "Icmpv2Support": "enable",
      "StaticSourcesSupport": "disable",
      "AutoAcceptSharedAssociations": "disable"
    },
    "State": "pending",
    "CreationTime": "2021-09-29T22:17:13.000Z"
  }
}

```

Example 2: To create a static multicast domain

The following `create-transit-gateway-multicast-domain` example creates a multicast domain for the specified transit gateway. With static sources enabled, you must statically add sources.

```

aws ec2 create-transit-gateway-multicast-domain \
  --transit-gateway-id tgw-0bf0bfffefaEXAMPLE \
  --options StaticSourcesSupport=enable,Icmpv2Support=disable

```

Output:

```

{
  "TransitGatewayMulticastDomain": {
    "TransitGatewayMulticastDomainId": "tgw-mcast-domain-000fb24d04EXAMPLE",
    "TransitGatewayId": "tgw-0bf0bfffefaEXAMPLE",
    "TransitGatewayMulticastDomainArn": "arn:aws:ec2:us-
west-2:123456789012:transit-gateway-multicast-domain/tgw-mcast-
domain-000fb24d04EXAMPLE",
    "OwnerId": "123456789012",
    "Options": {
      "Icmpv2Support": "disable",
      "StaticSourcesSupport": "enable",
      "AutoAcceptSharedAssociations": "disable"
    },
    "State": "pending",
  }
}

```

```
    "CreationTime": "2021-09-29T22:20:19.000Z"
  }
}
```

For more information, see [Managing multicast domains](#) in the *Transit Gateways Guide*.

- For API details, see [CreateTransitGatewayMulticastDomain](#) in *AWS CLI Command Reference*.

create-transit-gateway-peering-attachment

The following code example shows how to use `create-transit-gateway-peering-attachment`.

AWS CLI

To create a transit gateway peering attachment

The following `create-transit-gateway-peering-attachment` example creates a peering attachment request between the two specified transit gateways.

```
aws ec2 create-transit-gateway-peering-attachment \
  --transit-gateway-id tgw-123abc05e04123abc \
  --peer-transit-gateway-id tgw-11223344aabbcc112 \
  --peer-account-id 123456789012 \
  --peer-region us-east-2
```

Output:

```
{
  "TransitGatewayPeeringAttachment": {
    "TransitGatewayAttachmentId": "tgw-attach-4455667788aabbccd",
    "RequesterTgwInfo": {
      "TransitGatewayId": "tgw-123abc05e04123abc",
      "OwnerId": "123456789012",
      "Region": "us-west-2"
    },
    "AcceptorTgwInfo": {
      "TransitGatewayId": "tgw-11223344aabbcc112",
      "OwnerId": "123456789012",
      "Region": "us-east-2"
    },
    "State": "initiatingRequest",
```

```
    "CreationTime": "2019-12-09T11:38:05.000Z"
  }
}
```

For more information, see [Transit Gateway Peering Attachments](#) in the *Transit Gateways Guide*.

- For API details, see [CreateTransitGatewayPeeringAttachment](#) in *AWS CLI Command Reference*.

create-transit-gateway-policy-table

The following code example shows how to use `create-transit-gateway-policy-table`.

AWS CLI

To create a transit gateway policy table

The following `create-transit-gateway-policy-table` example creates a transit gateway policy table for the specified transit gateway.

```
aws ec2 create-transit-gateway-policy-table \
  --transit-gateway-id tgw-067f8505c18f0bd6e
```

Output:

```
{
  "TransitGatewayPolicyTable": {
    "TransitGatewayPolicyTableId": "tgw-ptb-0a16f134b78668a81",
    "TransitGatewayId": "tgw-067f8505c18f0bd6e",
    "State": "pending",
    "CreationTime": "2023-11-28T16:36:43+00:00"
  }
}
```

For more information, see [Transit gateway policy tables](#) in the *Transit Gateway User Guide*.

- For API details, see [CreateTransitGatewayPolicyTable](#) in *AWS CLI Command Reference*.

create-transit-gateway-prefix-list-reference

The following code example shows how to use `create-transit-gateway-prefix-list-reference`.

AWS CLI

To create a reference to a prefix list

The following `create-transit-gateway-prefix-list-reference` example creates a reference to the specified prefix list in the specified transit gateway route table.

```
aws ec2 create-transit-gateway-prefix-list-reference \  
  --transit-gateway-route-table-id tgw-rtb-0123456789abcd123 \  
  --prefix-list-id pl-111111222222223333 \  
  --transit-gateway-attachment-id tgw-attach-aaaaaabbbbb11111
```

Output:

```
{  
  "TransitGatewayPrefixListReference": {  
    "TransitGatewayRouteTableId": "tgw-rtb-0123456789abcd123",  
    "PrefixListId": "pl-111111222222223333",  
    "PrefixListOwnerId": "123456789012",  
    "State": "pending",  
    "Blackhole": false,  
    "TransitGatewayAttachment": {  
      "TransitGatewayAttachmentId": "tgw-attach-aaaaaabbbbb11111",  
      "ResourceType": "vpc",  
      "ResourceId": "vpc-112233445566aabbcc"  
    }  
  }  
}
```

For more information, see [Prefix list references](#) in the *Transit Gateways Guide*.

- For API details, see [CreateTransitGatewayPrefixListReference](#) in *AWS CLI Command Reference*.

create-transit-gateway-route-table

The following code example shows how to use `create-transit-gateway-route-table`.

AWS CLI

To create a Transit Gateway Route Table

The following `create-transit-gateway-route-table` example creates a route table for the specified transit gateway.

```
aws ec2 create-transit-gateway-route-table \  
  --transit-gateway-id tgw-0262a0e521EXAMPLE
```

Output:

```
{  
  "TransitGatewayRouteTable": {  
    "TransitGatewayRouteTableId": "tgw-rtb-0960981be7EXAMPLE",  
    "TransitGatewayId": "tgw-0262a0e521EXAMPLE",  
    "State": "pending",  
    "DefaultAssociationRouteTable": false,  
    "DefaultPropagationRouteTable": false,  
    "CreationTime": "2019-07-10T19:01:46.000Z"  
  }  
}
```

For more information, see [Create a transit gateway route table](#) in the *Transit Gateways Guide*.

- For API details, see [CreateTransitGatewayRouteTable](#) in *AWS CLI Command Reference*.

create-transit-gateway-route

The following code example shows how to use `create-transit-gateway-route`.

AWS CLI

To create a transit gateway route

The following `create-transit-gateway-route` example creates a route, with the specified destination, for the specified route table.

```
aws ec2 create-transit-gateway-route \  
  --destination-cidr-block 10.0.2.0/24 \  
  --transit-gateway-route-table-id tgw-rtb-0b6f6aaa01EXAMPLE \  
  --transit-gateway-attachment-id tgw-attach-0b5968d3b6EXAMPLE
```

Output:

```
{  
  "Route": {  
    "DestinationCidrBlock": "10.0.2.0/24",
```



```

    "TransitGatewayAttachments": [
      {
        "ResourceId": "vpc-0065acced4EXAMPLE",
        "TransitGatewayAttachmentId": "tgw-attach-0b5968d3b6EXAMPLE",
        "ResourceType": "vpc"
      }
    ],
    "Type": "static",
    "State": "active"
  }
}

```

For more information, see [Transit gateway route tables](#) in the *Transit Gateways Guide*.

- For API details, see [CreateTransitGatewayRoute](#) in *AWS CLI Command Reference*.

create-transit-gateway-vpc-attachment

The following code example shows how to use `create-transit-gateway-vpc-attachment`.

AWS CLI

Example 1: To associate a transit gateway with a VPC

The following `create-transit-gateway-vpc-attachment` example creates a transit gateway attachment to the specified VPC.

```

aws ec2 create-transit-gateway-vpc-attachment \
  --transit-gateway-id tgw-0262a0e521EXAMPLE \
  --vpc-id vpc-07e8ffd50f49335df \
  --subnet-id subnet-0752213d59EXAMPLE

```

Output:

```

{
  "TransitGatewayVpcAttachment": {
    "TransitGatewayAttachmentId": "tgw-attach-0a34fe6b4fEXAMPLE",
    "TransitGatewayId": "tgw-0262a0e521EXAMPLE",
    "VpcId": "vpc-07e8ffd50fEXAMPLE",
    "VpcOwnerId": "111122223333",
    "State": "pending",
    "SubnetIds": [

```

```

        "subnet-0752213d59EXAMPLE"
    ],
    "CreationTime": "2019-07-10T17:33:46.000Z",
    "Options": {
        "DnsSupport": "enable",
        "Ipv6Support": "disable"
    }
}
}

```

For more information, see [Create a transit gateway attachment to a VPC](#) in the *Transit Gateways Guide*.

Example 2: To associate a transit gateway with multiple subnets in a VPC

The following `create-transit-gateway-vpc-attachment` example creates a transit gateway attachment to the specified VPC and subnets.

```

aws ec2 create-transit-gateway-vpc-attachment \
  --transit-gateway-id tgw-02f776b1a7EXAMPLE \
  --vpc-id vpc-3EXAMPLE \
  --subnet-ids "subnet-dEXAMPLE" "subnet-6EXAMPLE"

```

Output:

```

{
  "TransitGatewayVpcAttachment": {
    "TransitGatewayAttachmentId": "tgw-attach-0e141e0bebEXAMPLE",
    "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",
    "VpcId": "vpc-3EXAMPLE",
    "VpcOwnerId": "111122223333",
    "State": "pending",
    "SubnetIds": [
      "subnet-6EXAMPLE",
      "subnet-dEXAMPLE"
    ],
    "CreationTime": "2019-12-17T20:07:52.000Z",
    "Options": {
      "DnsSupport": "enable",
      "Ipv6Support": "disable"
    }
  }
}

```

```
}

```

For more information, see [Create a transit gateway attachment to a VPC](#) in the *Transit Gateways Guide*.

- For API details, see [CreateTransitGatewayVpcAttachment](#) in *AWS CLI Command Reference*.

create-transit-gateway

The following code example shows how to use `create-transit-gateway`.

AWS CLI

To create a transit gateway

The following `create-transit-gateway` example creates a transit gateway.

```
aws ec2 create-transit-gateway \
  --description MyTGW \
  --options
  AmazonSideAsn=64516,AutoAcceptSharedAttachments=enable,DefaultRouteTableAssociation=enable,
```

Output:

```
{
  "TransitGateway": {
    "TransitGatewayId": "tgw-0262a0e521EXAMPLE",
    "TransitGatewayArn": "arn:aws:ec2:us-east-2:111122223333:transit-gateway/tgw-0262a0e521EXAMPLE",
    "State": "pending",
    "OwnerId": "111122223333",
    "Description": "MyTGW",
    "CreationTime": "2019-07-10T14:02:12.000Z",
    "Options": {
      "AmazonSideAsn": 64516,
      "AutoAcceptSharedAttachments": "enable",
      "DefaultRouteTableAssociation": "enable",
      "AssociationDefaultRouteTableId": "tgw-rtb-018774adf3EXAMPLE",
      "DefaultRouteTablePropagation": "enable",
      "PropagationDefaultRouteTableId": "tgw-rtb-018774adf3EXAMPLE",
      "VpnEcmpSupport": "enable",
      "DnsSupport": "enable"
    }
  }
}
```

```
}
}
```

For more information, see [Create a transit gateway](#) in the *Transit Gateways Guide*.

- For API details, see [CreateTransitGateway](#) in *AWS CLI Command Reference*.

create-verified-access-endpoint

The following code example shows how to use `create-verified-access-endpoint`.

AWS CLI

To create a Verified Access endpoint

The following `create-verified-access-endpoint` example creates a Verified Access endpoint for the specified Verified Access group. The specified network interface and security group must belong to the same VPC.

```
aws ec2 create-verified-access-endpoint \
  --verified-access-group-id vagr-0dbe967baf14b7235 \
  --endpoint-type network-interface \
  --attachment-type vpc \
  --domain-certificate-arn arn:aws:acm:us-east-2:123456789012:certificate/
eb065ea0-26f9-4e75-a6ce-0a1a7EXAMPLE \
  --application-domain example.com \
  --endpoint-domain-prefix my-ava-app \
  --security-group-ids sg-004915970c4c8f13a \
  --network-interface-options
NetworkInterfaceId=eni-0aec70418c8d87a0f,Protocol=https,Port=443 \
  --tag-specifications ResourceType=verified-access-
endpoint,Tags=[{Key=Name,Value=my-va-endpoint}]
```

Output:

```
{
  "VerifiedAccessEndpoint": {
    "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
    "VerifiedAccessGroupId": "vagr-0dbe967baf14b7235",
    "VerifiedAccessEndpointId": "vae-066fac616d4d546f2",
    "ApplicationDomain": "example.com",
    "EndpointType": "network-interface",
```

```

    "AttachmentType": "vpc",
    "DomainCertificateArn": "arn:aws:acm:us-east-2:123456789012:certificate/
eb065ea0-26f9-4e75-a6ce-0a1a7EXAMPLE",
    "EndpointDomain": "my-ava-
app.edge-00c3372d53b1540bb.vai-0ce000c0b7643abea.prod.verified-access.us-
east-2.amazonaws.com",
    "SecurityGroupIds": [
        "sg-004915970c4c8f13a"
    ],
    "NetworkInterfaceOptions": {
        "NetworkInterfaceId": "eni-0aec70418c8d87a0f",
        "Protocol": "https",
        "Port": 443
    },
    "Status": {
        "Code": "pending"
    },
    "Description": "",
    "CreationTime": "2023-08-25T20:54:43",
    "LastUpdatedTime": "2023-08-25T20:54:43",
    "Tags": [
        {
            "Key": "Name",
            "Value": "my-va-endpoint"
        }
    ]
}

```

For more information, see [Verified Access endpoints](#) in the *AWS Verified Access User Guide*.

- For API details, see [CreateVerifiedAccessEndpoint](#) in *AWS CLI Command Reference*.

create-verified-access-group

The following code example shows how to use create-verified-access-group.

AWS CLI

To create a Verified Access group

The following create-verified-access-group example creates a Verified Access group for the specified Verified Access instance.

```
aws ec2 create-verified-access-group \  
  --verified-access-instance-id vai-0ce000c0b7643abea \  
  --tag-specifications ResourceType=verified-access-  
group,Tags=[{Key=Name,Value=my-va-group}]
```

Output:

```
{  
  "VerifiedAccessGroup": {  
    "VerifiedAccessGroupId": "vagr-0dbe967baf14b7235",  
    "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",  
    "Description": "",  
    "Owner": "123456789012",  
    "VerifiedAccessGroupArn": "arn:aws:ec2:us-east-2:123456789012:verified-  
access-group/vagr-0dbe967baf14b7235",  
    "CreationTime": "2023-08-25T19:55:19",  
    "LastUpdatedTime": "2023-08-25T19:55:19",  
    "Tags": [  
      {  
        "Key": "Name",  
        "Value": "my-va-group"  
      }  
    ]  
  }  
}
```

For more information, see [Verified Access groups](#) in the *AWS Verified Access User Guide*.

- For API details, see [CreateVerifiedAccessGroup](#) in *AWS CLI Command Reference*.

create-verified-access-instance

The following code example shows how to use `create-verified-access-instance`.

AWS CLI

To create a Verified Access instance

The following `create-verified-access-instance` example creates a Verified Access instance with a Name tag.

```
aws ec2 create-verified-access-instance \  
  --verified-access-instance-id vai-0ce000c0b7643abea \  
  --tag-specifications ResourceType=verified-access-  
instance,Tags=[{Key=Name,Value=my-va-instance}]
```

```
--tag-specifications ResourceType=verified-access-
instance,Tags=[{Key=Name,Value=my-va-instance}]
```

Output:

```
{
  "VerifiedAccessInstance": {
    "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
    "Description": "",
    "VerifiedAccessTrustProviders": [],
    "CreationTime": "2023-08-25T18:27:56",
    "LastUpdatedTime": "2023-08-25T18:27:56",
    "Tags": [
      {
        "Key": "Name",
        "Value": "my-va-instance"
      }
    ]
  }
}
```

For more information, see [Verified Access instances](#) in the *AWS Verified Access User Guide*.

- For API details, see [CreateVerifiedAccessInstance](#) in *AWS CLI Command Reference*.

create-verified-access-trust-provider

The following code example shows how to use `create-verified-access-trust-provider`.

AWS CLI

To create a Verified Access trust provider

The following `create-verified-access-trust-provider` example sets up a Verified Access trust provider using AWS Identity Center.

```
aws ec2 create-verified-access-trust-provider \
  --trust-provider-type user \
  --user-trust-provider-type iam-identity-center \
  --policy-reference-name idc \
  --tag-specifications ResourceType=verified-access-trust-
provider,Tags=[{Key=Name,Value=my-va-trust-provider}]
```

Output:

```
{
  "VerifiedAccessTrustProvider": {
    "VerifiedAccessTrustProviderId": "vatp-0bb32de759a3e19e7",
    "Description": "",
    "TrustProviderType": "user",
    "UserTrustProviderType": "iam-identity-center",
    "PolicyReferenceName": "idc",
    "CreationTime": "2023-08-25T18:40:36",
    "LastUpdatedTime": "2023-08-25T18:40:36",
    "Tags": [
      {
        "Key": "Name",
        "Value": "my-va-trust-provider"
      }
    ]
  }
}
```

For more information, see [Trust providers for Verified Access](#) in the *AWS Verified Access User Guide*.

- For API details, see [CreateVerifiedAccessTrustProvider](#) in *AWS CLI Command Reference*.

create-volume

The following code example shows how to use `create-volume`.

AWS CLI**To create an empty General Purpose SSD (gp2) volume**

The following `create-volume` example creates an 80 GiB General Purpose SSD (gp2) volume in the specified Availability Zone. Note that the current Region must be `us-east-1`, or you can add the `--region` parameter to specify the Region for the command.

```
aws ec2 create-volume \
  --volume-type gp2 \
  --size 80 \
  --availability-zone us-east-1a
```


Output:

```
{
  "AvailabilityZone": "us-east-1a",
  "Tags": [],
  "Encrypted": false,
  "VolumeType": "gp2",
  "VolumeId": "vol-1234567890abcdef0",
  "State": "creating",
  "Iops": 240,
  "SnapshotId": "",
  "CreateTime": "YYYY-MM-DDTHH:MM:SS.000Z",
  "Size": 80
}
```

If you do not specify a volume type, the default volume type is gp2.

```
aws ec2 create-volume \
  --size 80 \
  --availability-zone us-east-1a
```

Example 2: To create a Provisioned IOPS SSD (io1) volume from a snapshot

The following `create-volume` example creates a Provisioned IOPS SSD (io1) volume with 1000 provisioned IOPS in the specified Availability Zone using the specified snapshot.

```
aws ec2 create-volume \
  --volume-type io1 \
  --iops 1000 \
  --snapshot-id snap-066877671789bd71b \
  --availability-zone us-east-1a
```

Output:

```
{
  "AvailabilityZone": "us-east-1a",
  "Tags": [],
  "Encrypted": false,
  "VolumeType": "io1",
  "VolumeId": "vol-1234567890abcdef0",
  "State": "creating",
}
```

```
"Iops": 1000,  
"SnapshotId": "snap-066877671789bd71b",  
"CreateTime": "YYYY-MM-DDTHH:MM:SS.000Z",  
"Size": 500  
}
```

Example 3: To create an encrypted volume

The following `create-volume` example creates an encrypted volume using the default CMK for EBS encryption. If encryption by default is disabled, you must specify the `--encrypted` parameter as follows.

```
aws ec2 create-volume \  
  --size 80 \  
  --encrypted \  
  --availability-zone us-east-1a
```

Output:

```
{  
  "AvailabilityZone": "us-east-1a",  
  "Tags": [],  
  "Encrypted": true,  
  "VolumeType": "gp2",  
  "VolumeId": "vol-1234567890abcdef0",  
  "State": "creating",  
  "Iops": 240,  
  "SnapshotId": "",  
  "CreateTime": "YYYY-MM-DDTHH:MM:SS.000Z",  
  "Size": 80  
}
```

If encryption by default is enabled, the following example command creates an encrypted volume, even without the `--encrypted` parameter.

```
aws ec2 create-volume \  
  --size 80 \  
  --availability-zone us-east-1a
```

If you use the `--kms-key-id` parameter to specify a customer managed CMK, you must specify the `--encrypted` parameter even if encryption by default is enabled.

```
aws ec2 create-volume \  
  --volume-type gp2 \  
  --size 80 \  
  --encrypted \  
  --kms-key-id 0ea3fef3-80a7-4778-9d8c-1c0c6EXAMPLE \  
  --availability-zone us-east-1a
```

Example 4: To create a volume with tags

The following `create-volume` example creates a volume and adds two tags.

```
aws ec2 create-volume \  
  --availability-zone us-east-1a \  
  --volume-type gp2 \  
  --size 80 \  
  --tag-specifications 'ResourceType=volume,Tags=[{Key=purpose,Value=production},  
{Key=cost-center,Value=cc123}]'
```

- For API details, see [CreateVolume](#) in *AWS CLI Command Reference*.

create-vpc-endpoint-connection-notification

The following code example shows how to use `create-vpc-endpoint-connection-notification`.

AWS CLI

To create an endpoint connection notification

This example creates a notification for a specific endpoint service that alerts you when interface endpoints have connected to your service and when endpoints have been accepted for your service.

Command:

```
aws ec2 create-vpc-endpoint-connection-notification --connection-notification-arn  
arn:aws:sns:us-east-2:123456789012:VpceNotification --connection-events Connect  
Accept --service-id vpce-svc-1237881c0d25a3abc
```

Output:

```
{
  "ConnectionNotification": {
    "ConnectionNotificationState": "Enabled",
    "ConnectionNotificationType": "Topic",
    "ServiceId": "vpce-svc-1237881c0d25a3abc",
    "ConnectionEvents": [
      "Accept",
      "Connect"
    ],
    "ConnectionNotificationId": "vpce-nfn-008776de7e03f5abc",
    "ConnectionNotificationArn": "arn:aws:sns:us-
east-2:123456789012:VpceNotification"
  }
}
```

- For API details, see [CreateVpcEndpointConnectionNotification](#) in *AWS CLI Command Reference*.

create-vpc-endpoint-service-configuration

The following code example shows how to use `create-vpc-endpoint-service-configuration`.

AWS CLI

Example 1: To create an endpoint service configuration for an interface endpoint

The following `create-vpc-endpoint-service-configuration` example creates a VPC endpoint service configuration using the Network Load Balancer `nlb-vpce`. This example also specifies that requests to connect to the service through an interface endpoint must be accepted.

```
aws ec2 create-vpc-endpoint-service-configuration \
  --network-load-balancer-arns arn:aws:elasticloadbalancing:us-
east-1:123456789012:loadbalancer/net/nlb-vpce/e94221227f1ba532 \
  --acceptance-required
```

Output:

```
{
  "ServiceConfiguration": {
```

```

    "ServiceType": [
      {
        "ServiceType": "Interface"
      }
    ],
    "NetworkLoadBalancerArns": [
      "arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/net/
nlb-vpce/e94221227f1ba532"
    ],
    "ServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-03d5ebb7d9579a2b3",
    "ServiceState": "Available",
    "ServiceId": "vpce-svc-03d5ebb7d9579a2b3",
    "AcceptanceRequired": true,
    "AvailabilityZones": [
      "us-east-1d"
    ],
    "BaseEndpointDnsNames": [
      "vpce-svc-03d5ebb7d9579a2b3.us-east-1.vpce.amazonaws.com"
    ]
  }
}

```

Example 2: To create an endpoint service configuration for a Gateway Load Balancer endpoint

The following `create-vpc-endpoint-service-configuration` example creates a VPC endpoint service configuration using the Gateway Load Balancer `GWLBService`. Requests to connect to the service through a Gateway Load Balancer endpoint are automatically accepted.

```

aws ec2 create-vpc-endpoint-service-configuration \
  --gateway-load-balancer-arns arn:aws:elasticloadbalancing:us-
east-1:123456789012:loadbalancer/gwy/GWLBService/123123123123abcc \
  --no-acceptance-required

```

Output:

```

{
  "ServiceConfiguration": {
    "ServiceType": [
      {
        "ServiceType": "GatewayLoadBalancer"
      }
    ]
  }
}

```

```

    ],
    "ServiceId": "vpce-svc-123123a1c43abc123",
    "ServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-123123a1c43abc123",
    "ServiceState": "Available",
    "AvailabilityZones": [
      "us-east-1d"
    ],
    "AcceptanceRequired": false,
    "ManagesVpcEndpoints": false,
    "GatewayLoadBalancerArns": [
      "arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/gwy/
      GWLBService/123123123123abcc"
    ]
  }
}

```

For more information, see [VPC endpoint services](#) in the *Amazon VPC User Guide*.

- For API details, see [CreateVpcEndpointServiceConfiguration](#) in *AWS CLI Command Reference*.

create-vpc-endpoint

The following code example shows how to use `create-vpc-endpoint`.

AWS CLI

Example 1: To create a gateway endpoint

The following `create-vpc-endpoint` example creates a gateway VPC endpoint between VPC `vpc-1a2b3c4d` and Amazon S3 in the `us-east-1` region, and associates route table `rtb-11aa22bb` with the endpoint.

```

aws ec2 create-vpc-endpoint \
  --vpc-id vpc-1a2b3c4d \
  --service-name com.amazonaws.us-east-1.s3 \
  --route-table-ids rtb-11aa22bb

```

Output:

```

{
  "VpcEndpoint": {
    "PolicyDocument": "{\"Version\":\"2008-10-17\",\"Statement\":[{\"Sid\":\"\",
    \"Effect\":\"Allow\",\"Principal\":\"*\",\"Action\":\"*\",\"Resource\":\"*\"}]}",

```

```

    "VpcId": "vpc-1a2b3c4d",
    "State": "available",
    "ServiceName": "com.amazonaws.us-east-1.s3",
    "RouteTableIds": [
      "rtb-11aa22bb"
    ],
    "VpcEndpointId": "vpc-1a2b3c4d",
    "CreationTimestamp": "2015-05-15T09:40:50Z"
  }
}

```

For more information, see [Creating a gateway endpoint](#) in the *AWSPrivateLink Guide*.

Example 2: To create an interface endpoint

The following `create-vpc-endpoint` example creates an interface VPC endpoint between VPC `vpc-1a2b3c4d` and Amazon S3 in the `us-east-1` region. The command creates the endpoint in subnet `subnet-1a2b3c4d`, associates it with security group `sg-1a2b3c4d`, and adds a tag with a key of "Service" and a Value of "S3".

```

aws ec2 create-vpc-endpoint \
  --vpc-id vpc-1a2b3c4d \
  --vpc-endpoint-type Interface \
  --service-name com.amazonaws.us-east-1.s3 \
  --subnet-ids subnet-7b16de0c \
  --security-group-id sg-1a2b3c4d \
  --tag-specifications ResourceType=vpc-endpoint,Tags=[{Key=service,Value=S3}]

```

Output:

```

{
  "VpcEndpoint": {
    "VpcEndpointId": "vpce-1a2b3c4d5e6f1a2b3",
    "VpcEndpointType": "Interface",
    "VpcId": "vpc-1a2b3c4d",
    "ServiceName": "com.amazonaws.us-east-1.s3",
    "State": "pending",
    "RouteTableIds": [],
    "SubnetIds": [
      "subnet-1a2b3c4d"
    ],
    "Groups": [

```

```

        {
            "GroupId": "sg-1a2b3c4d",
            "GroupName": "default"
        }
    ],
    "PrivateDnsEnabled": false,
    "RequesterManaged": false,
    "NetworkInterfaceIds": [
        "eni-0b16f0581c8ac6877"
    ],
    "DnsEntries": [
        {
            "DnsName": "*.vpce-1a2b3c4d5e6f1a2b3-9hnenorg.s3.us-
east-1.vpce.amazonaws.com",
            "HostedZoneId": "Z7HUB22UULQXV"
        },
        {
            "DnsName": "*.vpce-1a2b3c4d5e6f1a2b3-9hnenorg-us-east-1c.s3.us-
east-1.vpce.amazonaws.com",
            "HostedZoneId": "Z7HUB22UULQXV"
        }
    ],
    "CreationTimestamp": "2021-03-05T14:46:16.030000+00:00",
    "Tags": [
        {
            "Key": "service",
            "Value": "S3"
        }
    ],
    "OwnerId": "123456789012"
}
}

```

For more information, see [Creating an interface endpoint](#) in the *User Guide for AWSPrivateLink*.

Example 3: To create a Gateway Load Balancer endpoint

The following `create-vpc-endpoint` example creates a Gateway Load Balancer endpoint between VPC `vpc-111122223333aabbcc` and a service that is configured using a Gateway Load Balancer.

```

aws ec2 create-vpc-endpoint \
    --service-name com.amazonaws.vpce.us-east-1.vpce-svc-123123a1c43abc123 \

```



```
--vpc-endpoint-type GatewayLoadBalancer \  
--vpc-id vpc-111122223333aabbcc \  
--subnet-ids subnet-0011aabbcc2233445
```

Output:

```
{  
  "VpcEndpoint": {  
    "VpcEndpointId": "vpce-aabbaabbaabbaabba",  
    "VpcEndpointType": "GatewayLoadBalancer",  
    "VpcId": "vpc-111122223333aabbcc",  
    "ServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-123123a1c43abc123",  
    "State": "pending",  
    "SubnetIds": [  
      "subnet-0011aabbcc2233445"  
    ],  
    "RequesterManaged": false,  
    "NetworkInterfaceIds": [  
      "eni-01010120203030405"  
    ],  
    "CreationTimestamp": "2020-11-11T08:06:03.522Z",  
    "OwnerId": "123456789012"  
  }  
}
```

For more information, see [Gateway Load Balancer endpoints](#) in the *User Guide for AWS PrivateLink*.

- For API details, see [CreateVpcEndpoint](#) in *AWS CLI Command Reference*.

create-vpc-peering-connection

The following code example shows how to use `create-vpc-peering-connection`.

AWS CLI

To create a VPC peering connection between your VPCs

This example requests a peering connection between your VPCs `vpc-1a2b3c4d` and `vpc-11122233`.

Command:

```
aws ec2 create-vpc-peering-connection --vpc-id vpc-1a2b3c4d --peer-vpc-id
vpc-11122233
```

Output:

```
{
  "VpcPeeringConnection": {
    "Status": {
      "Message": "Initiating Request to 444455556666",
      "Code": "initiating-request"
    },
    "Tags": [],
    "RequesterVpcInfo": {
      "OwnerId": "444455556666",
      "VpcId": "vpc-1a2b3c4d",
      "CidrBlock": "10.0.0.0/28"
    },
    "VpcPeeringConnectionId": "pcx-111aaa111",
    "ExpirationTime": "2014-04-02T16:13:36.000Z",
    "AccepterVpcInfo": {
      "OwnerId": "444455556666",
      "VpcId": "vpc-11122233"
    }
  }
}
```

To create a VPC peering connection with a VPC in another account

This example requests a peering connection between your VPC (vpc-1a2b3c4d), and a VPC (vpc-11122233) that belongs AWS account 123456789012.

Command:

```
aws ec2 create-vpc-peering-connection --vpc-id vpc-1a2b3c4d --peer-vpc-id
vpc-11122233 --peer-owner-id 123456789012
```

To create a VPC peering connection with a VPC in a different region

This example requests a peering connection between your VPC in the current region (vpc-1a2b3c4d), and a VPC (vpc-11122233) in your account in the us-west-2 region.

Command:

```
aws ec2 create-vpc-peering-connection --vpc-id vpc-1a2b3c4d --peer-vpc-id
vpc-11122233 --peer-region us-west-2
```

This example requests a peering connection between your VPC in the current region (vpc-1a2b3c4d), and a VPC (vpc-11122233) that belongs AWS account 123456789012 that's in the us-west-2 region.

Command:

```
aws ec2 create-vpc-peering-connection --vpc-id vpc-1a2b3c4d --peer-vpc-id
vpc-11122233 --peer-owner-id 123456789012 --peer-region us-west-2
```

- For API details, see [CreateVpcPeeringConnection](#) in *AWS CLI Command Reference*.

create-vpc

The following code example shows how to use create-vpc.

AWS CLI

Example 1: To create a VPC

The following create-vpc example creates a VPC with the specified IPv4 CIDR block and a Name tag.

```
aws ec2 create-vpc \
  --cidr-block 10.0.0.0/16 \
  --tag-specification ResourceType=vpc,Tags=[{Key=Name,Value=MyVpc}]
```

Output:

```
{
  "Vpc": {
    "CidrBlock": "10.0.0.0/16",
    "DhcpOptionsId": "dopt-5EXAMPLE",
    "State": "pending",
    "VpcId": "vpc-0a60eb65b4EXAMPLE",
    "OwnerId": "123456789012",
    "InstanceTenancy": "default",
    "Ipv6CidrBlockAssociationSet": [],
```

```

    "CidrBlockAssociationSet": [
      {
        "AssociationId": "vpc-cidr-assoc-07501b79ecEXAMPLE",
        "CidrBlock": "10.0.0.0/16",
        "CidrBlockState": {
          "State": "associated"
        }
      }
    ],
    "IsDefault": false,
    "Tags": [
      {
        "Key": "Name",
        "Value": "MyVpc"
      }
    ]
  }
}

```

Example 2: To create a VPC with dedicated tenancy

The following `create-vpc` example creates a VPC with the specified IPv4 CIDR block and dedicated tenancy.

```

aws ec2 create-vpc \
  --cidr-block 10.0.0.0/16 \
  --instance-tenancy dedicated

```

Output:

```

{
  "Vpc": {
    "CidrBlock": "10.0.0.0/16",
    "DhcpOptionsId": "dopt-19edf471",
    "State": "pending",
    "VpcId": "vpc-0a53287fa4EXAMPLE",
    "OwnerId": "111122223333",
    "InstanceTenancy": "dedicated",
    "Ipv6CidrBlockAssociationSet": [],
    "CidrBlockAssociationSet": [
      {
        "AssociationId": "vpc-cidr-assoc-00b24cc1c2EXAMPLE",
        "CidrBlock": "10.0.0.0/16",

```

```

        "CidrBlockState": {
            "State": "associated"
        }
    },
    "IsDefault": false
}

```

Example 3: To create a VPC with an IPv6 CIDR block

The following `create-vpc` example creates a VPC with an Amazon-provided IPv6 CIDR block.

```

aws ec2 create-vpc \
  --cidr-block 10.0.0.0/16 \
  --amazon-provided-ipv6-cidr-block

```

Output:

```

{
  "Vpc": {
    "CidrBlock": "10.0.0.0/16",
    "DhcpOptionsId": "dopt-dEXAMPLE",
    "State": "pending",
    "VpcId": "vpc-0fc5e3406bEXAMPLE",
    "OwnerId": "123456789012",
    "InstanceTenancy": "default",
    "Ipv6CidrBlockAssociationSet": [
      {
        "AssociationId": "vpc-cidr-assoc-068432c60bEXAMPLE",
        "Ipv6CidrBlock": "",
        "Ipv6CidrBlockState": {
          "State": "associating"
        }
      },
      {
        "Ipv6Pool": "Amazon",
        "NetworkBorderGroup": "us-west-2"
      }
    ],
    "CidrBlockAssociationSet": [
      {
        "AssociationId": "vpc-cidr-assoc-0669f8f9f5EXAMPLE",
        "CidrBlock": "10.0.0.0/16",
        "CidrBlockState": {

```

```

        "State": "associated"
      }
    }
  ],
  "IsDefault": false
}
}

```

Example 4: To create a VPC with a CIDR from an IPAM pool

The following `create-vpc` example creates a VPC with a CIDR from an Amazon VPC IP Address Manager (IPAM) pool.

Linux and macOS:

```

aws ec2 create-vpc \
  --ipv4-ipam-pool-id ipam-pool-0533048da7d823723 \
  --tag-specifications ResourceType=vpc,Tags='[{"Key=Environment,Value="Preprod"}, {"Key=Owner,Value="Build Team"}]'

```

Windows:

```

aws ec2 create-vpc ^
  --ipv4-ipam-pool-id ipam-pool-0533048da7d823723 ^
  --tag-specifications ResourceType=vpc,Tags=[{"Key=Environment,Value="Preprod"}, {"Key=Owner,Value="Build Team"}]

```

Output:

```

{
  "Vpc": {
    "CidrBlock": "10.0.1.0/24",
    "DhcpOptionsId": "dopt-2afccf50",
    "State": "pending",
    "VpcId": "vpc-010e1791024eb0af9",
    "OwnerId": "123456789012",
    "InstanceTenancy": "default",
    "Ipv6CidrBlockAssociationSet": [],
    "CidrBlockAssociationSet": [
      {
        "AssociationId": "vpc-cidr-assoc-0a77de1d803226d4b",

```

```
        "CidrBlock": "10.0.1.0/24",
        "CidrBlockState": {
            "State": "associated"
        }
    },
    "IsDefault": false,
    "Tags": [
        {
            "Key": "Environment",
            "Value": "Preprod"
        },
        {
            "Key": "Owner",
            "Value": "Build Team"
        }
    ]
}
```

For more information, see [Create a VPC that uses an IPAM pool CIDR](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [CreateVpc](#) in *AWS CLI Command Reference*.

create-vpn-connection-route

The following code example shows how to use `create-vpn-connection-route`.

AWS CLI

To create a static route for a VPN connection

This example creates a static route for the specified VPN connection. If the command succeeds, no output is returned.

Command:

```
aws ec2 create-vpn-connection-route --vpn-connection-id vpn-40f41529 --destination-
cidr-block 11.12.0.0/16
```

- For API details, see [CreateVpnConnectionRoute](#) in *AWS CLI Command Reference*.

create-vpn-connection

The following code example shows how to use `create-vpn-connection`.

AWS CLI

Example 1: To create a VPN connection with dynamic routing

The following `create-vpn-connection` example creates a VPN connection between the specified virtual private gateway and the specified customer gateway, and applies tags to the VPN connection. The output includes the configuration information for your customer gateway device, in XML format.

```
aws ec2 create-vpn-connection \  
  --type ipsec.1 \  
  --customer-gateway-id cgw-001122334455aabbc \  
  --vpn-gateway-id vgw-1a1a1a1a1a1a2b2b2 \  
  --tag-specification 'ResourceType=vpn-connection,Tags=[{Key=Name,Value=BGP-  
VPN}]'
```

Output:

```
{  
  "VpnConnection": {  
    "CustomerGatewayConfiguration": "...configuration information...",  
    "CustomerGatewayId": "cgw-001122334455aabbc",  
    "Category": "VPN",  
    "State": "pending",  
    "VpnConnectionId": "vpn-123123123123abcab",  
    "VpnGatewayId": "vgw-1a1a1a1a1a1a2b2b2",  
    "Options": {  
      "EnableAcceleration": false,  
      "StaticRoutesOnly": false,  
      "LocalIpv4NetworkCidr": "0.0.0.0/0",  
      "RemoteIpv4NetworkCidr": "0.0.0.0/0",  
      "TunnelInsideIpVersion": "ipv4",  
      "TunnelOptions": [  
        {},  
        {}  
      ]  
    },  
    "Routes": [],  
    "Tags": [  

```



```

    {
      "Key": "Name",
      "Value": "BGP-VPN"
    }
  ]
}

```

For more information, see [How AWS Site-to-Site VPN works](#) in the *AWS Site-to-Site VPN User Guide*.

Example 2: To create a VPN connection with static routing

The following `create-vpn-connection` example creates a VPN connection between the specified virtual private gateway and the specified customer gateway. The options specify static routing. The output includes the configuration information for your customer gateway device, in XML format.

```

aws ec2 create-vpn-connection \
  --type ipsec.1 \
  --customer-gateway-id cgw-001122334455aabbc \
  --vpn-gateway-id vgw-1a1a1a1a1a1a2b2b2 \
  --options "{\"StaticRoutesOnly\":true}"

```

Output:

```

{
  "VpnConnection": {
    "CustomerGatewayConfiguration": "..configuration information...",
    "CustomerGatewayId": "cgw-001122334455aabbc",
    "Category": "VPN",
    "State": "pending",
    "VpnConnectionId": "vpn-123123123123abcab",
    "VpnGatewayId": "vgw-1a1a1a1a1a1a2b2b2",
    "Options": {
      "EnableAcceleration": false,
      "StaticRoutesOnly": true,
      "LocalIpv4NetworkCidr": "0.0.0.0/0",
      "RemoteIpv4NetworkCidr": "0.0.0.0/0",
      "TunnelInsideIpVersion": "ipv4",
      "TunnelOptions": [
        {}
      ]
    }
  }
}

```

```

        {}
      ]
    },
    "Routes": [],
    "Tags": []
  }
}

```

For more information, see [How AWS Site-to-Site VPN works](#) in the *AWS Site-to-Site VPN User Guide*.

Example 3: To create a VPN connection and specify your own inside CIDR and pre-shared key

The following `create-vpn-connection` example creates a VPN connection and specifies the inside IP address CIDR block and a custom pre-shared key for each tunnel. The specified values are returned in the `CustomerGatewayConfiguration` information.

```

aws ec2 create-vpn-connection \
  --type ipsec.1 \
  --customer-gateway-id cgw-001122334455aabbcc \
  --vpn-gateway-id vgw-1a1a1a1a1a1a2b2b2 \
  --options
  TunnelOptions='[{"TunnelInsideCidr": "169.254.12.0/30", "PreSharedKey": "ExamplePreSharedKey1"},
  {"TunnelInsideCidr": "169.254.13.0/30", "PreSharedKey": "ExamplePreSharedKey2"}]'

```

Output:

```

{
  "VpnConnection": {
    "CustomerGatewayConfiguration": "..configuration information...",
    "CustomerGatewayId": "cgw-001122334455aabbcc",
    "Category": "VPN",
    "State": "pending",
    "VpnConnectionId": "vpn-123123123123abcab",
    "VpnGatewayId": "vgw-1a1a1a1a1a1a2b2b2",
    "Options": {
      "EnableAcceleration": false,
      "StaticRoutesOnly": false,
      "LocalIpv4NetworkCidr": "0.0.0.0/0",
      "RemoteIpv4NetworkCidr": "0.0.0.0/0",
      "TunnelInsideIpVersion": "ipv4",
      "TunnelOptions": [

```

```

        {
            "OutsideIpAddress": "203.0.113.3",
            "TunnelInsideCidr": "169.254.12.0/30",
            "PreSharedKey": "ExamplePreSharedKey1"
        },
        {
            "OutsideIpAddress": "203.0.113.5",
            "TunnelInsideCidr": "169.254.13.0/30",
            "PreSharedKey": "ExamplePreSharedKey2"
        }
    ]
},
"Routes": [],
"Tags": []
}
}

```

For more information, see [How AWS Site-to-Site VPN works](#) in the *AWS Site-to-Site VPN User Guide*.

Example 4: To create a VPN connection that supports IPv6 traffic

The following `create-vpn-connection` example creates a VPN connection that supports IPv6 traffic between the specified transit gateway and specified customer gateway. The tunnel options for both tunnels specify that AWS must initiate the IKE negotiation.

```

aws ec2 create-vpn-connection \
  --type ipsec.1 \
  --transit-gateway-id tgw-12312312312312312 \
  --customer-gateway-id cgw-001122334455aabbcc \
  --options TunnelInsideIpVersion=ipv6,TunnelOptions=[{StartupAction=start},
  {StartupAction=start}]

```

Output:

```

{
  "VpnConnection": {
    "CustomerGatewayConfiguration": "..configuration information...",
    "CustomerGatewayId": "cgw-001122334455aabbcc",
    "Category": "VPN",
    "State": "pending",
    "VpnConnectionId": "vpn-111111111122222222",

```

```
"TransitGatewayId": "tgw-12312312312312312",
"Options": {
  "EnableAcceleration": false,
  "StaticRoutesOnly": false,
  "LocalIpv6NetworkCidr": "::/0",
  "RemoteIpv6NetworkCidr": "::/0",
  "TunnelInsideIpVersion": "ipv6",
  "TunnelOptions": [
    {
      "OutsideIpAddress": "203.0.113.3",
      "StartupAction": "start"
    },
    {
      "OutsideIpAddress": "203.0.113.5",
      "StartupAction": "start"
    }
  ]
},
"Routes": [],
"Tags": []
}
}
```

For more information, see [How AWS Site-to-Site VPN works](#) in the *AWS Site-to-Site VPN User Guide*.

- For API details, see [CreateVpnConnection](#) in *AWS CLI Command Reference*.

create-vpn-gateway

The following code example shows how to use `create-vpn-gateway`.

AWS CLI

To create a virtual private gateway

This example creates a virtual private gateway.

Command:

```
aws ec2 create-vpn-gateway --type ipsec.1
```

Output:

```
{
  "VpnGateway": {
    "AmazonSideAsn": 64512,
    "State": "available",
    "Type": "ipsec.1",
    "VpnGatewayId": "vgw-9a4cacf3",
    "VpcAttachments": []
  }
}
```

To create a virtual private gateway with a specific Amazon-side ASN

This example creates a virtual private gateway and specifies the Autonomous System Number (ASN) for the Amazon side of the BGP session.

Command:

```
aws ec2 create-vpn-gateway --type ipsec.1 --amazon-side-asn 65001
```

Output:

```
{
  "VpnGateway": {
    "AmazonSideAsn": 65001,
    "State": "available",
    "Type": "ipsec.1",
    "VpnGatewayId": "vgw-9a4cacf3",
    "VpcAttachments": []
  }
}
```

- For API details, see [CreateVpnGateway](#) in *AWS CLI Command Reference*.

delete-carrier-gateway

The following code example shows how to use `delete-carrier-gateway`.

AWS CLI

To delete your carrier gateway

The following `delete-carrier-gateway` example deletes the specified carrier gateway.

```
aws ec2 delete-carrier-gateway \  
  --carrier-gateway-id cagw-0465cdEXAMPLE1111
```

Output:

```
{  
  "CarrierGateway": {  
    "CarrierGatewayId": "cagw-0465cdEXAMPLE1111",  
    "VpcId": "vpc-0c529aEXAMPLE1111",  
    "State": "deleting",  
    "OwnerId": "123456789012"  
  }  
}
```

For more information, see [Carrier gateways](#) in the *Amazon Virtual Private Cloud User Guide*.

- For API details, see [DeleteCarrierGateway](#) in *AWS CLI Command Reference*.

`delete-client-vpn-endpoint`

The following code example shows how to use `delete-client-vpn-endpoint`.

AWS CLI

To delete a Client VPN endpoint

The following `delete-client-vpn-endpoint` example deletes the specified Client VPN endpoint.

```
aws ec2 delete-client-vpn-endpoint \  
  --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde
```

Output:

```
{  
  "Status": {  
    "Code": "deleting"  
  }  
}
```

For more information, see [Client VPN Endpoints](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [DeleteClientVpnEndpoint](#) in *AWS CLI Command Reference*.

delete-client-vpn-route

The following code example shows how to use `delete-client-vpn-route`.

AWS CLI

To delete a route for a Client VPN endpoint

The following `delete-client-vpn-route` example deletes the `0.0.0.0/0` route for the specified subnet of a Client VPN endpoint.

```
aws ec2 delete-client-vpn-route \
  --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde \
  --destination-cidr-block 0.0.0.0/0 \
  --target-vpc-subnet-id subnet-0123456789abcabca
```

Output:

```
{
  "Status": {
    "Code": "deleting"
  }
}
```

For more information, see [Routes](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [DeleteClientVpnRoute](#) in *AWS CLI Command Reference*.

delete-coip-cidr

The following code example shows how to use `delete-coip-cidr`.

AWS CLI

To delete a range of customer-owned IP (CoIP) addresses

The following `delete-coip-cidr` example deletes the specified range of CoIP addresses in the specified CoIP pool.

```
aws ec2 delete-coip-cidr \  
  --cidr 14.0.0.0/24 \  
  --coip-pool-id ipv4pool-coip-1234567890abcdefg
```

Output:

```
{  
  "CoipCidr": {  
    "Cidr": "14.0.0.0/24",  
    "CoipPoolId": "ipv4pool-coip-1234567890abcdefg",  
    "LocalGatewayRouteTableId": "lgw-rtb-abcdefg1234567890"  
  }  
}
```

For more information, see [Customer-owned IP addresses](#) in the *AWS Outposts User Guide*.

- For API details, see [DeleteCoipCidr](#) in *AWS CLI Command Reference*.

delete-coip-pool

The following code example shows how to use `delete-coip-pool`.

AWS CLI

To delete a pool of customer-owned IP (CoIP) addresses

The following `delete-coip-pool` example deletes a CoIP pool of CoIP addresses.

```
aws ec2 delete-coip-pool \  
  --coip-pool-id ipv4pool-coip-1234567890abcdefg
```

Output:

```
{  
  "CoipPool": {  
    "PoolId": "ipv4pool-coip-1234567890abcdefg",  
    "LocalGatewayRouteTableId": "lgw-rtb-abcdefg1234567890",  
    "PoolArn": "arn:aws:ec2:us-west-2:123456789012:coip-pool/ipv4pool-coip-1234567890abcdefg"  
  }  
}
```


For more information, see [Customer-owned IP addresses](#) in the *AWS Outposts User Guide*.

- For API details, see [DeleteCoipPool](#) in *AWS CLI Command Reference*.

delete-customer-gateway

The following code example shows how to use `delete-customer-gateway`.

AWS CLI

To delete a customer gateway

This example deletes the specified customer gateway. If the command succeeds, no output is returned.

Command:

```
aws ec2 delete-customer-gateway --customer-gateway-id cgw-0e11f167
```

- For API details, see [DeleteCustomerGateway](#) in *AWS CLI Command Reference*.

delete-dhcp-options

The following code example shows how to use `delete-dhcp-options`.

AWS CLI

To delete a DHCP options set

This example deletes the specified DHCP options set. If the command succeeds, no output is returned.

Command:

```
aws ec2 delete-dhcp-options --dhcp-options-id dopt-d9070ebb
```

- For API details, see [DeleteDhcpOptions](#) in *AWS CLI Command Reference*.

delete-egress-only-internet-gateway

The following code example shows how to use `delete-egress-only-internet-gateway`.

AWS CLI

To delete an egress-only Internet gateway

This example deletes the specified egress-only Internet gateway.

Command:

```
aws ec2 delete-egress-only-internet-gateway --egress-only-internet-gateway-id
eigw-01eadbd45ecd7943f
```

Output:

```
{
  "ReturnCode": true
}
```

- For API details, see [DeleteEgressOnlyInternetGateway](#) in *AWS CLI Command Reference*.

delete-fleets

The following code example shows how to use delete-fleets.

AWS CLI

Example 1: To delete an EC2 Fleet and terminate the associated instances

The following delete-fleets example deletes the specified EC2 Fleet and terminates the associated On-Demand Instances and Spot Instances.

```
aws ec2 delete-fleets \
  --fleet-ids fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE \
  --terminate-instances
```

Output:

```
{
  "SuccessfulFleetDeletions": [
    {
      "CurrentFleetState": "deleted_terminating",
      "PreviousFleetState": "active",
      "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE"
    }
  ]
}
```

```

    }
  ],
  "UnsuccessfulFleetDeletions": []
}

```

For more information, see [Delete an EC2 Fleet](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

Example 2: To delete an EC2 Fleet without terminating the associated instances

The following `delete-fleets` example deletes the specified EC2 Fleet without terminating the associated On-Demand Instances and Spot Instances.

```

aws ec2 delete-fleets \
  --fleet-ids fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE \
  --no-terminate-instances

```

Output:

```

{
  "SuccessfulFleetDeletions": [
    {
      "CurrentFleetState": "deleted_running",
      "PreviousFleetState": "active",
      "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE"
    }
  ],
  "UnsuccessfulFleetDeletions": []
}

```

For more information, see [Delete an EC2 Fleet](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [DeleteFleets](#) in *AWS CLI Command Reference*.

delete-flow-logs

The following code example shows how to use `delete-flow-logs`.

AWS CLI

To delete a flow log

The following `delete-flow-logs` example deletes the specified flow log.

```
aws ec2 delete-flow-logs --flow-log-id fl-11223344556677889
```

Output:

```
{
  "Unsuccessful": []
}
```

- For API details, see [DeleteFlowLogs](#) in *AWS CLI Command Reference*.

delete-fpga-image

The following code example shows how to use `delete-fpga-image`.

AWS CLI

To delete an Amazon FPGA image

This example deletes the specified AFI.

Command:

```
aws ec2 delete-fpga-image --fpga-image-id afi-06b12350a123fbabc
```

Output:

```
{
  "Return": true
}
```

- For API details, see [DeleteFpgaImage](#) in *AWS CLI Command Reference*.

delete-instance-connect-endpoint

The following code example shows how to use `delete-instance-connect-endpoint`.

AWS CLI

To delete an EC2 Instance Connect Endpoint

The following `delete-instance-connect-endpoint` example deletes the specified EC2 Instance Connect Endpoint.

```
aws ec2 delete-instance-connect-endpoint \  
  --instance-connect-endpoint-id eice-03f5e49b83924bbc7
```

Output:

```
{  
  "InstanceConnectEndpoint": {  
    "OwnerId": "111111111111",  
    "InstanceConnectEndpointId": "eice-0123456789example",  
    "InstanceConnectEndpointArn": "arn:aws:ec2:us-east-1:111111111111:instance-  
connect-endpoint/eice-0123456789example",  
    "State": "delete-in-progress",  
    "StateMessage": "",  
    "NetworkInterfaceIds": [],  
    "VpcId": "vpc-0123abcd",  
    "AvailabilityZone": "us-east-1d",  
    "CreatedAt": "2023-02-07T12:05:37+00:00",  
    "SubnetId": "subnet-0123abcd"  
  }  
}
```

For more information, see [Remove EC2 Instance Connect Endpoint](#) in the *Amazon EC2 User Guide*.

- For API details, see [DeleteInstanceConnectEndpoint](#) in *AWS CLI Command Reference*.

delete-instance-event-window

The following code example shows how to use `delete-instance-event-window`.

AWS CLI

Example 1: To delete an event window

The following `delete-instance-event-window` example deletes an event window.

```
aws ec2 delete-instance-event-window \  
  --region us-east-1 \  
  --event-window-id ew-0123456789example
```

```
--instance-event-window-id iew-0abcdef1234567890
```

Output:

```
{
  "InstanceEventWindowState": {
    "InstanceEventWindowId": "iew-0abcdef1234567890",
    "State": "deleting"
  }
}
```

For event window constraints, see [Considerations](#) in the Scheduled Events section of the *Amazon EC2 User Guide*.

Example 2: To force delete an event window

The following `delete-instance-event-window` example force deletes an event window if the event window is currently associated with targets.

```
aws ec2 delete-instance-event-window \
  --region us-east-1 \
  --instance-event-window-id iew-0abcdef1234567890 \
  --force-delete
```

Output:

```
{
  "InstanceEventWindowState": {
    "InstanceEventWindowId": "iew-0abcdef1234567890",
    "State": "deleting"
  }
}
```

For event window constraints, see [Considerations](#) in the Scheduled Events section of the *Amazon EC2 User Guide*.

- For API details, see [DeleteInstanceEventWindow](#) in *AWS CLI Command Reference*.

delete-internet-gateway

The following code example shows how to use `delete-internet-gateway`.

AWS CLI

To delete an internet gateway

The following `delete-internet-gateway` example deletes the specified internet gateway.

```
aws ec2 delete-internet-gateway \  
  --internet-gateway-id igw-0d0fb496b3EXAMPLE
```

This command produces no output.

For more information, see [Internet gateways](#) in the *Amazon VPC User Guide*.

- For API details, see [DeleteInternetGateway](#) in *AWS CLI Command Reference*.

`delete-ipam-pool`

The following code example shows how to use `delete-ipam-pool`.

AWS CLI

To delete an IPAM pool

In this example, you're a IPAM delegated admin who wants to delete an IPAM pool that you no longer need, but the pool has a CIDR provisioned to it. You cannot delete a pool if it has CIDRs provisioned to it unless you use the `--cascade` option, so you'll use `--cascade`.

To complete this request:

You'll need the IPAM pool ID which you can get with [describe-ipam-pools](#). The `--region` must be the IPAM home Region.

The following `delete-ipam-pool` example deletes an IPAM pool in your AWS account.

```
aws ec2 delete-ipam-pool \  
  --ipam-pool-id ipam-pool-050c886a3ca41cd5b \  
  --cascade \  
  --region us-east-1
```

Output:

```
{
```

```
"IpamPool": {
  "OwnerId": "320805250157",
  "IpamPoolId": "ipam-pool-050c886a3ca41cd5b",
  "IpamPoolArn": "arn:aws:ec2::320805250157:ipam-pool/ipam-
pool-050c886a3ca41cd5b",
  "IpamScopeArn": "arn:aws:ec2::320805250157:ipam-scope/ipam-
scope-0a158dde35c51107b",
  "IpamScopeType": "private",
  "IpamArn": "arn:aws:ec2::320805250157:ipam/ipam-005f921c17ebd5107",
  "IpamRegion": "us-east-1",
  "Locale": "None",
  "PoolDepth": 1,
  "State": "delete-in-progress",
  "Description": "example",
  "AutoImport": false,
  "AddressFamily": "ipv4",
  "AllocationMinNetmaskLength": 0,
  "AllocationMaxNetmaskLength": 32
}
```

For more information, see [Delete a pool](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [DeleteIpamPool](#) in *AWS CLI Command Reference*.

delete-ipam-resource-discovery

The following code example shows how to use `delete-ipam-resource-discovery`.

AWS CLI

To delete a resource discovery

In this example, you're a IPAM delegated admin who wants to delete a non-default resource discovery that you created to share with another IPAM admin during the process of integrating IPAM with accounts outside of your organization.

To complete this request:

The `--region` must be the Region where you created the resource discovery. You cannot delete a default resource discovery if `"IsDefault": true`. A default resource discovery is one that is created automatically in the account that creates an IPAM. To delete a default resource discovery, you have to delete the IPAM.

The following `delete-ipam-resource-discovery` example deletes a resource discovery.

```
aws ec2 delete-ipam-resource-discovery \  
  --ipam-resource-discovery-id ipam-res-disco-0e39761475298ee0f \  
  --region us-east-1
```

Output:

```
{  
  "IpamResourceDiscovery": {  
    "OwnerId": "149977607591",  
    "IpamResourceDiscoveryId": "ipam-res-disco-0e39761475298ee0f",  
    "IpamResourceDiscoveryArn": "arn:aws:ec2::149977607591:ipam-resource-  
discovery/ipam-res-disco-0e39761475298ee0f",  
    "IpamResourceDiscoveryRegion": "us-east-1",  
    "OperatingRegions": [  
      {  
        "RegionName": "us-east-1"  
      }  
    ],  
    "IsDefault": false,  
    "State": "delete-in-progress"  
  }  
}
```

For more information about resource discoveries, see [Work with resource discoveries](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [DeleteIpamResourceDiscovery](#) in *AWS CLI Command Reference*.

delete-ipam-scope

The following code example shows how to use `delete-ipam-scope`.

AWS CLI

To delete an IPAM scope

The following `delete-ipam-scope` example deletes an IPAM.

```
aws ec2 delete-ipam-scope \  
  --ipam-scope-id ipam-scope-01c1ebab2b63bd7e4
```

Output:

```
{
  "IpamScope": {
    "OwnerId": "123456789012",
    "IpamScopeId": "ipam-scope-01c1ebab2b63bd7e4",
    "IpamScopeArn": "arn:aws:ec2::123456789012:ipam-scope/ipam-
scope-01c1ebab2b63bd7e4",
    "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",
    "IpamRegion": "us-east-1",
    "IpamScopeType": "private",
    "IsDefault": false,
    "Description": "Example description",
    "PoolCount": 0,
    "State": "delete-in-progress"
  }
}
```

For more information, see [Delete a scope](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [DeleteIpamScope](#) in *AWS CLI Command Reference*.

delete-ipam

The following code example shows how to use `delete-ipam`.

AWS CLI**To delete an IPAM**

The following `delete-ipam` example deletes an IPAM.

```
aws ec2 delete-ipam \
  --ipam-id ipam-036486dfa6af58ee0
```

Output:

```
{
  "Ipam": {
    "OwnerId": "123456789012",
    "IpamId": "ipam-036486dfa6af58ee0",
    "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-036486dfa6af58ee0",
```

```
"IpamRegion": "us-east-1",
"PublicDefaultScopeId": "ipam-scope-071b8042b0195c183",
"PrivateDefaultScopeId": "ipam-scope-0807405dece705a30",
"ScopeCount": 2,
"OperatingRegions": [
  {
    "RegionName": "us-east-1"
  },
  {
    "RegionName": "us-east-2"
  },
  {
    "RegionName": "us-west-1"
  }
],
"State": "delete-in-progress"
}
}
```

For more information, see [Delete an IPAM](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [Deletelpam](#) in *AWS CLI Command Reference*.

delete-key-pair

The following code example shows how to use `delete-key-pair`.

AWS CLI

To delete a key pair

The following `delete-key-pair` example deletes the specified key pair.

```
aws ec2 delete-key-pair \
  --key-name my-key-pair
```

Output:

```
{
  "Return": true,
  "KeyPairId": "key-03c8d3aceb53b507"
}
```

For more information, see [Create and delete key pairs](#) in the *AWS Command Line Interface User Guide*.

- For API details, see [DeleteKeyPair](#) in *AWS CLI Command Reference*.

delete-launch-template-versions

The following code example shows how to use `delete-launch-template-versions`.

AWS CLI

To delete a launch template version

This example deletes the specified launch template version.

Command:

```
aws ec2 delete-launch-template-versions --launch-template-id lt-0abcd290751193123 --versions 1
```

Output:

```
{
  "UnsuccessfullyDeletedLaunchTemplateVersions": [],
  "SuccessfullyDeletedLaunchTemplateVersions": [
    {
      "LaunchTemplateName": "TestVersion",
      "VersionNumber": 1,
      "LaunchTemplateId": "lt-0abcd290751193123"
    }
  ]
}
```

- For API details, see [DeleteLaunchTemplateVersions](#) in *AWS CLI Command Reference*.

delete-launch-template

The following code example shows how to use `delete-launch-template`.

AWS CLI

To delete a launch template

This example deletes the specified launch template.

Command:

```
aws ec2 delete-launch-template --launch-template-id lt-0abcd290751193123
```

Output:

```
{
  "LaunchTemplate": {
    "LatestVersionNumber": 2,
    "LaunchTemplateId": "lt-0abcd290751193123",
    "LaunchTemplateName": "TestTemplate",
    "DefaultVersionNumber": 2,
    "CreatedBy": "arn:aws:iam::123456789012:root",
    "CreateTime": "2017-11-23T16:46:25.000Z"
  }
}
```

- For API details, see [DeleteLaunchTemplate](#) in *AWS CLI Command Reference*.

delete-local-gateway-route-table-virtual-interface-group-association

The following code example shows how to use `delete-local-gateway-route-table-virtual-interface-group-association`.

AWS CLI

To disassociate a local gateway route table from a virtual interfaces (VIFs) group

The following `delete-local-gateway-route-table-virtual-interface-group-association` example deletes the association between the specified local gateway route table and VIF group.

```
aws ec2 delete-local-gateway-route-table-virtual-interface-group-association \
  --local-gateway-route-table-virtual-interface-group-association-id lgw-vif-grp-
  assoc-exampleid12345678
```

Output:

```
{
```

```

    "LocalGatewayRouteTableVirtualInterfaceGroupAssociation": {
      "LocalGatewayRouteTableVirtualInterfaceGroupAssociationId": "lgw-vif-grp-
assoc-exampleid12345678",
      "LocalGatewayVirtualInterfaceGroupId": "lgw-vif-grp-exampleid0123abcd",
      "LocalGatewayId": "lgw-exampleid11223344",
      "LocalGatewayRouteTableId": "lgw-rtb-exampleidabcd1234",
      "LocalGatewayRouteTableArn": "arn:aws:ec2:us-west-2:111122223333:local-
gateway-route-table/lgw-rtb-exampleidabcd1234",
      "OwnerId": "111122223333",
      "State": "disassociating",
      "Tags": []
    }
  }
}

```

For more information, see [VIF group associations](#) in the *AWS Outposts User Guide*.

- For API details, see [DeleteLocalGatewayRouteTableVirtualInterfaceGroupAssociation](#) in *AWS CLI Command Reference*.

delete-local-gateway-route-table-vpc-association

The following code example shows how to use `delete-local-gateway-route-table-vpc-association`.

AWS CLI

To disassociate a local gateway route table from a VPC

The following `delete-local-gateway-route-table-vpc-association` example deletes the association between the specified local gateway route table and VPC.

```

aws ec2 delete-local-gateway-route-table-vpc-association \
  --local-gateway-route-table-vpc-association-id vpc-example0123456789

```

Output:

```

{
  "LocalGatewayRouteTableVpcAssociation": {
    "LocalGatewayRouteTableVpcAssociationId": "lgw-vpc-assoc-abcd1234wxyz56789",
    "LocalGatewayRouteTableId": "lgw-rtb-abcdefg1234567890",
    "LocalGatewayRouteTableArn": "arn:aws:ec2:us-west-2:555555555555:local-
gateway-route-table/lgw-rtb-abcdefg1234567890",
  }
}

```

```
    "LocalGatewayId": "lgw-exampleid01234567",
    "VpcId": "vpc-example0123456789",
    "OwnerId": "555555555555",
    "State": "disassociating"
  }
}
```

For more information, see [VPC associations](#) in the *AWS Outposts User Guide*.

- For API details, see [DeleteLocalGatewayRouteTableVpcAssociation](#) in *AWS CLI Command Reference*.

delete-local-gateway-route-table

The following code example shows how to use `delete-local-gateway-route-table`.

AWS CLI

To delete a local gateway route table

The following `delete-local-gateway-route-table` example creates a local gateway route table with the direct VPC routing mode.

```
aws ec2 delete-local-gateway-route-table \
  --local-gateway-route-table-id lgw-rtb-abcdefg1234567890
```

Output:

```
{
  "LocalGatewayRouteTable": {
    "LocalGatewayRouteTableId": "lgw-rtb-abcdefg1234567890",
    "LocalGatewayRouteTableArn": "arn:aws:ec2:us-west-2:111122223333:local-gateway-route-table/lgw-rtb-abcdefg1234567890",
    "LocalGatewayId": "lgw-1a2b3c4d5e6f7g8h9",
    "OutpostArn": "arn:aws:outposts:us-west-2:111122223333:outpost/op-021345abcdef67890",
    "OwnerId": "111122223333",
    "State": "deleting",
    "Tags": [],
    "Mode": "direct-vpc-routing"
  }
}
```

For more information, see [Local gateway route tables](#) in the *AWS Outposts User Guide*.

- For API details, see [DeleteLocalGatewayRouteTable](#) in *AWS CLI Command Reference*.

delete-local-gateway-route

The following code example shows how to use `delete-local-gateway-route`.

AWS CLI

To delete a route from a local gateway route table

The following `delete-local-gateway-route` example deletes the specified route from the specified local gateway route table.

```
aws ec2 delete-local-gateway-route \
  --destination-cidr-block 0.0.0.0/0 \
  --local-gateway-route-table-id lgw-rtb-059615ef7dEXAMPLE
```

Output:

```
{
  "Route": {
    "DestinationCidrBlock": "0.0.0.0/0",
    "LocalGatewayVirtualInterfaceGroupId": "lgw-vif-grp-07145b276bEXAMPLE",
    "Type": "static",
    "State": "deleted",
    "LocalGatewayRouteTableId": "lgw-rtb-059615ef7EXAMPLE"
  }
}
```

- For API details, see [DeleteLocalGatewayRoute](#) in *AWS CLI Command Reference*.

delete-managed-prefix-list

The following code example shows how to use `delete-managed-prefix-list`.

AWS CLI

To delete a prefix list

The following `delete-managed-prefix-list` example deletes the specified prefix list.


```
aws ec2 delete-managed-prefix-list \  
  --prefix-list-id pl-0123456abcabcabc1
```

Output:

```
{  
  "PrefixList": {  
    "PrefixListId": "pl-0123456abcabcabc1",  
    "AddressFamily": "IPv4",  
    "State": "delete-in-progress",  
    "PrefixListArn": "arn:aws:ec2:us-west-2:123456789012:prefix-list/  
pl-0123456abcabcabc1",  
    "PrefixListName": "test",  
    "MaxEntries": 10,  
    "Version": 1,  
    "OwnerId": "123456789012"  
  }  
}
```

For more information, see [Managed prefix lists](#) in the *Amazon VPC User Guide*.

- For API details, see [DeleteManagedPrefixList](#) in *AWS CLI Command Reference*.

delete-nat-gateway

The following code example shows how to use `delete-nat-gateway`.

AWS CLI

To delete a NAT gateway

This example deletes NAT gateway `nat-04ae55e711cec5680`.

Command:

```
aws ec2 delete-nat-gateway --nat-gateway-id nat-04ae55e711cec5680
```

Output:

```
{  
  "NatGatewayId": "nat-04ae55e711cec5680"  
}
```

- For API details, see [DeleteNatGateway](#) in *AWS CLI Command Reference*.

delete-network-acl-entry

The following code example shows how to use `delete-network-acl-entry`.

AWS CLI

To delete a network ACL entry

This example deletes ingress rule number 100 from the specified network ACL. If the command succeeds, no output is returned.

Command:

```
aws ec2 delete-network-acl-entry --network-acl-id acl-5fb85d36 --ingress --rule-number 100
```

- For API details, see [DeleteNetworkAclEntry](#) in *AWS CLI Command Reference*.

delete-network-acl

The following code example shows how to use `delete-network-acl`.

AWS CLI

To delete a network ACL

This example deletes the specified network ACL. If the command succeeds, no output is returned.

Command:

```
aws ec2 delete-network-acl --network-acl-id acl-5fb85d36
```

- For API details, see [DeleteNetworkAcl](#) in *AWS CLI Command Reference*.

delete-network-insights-access-scope-analysis

The following code example shows how to use `delete-network-insights-access-scope-analysis`.

AWS CLI

To delete a Network Access Scope analysis

The following `delete-network-insights-access-scope-analysis` example deletes the specified Network Access Scope analysis.

```
aws ec2 delete-network-insights-access-scope-analysis \  
  --network-insights-access-scope-analysis-id nisa-01234567891abcdef
```

Output:

```
{  
  "NetworkInsightsAccessScopeAnalysisId": "nisa-01234567891abcdef"  
}
```

For more information, see [Getting started with Network Access Analyzer using the AWS CLI](#) in the *Network Access Analyzer Guide*.

- For API details, see [DeleteNetworkInsightsAccessScopeAnalysis](#) in *AWS CLI Command Reference*.

delete-network-insights-access-scope

The following code example shows how to use `delete-network-insights-access-scope`.

AWS CLI

To delete a Network Access Scope

The following `delete-network-insights-access-scope` example deletes the specified Network Access Scope.

```
aws ec2 delete-network-insights-access-scope \  
  --network-insights-access-scope-id nis-123456789abc01234
```

Output:

```
{  
  "NetworkInsightsAccessScopeId": "nis-123456789abc01234"  
}
```

For more information, see [Getting started with Network Access Analyzer using the AWS CLI](#) in the *Network Access Analyzer Guide*.

- For API details, see [DeleteNetworkInsightsAccessScope](#) in *AWS CLI Command Reference*.

delete-network-insights-analysis

The following code example shows how to use `delete-network-insights-analysis`.

AWS CLI

To delete a path analysis

The following `delete-network-insights-analysis` example deletes the specified analysis.

```
aws ec2 delete-network-insights-analysis \  
  --network-insights-analysis-id nia-02207aa13eb480c7a
```

Output:

```
{  
  "NetworkInsightsAnalysisId": "nia-02207aa13eb480c7a"  
}
```

For more information, see [Getting started using the AWS CLI](#) in the *Reachability Analyzer Guide*.

- For API details, see [DeleteNetworkInsightsAnalysis](#) in *AWS CLI Command Reference*.

delete-network-insights-path

The following code example shows how to use `delete-network-insights-path`.

AWS CLI

To delete a path

The following `delete-network-insights-path` example deletes the specified path. Before you can delete a path, you must delete all its analyses using the `delete-network-insights-analysis` command.

```
aws ec2 delete-network-insights-path \  
  --network-insights-path-id nia-02207aa13eb480c7a
```

```
--network-insights-path-id nip-0b26f224f1d131fa8
```

Output:

```
{
  "NetworkInsightsPathId": "nip-0b26f224f1d131fa8"
}
```

For more information, see [Getting started using the AWS CLI](#) in the *Reachability Analyzer Guide*.

- For API details, see [DeleteNetworkInsightsPath](#) in *AWS CLI Command Reference*.

delete-network-interface-permission

The following code example shows how to use `delete-network-interface-permission`.

AWS CLI

To delete a network interface permission

This example deletes the specified network interface permission.

Command:

```
aws ec2 delete-network-interface-permission --network-interface-permission-id eni-
perm-06fd19020ede149ea
```

Output:

```
{
  "Return": true
}
```

- For API details, see [DeleteNetworkInterfacePermission](#) in *AWS CLI Command Reference*.

delete-network-interface

The following code example shows how to use `delete-network-interface`.

AWS CLI

To delete a network interface

This example deletes the specified network interface. If the command succeeds, no output is returned.

Command:

```
aws ec2 delete-network-interface --network-interface-id eni-e5aa89a3
```

- For API details, see [DeleteNetworkInterface](#) in *AWS CLI Command Reference*.

delete-placement-group

The following code example shows how to use `delete-placement-group`.

AWS CLI

To delete a placement group

This example command deletes the specified placement group.

Command:

```
aws ec2 delete-placement-group --group-name my-cluster
```

- For API details, see [DeletePlacementGroup](#) in *AWS CLI Command Reference*.

delete-queued-reserved-instances

The following code example shows how to use `delete-queued-reserved-instances`.

AWS CLI

To delete a queued purchase

The following `delete-queued-reserved-instances` example deletes the specified Reserved Instance, which was queued for purchase.

```
aws ec2 delete-queued-reserved-instances \  
  --reserved-instances-ids af9f760e-6f91-4559-85f7-4980eexample
```

Output:

```
{
```

```
"SuccessfulQueuedPurchaseDeletions": [  
  {  
    "ReservedInstancesId": "af9f760e-6f91-4559-85f7-4980eexample"  
  }  
],  
"FailedQueuedPurchaseDeletions": []  
}
```

- For API details, see [DeleteQueuedReservedInstances](#) in *AWS CLI Command Reference*.

delete-route-table

The following code example shows how to use `delete-route-table`.

AWS CLI

To delete a route table

This example deletes the specified route table. If the command succeeds, no output is returned.

Command:

```
aws ec2 delete-route-table --route-table-id rtb-22574640
```

- For API details, see [DeleteRouteTable](#) in *AWS CLI Command Reference*.

delete-route

The following code example shows how to use `delete-route`.

AWS CLI

To delete a route

This example deletes the specified route from the specified route table. If the command succeeds, no output is returned.

Command:

```
aws ec2 delete-route --route-table-id rtb-22574640 --destination-cidr-block  
0.0.0.0/0
```

- For API details, see [DeleteRoute](#) in *AWS CLI Command Reference*.

delete-security-group

The following code example shows how to use delete-security-group.

AWS CLI

[EC2-Classic] To delete a security group

This example deletes the security group named MySecurityGroup. If the command succeeds, no output is returned.

Command:

```
aws ec2 delete-security-group --group-name MySecurityGroup
```

[EC2-VPC] To delete a security group

This example deletes the security group with the ID sg-903004f8. Note that you can't reference a security group for EC2-VPC by name. If the command succeeds, no output is returned.

Command:

```
aws ec2 delete-security-group --group-id sg-903004f8
```

For more information, see *Using Security Groups in the AWS Command Line Interface User Guide*.

- For API details, see [DeleteSecurityGroup](#) in *AWS CLI Command Reference*.

delete-snapshot

The following code example shows how to use delete-snapshot.

AWS CLI

To delete a snapshot

This example command deletes a snapshot with the snapshot ID of snap-1234567890abcdef0. If the command succeeds, no output is returned.

Command:

```
aws ec2 delete-snapshot --snapshot-id snap-1234567890abcdef0
```

- For API details, see [DeleteSnapshot](#) in *AWS CLI Command Reference*.

delete-spot-datafeed-subscription

The following code example shows how to use `delete-spot-datafeed-subscription`.

AWS CLI**To cancel a Spot Instance data feed subscription**

This example command deletes a Spot data feed subscription for the account. If the command succeeds, no output is returned.

Command:

```
aws ec2 delete-spot-datafeed-subscription
```

- For API details, see [DeleteSpotDatafeedSubscription](#) in *AWS CLI Command Reference*.

delete-subnet-cidr-reservation

The following code example shows how to use `delete-subnet-cidr-reservation`.

AWS CLI**To delete a subnet CIDR reservation**

The following `delete-subnet-cidr-reservation` example deletes the specified subnet CIDR reservation.

```
aws ec2 delete-subnet-cidr-reservation \  
  --subnet-cidr-reservation-id scr-044f977c4eEXAMPLE
```

Output:

```
{  
  "DeletedSubnetCidrReservation": {
```

```
    "SubnetCidrReservationId": "scr-044f977c4eEXAMPLE",
    "SubnetId": "subnet-03c51e2e6cEXAMPLE",
    "Cidr": "10.1.0.16/28",
    "ReservationType": "prefix",
    "OwnerId": "123456789012"
  }
}
```

For more information, see [Subnet CIDR reservations](#) in the *Amazon VPC User Guide*.

- For API details, see [DeleteSubnetCidrReservation](#) in *AWS CLI Command Reference*.

delete-subnet

The following code example shows how to use `delete-subnet`.

AWS CLI

To delete a subnet

This example deletes the specified subnet. If the command succeeds, no output is returned.

Command:

```
aws ec2 delete-subnet --subnet-id subnet-9d4a7b6c
```

- For API details, see [DeleteSubnet](#) in *AWS CLI Command Reference*.

delete-tags

The following code example shows how to use `delete-tags`.

AWS CLI

Example 1: To delete a tag from a resource

The following `delete-tags` example deletes the tag `Stack=Test` from the specified image. When you specify both a value and a key name, the tag is deleted only if the tag's value matches the specified value.

```
aws ec2 delete-tags \  
  --resources ami-1234567890abcdef0 \  
  --tags Stack=Test
```

```
--tags Key=Stack,Value=Test
```

It's optional to specify the value for a tag. The following `delete-tags` example deletes the tag with the key name `purpose` from the specified instance, regardless of the tag value for the tag.

```
aws ec2 delete-tags \  
  --resources i-1234567890abcdef0 \  
  --tags Key=purpose
```

If you specify the empty string as the tag value, the tag is deleted only if the tag's value is the empty string. The following `delete-tags` example specifies the empty string as the tag value for the tag to delete.

```
aws ec2 delete-tags \  
  --resources i-1234567890abcdef0 \  
  --tags Key=Name,Value=
```

Example 2: To delete a tag from multiple resources

The following `delete-tags` example deletes the tag ```Purpose=Test``` from both an instance and an AMI. As shown in the previous example, you can omit the tag value from the command.

```
aws ec2 delete-tags \  
  --resources i-1234567890abcdef0 ami-1234567890abcdef0 \  
  --tags Key=Purpose
```

- For API details, see [DeleteTags](#) in *AWS CLI Command Reference*.

delete-traffic-mirror-filter-rule

The following code example shows how to use `delete-traffic-mirror-filter-rule`.

AWS CLI

To delete a traffic mirror filter rule

The following `delete-traffic-mirror-filter-rule` example deletes the specified traffic mirror filter rule.

```
aws ec2 delete-traffic-mirror-filter-rule \  
  --traffic-mirror-filter-rule-id <id>
```

```
--traffic-mirror-filter-rule-id tmfr-081f71283bEXAMPLE
```

Output:

```
{  
  "TrafficMirrorFilterRuleId": "tmfr-081f71283bEXAMPLE"  
}
```

For more information, see [Modify Your Traffic Mirror Filter Rules](#) in the *AWS Traffic Mirroring Guide*.

- For API details, see [DeleteTrafficMirrorFilterRule](#) in *AWS CLI Command Reference*.

delete-traffic-mirror-filter

The following code example shows how to use `delete-traffic-mirror-filter`.

AWS CLI**To delete a traffic mirror filter**

The following `delete-traffic-mirror-filter` example deletes the specified traffic mirror filter.

```
aws ec2 delete-traffic-mirror-filter \  
  --traffic-mirror-filter-id tmf-0be0b25fcdEXAMPLE
```

Output:

```
{  
  "TrafficMirrorFilterId": "tmf-0be0b25fcdEXAMPLE"  
}
```

For more information, see [Delete a Traffic Mirror Filter](#) in the *AWS Traffic Mirroring Guide*.

- For API details, see [DeleteTrafficMirrorFilter](#) in *AWS CLI Command Reference*.

delete-traffic-mirror-session

The following code example shows how to use `delete-traffic-mirror-session`.

AWS CLI

To delete a traffic mirror session

The following `delete-traffic-mirror-session` example deletes the specified traffic mirror-session.

```
aws ec2 delete-traffic-mirror-session \  
  --traffic-mirror-session-id tms-0af3141ce5EXAMPLE
```

Output:

```
{  
  "TrafficMirrorSessionId": "tms-0af3141ce5EXAMPLE"  
}
```

For more information, see [Delete a Traffic Mirror Session](#) in the *AWS Traffic Mirroring Guide*.

- For API details, see [DeleteTrafficMirrorSession](#) in *AWS CLI Command Reference*.

delete-traffic-mirror-target

The following code example shows how to use `delete-traffic-mirror-target`.

AWS CLI

To delete a traffic mirror target

The following `delete-traffic-mirror-target` example deletes the specified traffic mirror target.

```
aws ec2 delete-traffic-mirror-target \  
  --traffic-mirror-target-id tmt-060f48ce9EXAMPLE
```

Output:

```
{  
  "TrafficMirrorTargetId": "tmt-060f48ce9EXAMPLE"  
}
```

For more information, see [Delete a Traffic Mirror Target](#) in the *AWS Traffic Mirroring Guide*.

- For API details, see [DeleteTrafficMirrorTarget](#) in *AWS CLI Command Reference*.

delete-transit-gateway-connect-peer

The following code example shows how to use `delete-transit-gateway-connect-peer`.

AWS CLI

To delete a Transit Gateway Connect peer

The following `delete-transit-gateway-connect-peer` example deletes the specified Connect peer.

```
aws ec2 delete-transit-gateway-connect-peer \  
  --transit-gateway-connect-peer-id tgw-connect-peer-0666adbac4EXAMPLE
```

Output:

```
{  
  "TransitGatewayConnectPeer": {  
    "TransitGatewayAttachmentId": "tgw-attach-0f0927767cEXAMPLE",  
    "TransitGatewayConnectPeerId": "tgw-connect-peer-0666adbac4EXAMPLE",  
    "State": "deleting",  
    "CreationTime": "2021-10-13T03:35:17.000Z",  
    "ConnectPeerConfiguration": {  
      "TransitGatewayAddress": "10.0.0.234",  
      "PeerAddress": "172.31.1.11",  
      "InsideCidrBlocks": [  
        "169.254.6.0/29"  
      ],  
      "Protocol": "gre",  
      "BgpConfigurations": [  
        {  
          "TransitGatewayAsn": 64512,  
          "PeerAsn": 64512,  
          "TransitGatewayAddress": "169.254.6.2",  
          "PeerAddress": "169.254.6.1",  
          "BgpStatus": "down"  
        },  
        {  
          "TransitGatewayAsn": 64512,  
          "PeerAsn": 64512,
```

```

        "TransitGatewayAddress": "169.254.6.3",
        "PeerAddress": "169.254.6.1",
        "BgpStatus": "down"
      }
    ]
  }
}

```

For more information, see [Transit gateway Connect attachments and Transit Gateway Connect peers](#) in the *Transit Gateways Guide*.

- For API details, see [DeleteTransitGatewayConnectPeer](#) in *AWS CLI Command Reference*.

delete-transit-gateway-connect

The following code example shows how to use delete-transit-gateway-connect.

AWS CLI

To delete a transit gateway Connect attachment

The following delete-transit-gateway-connect example deletes the specified Connect attachment.

```

aws ec2 delete-transit-gateway-connect \
  --transit-gateway-attachment-id tgw-attach-037012e5dcEXAMPLE

```

Output:

```

{
  "TransitGatewayConnect": {
    "TransitGatewayAttachmentId": "tgw-attach-037012e5dcEXAMPLE",
    "TransportTransitGatewayAttachmentId": "tgw-attach-0a89069f57EXAMPLE",
    "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",
    "State": "deleting",
    "CreationTime": "2021-03-09T19:59:17+00:00",
    "Options": {
      "Protocol": "gre"
    }
  }
}

```

For more information, see [Transit gateway Connect attachments and Transit Gateway Connect peers](#) in the *Transit Gateways Guide*.

- For API details, see [DeleteTransitGatewayConnect](#) in *AWS CLI Command Reference*.

delete-transit-gateway-multicast-domain

The following code example shows how to use `delete-transit-gateway-multicast-domain`.

AWS CLI

To delete a transit gateway multicast domain

The following `delete-transit-gateway-multicast-domain` example deletes the specified multicast domain.

```
aws ec2 delete-transit-gateway-multicast-domain \  
  --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef7EXAMPLE
```

Output:

```
{  
  "TransitGatewayMulticastDomain": {  
    "TransitGatewayMulticastDomainId": "tgw-mcast-domain-02bb79002bEXAMPLE",  
    "TransitGatewayId": "tgw-0d88d2d0d5EXAMPLE",  
    "State": "deleting",  
    "CreationTime": "2019-11-20T22:02:03.000Z"  
  }  
}
```

For more information, see [Managing multicast domains](#) in the *Transit Gateways Guide*.

- For API details, see [DeleteTransitGatewayMulticastDomain](#) in *AWS CLI Command Reference*.

delete-transit-gateway-peering-attachment

The following code example shows how to use `delete-transit-gateway-peering-attachment`.

AWS CLI

To delete a transit gateway peering attachment

The following `delete-transit-gateway-peering-attachment` example deletes the specified transit gateway peering attachment.

```
aws ec2 delete-transit-gateway-peering-attachment \  
  --transit-gateway-attachment-id tgw-attach-4455667788aabbccd
```

Output:

```
{  
  "TransitGatewayPeeringAttachment": {  
    "TransitGatewayAttachmentId": "tgw-attach-4455667788aabbccd",  
    "RequesterTgwInfo": {  
      "TransitGatewayId": "tgw-123abc05e04123abc",  
      "OwnerId": "123456789012",  
      "Region": "us-west-2"  
    },  
    "AcceptorTgwInfo": {  
      "TransitGatewayId": "tgw-11223344aabbcc112",  
      "OwnerId": "123456789012",  
      "Region": "us-east-2"  
    },  
    "State": "deleting",  
    "CreationTime": "2019-12-09T11:38:31.000Z"  
  }  
}
```

For more information, see [Transit Gateway Peering Attachments](#) in the *Transit Gateways Guide*.

- For API details, see [DeleteTransitGatewayPeeringAttachment](#) in *AWS CLI Command Reference*.

delete-transit-gateway-policy-table

The following code example shows how to use `delete-transit-gateway-policy-table`.

AWS CLI

To delete a transit gateway policy table

The following `delete-transit-gateway-policy-table` example deletes the specified transit gateway policy table.

```
aws ec2 delete-transit-gateway-policy-table \  
  --transit-gateway-policy-table-id tgw-policy-table-123456789012
```

```
--transit-gateway-policy-table-id tgw-ptb-0a16f134b78668a81
```

Output:

```
{
  "TransitGatewayPolicyTables": [
    {
      "TransitGatewayPolicyTableId": "tgw-ptb-0a16f134b78668a81",
      "TransitGatewayId": "tgw-067f8505c18f0bd6e",
      "State": "deleting",
      "CreationTime": "2023-11-28T16:36:43+00:00",
      "Tags": []
    }
  ]
}
```

For more information, see [Transit gateway policy tables](#) in the *Transit Gateway User Guide*.

- For API details, see [DeleteTransitGatewayPolicyTable](#) in *AWS CLI Command Reference*.

delete-transit-gateway-prefix-list-reference

The following code example shows how to use `delete-transit-gateway-prefix-list-reference`.

AWS CLI

To delete a prefix list reference

The following `delete-transit-gateway-prefix-list-reference` example deletes the specified prefix list reference.

```
aws ec2 delete-transit-gateway-prefix-list-reference \
  --transit-gateway-route-table-id tgw-rtb-0123456789abcd123 \
  --prefix-list-id pl-11111122222222333
```

Output:

```
{
  "TransitGatewayPrefixListReference": {
    "TransitGatewayRouteTableId": "tgw-rtb-0123456789abcd123",
```

```

    "PrefixListId": "pl-11111122222222333",
    "PrefixListOwnerId": "123456789012",
    "State": "deleting",
    "Blackhole": false,
    "TransitGatewayAttachment": {
      "TransitGatewayAttachmentId": "tgw-attach-aabbccddaabbccaab",
      "ResourceType": "vpc",
      "ResourceId": "vpc-112233445566aabbcc"
    }
  }
}

```

For more information, see [Prefix list references](#) in the *Transit Gateways Guide*.

- For API details, see [DeleteTransitGatewayPrefixListReference](#) in *AWS CLI Command Reference*.

delete-transit-gateway-route-table

The following code example shows how to use `delete-transit-gateway-route-table`.

AWS CLI

To delete a transit gateway route table

The following `delete-transit-gateway-route-table` example deletes the specified transit gateway route table.

```

aws ec2 delete-transit-gateway-route-table \
  --transit-gateway-route-table-id tgw-rtb-0b6f6aaa01EXAMPLE

```

Output:

```

{
  "TransitGatewayRouteTable": {
    "TransitGatewayRouteTableId": "tgw-rtb-0b6f6aaa01EXAMPLE",
    "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",
    "State": "deleting",
    "DefaultAssociationRouteTable": false,
    "DefaultPropagationRouteTable": false,
    "CreationTime": "2019-07-17T20:27:26.000Z"
  }
}

```

For more information, see [Delete a transit gateway route table](#) in the *Transit Gateways Guide*.

- For API details, see [DeleteTransitGatewayRouteTable](#) in *AWS CLI Command Reference*.

delete-transit-gateway-route

The following code example shows how to use delete-transit-gateway-route.

AWS CLI

To delete a CIDR block from a route table

The following delete-transit-gateway-route example deletes the CIDR block from the specified transit gateway route table.

```
aws ec2 delete-transit-gateway-route \
  --transit-gateway-route-table-id tgw-rtb-0b6f6aaa01EXAMPLE \
  --destination-cidr-block 10.0.2.0/24
```

Output:

```
{
  "Route": {
    "DestinationCidrBlock": "10.0.2.0/24",
    "TransitGatewayAttachments": [
      {
        "ResourceId": "vpc-0065acced4EXAMPLE",
        "TransitGatewayAttachmentId": "tgw-attach-0b5968d3b6EXAMPLE",
        "ResourceType": "vpc"
      }
    ],
    "Type": "static",
    "State": "deleted"
  }
}
```

For more information, see [Delete a static route](#) in the *Transit Gateways Guide*.

- For API details, see [DeleteTransitGatewayRoute](#) in *AWS CLI Command Reference*.

delete-transit-gateway-vpc-attachment

The following code example shows how to use delete-transit-gateway-vpc-attachment.

AWS CLI

To delete a transit gateway VPC attachment

The following `delete-transit-gateway-vpc-attachment` example deletes the specified VPC attachment.

```
aws ec2 delete-transit-gateway-vpc-attachment \  
  --transit-gateway-attachment-id tgw-attach-0d2c54bdbEXAMPLE
```

Output:

```
{  
  "TransitGatewayVpcAttachment": {  
    "TransitGatewayAttachmentId": "tgw-attach-0d2c54bdb3EXAMPLE",  
    "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",  
    "VpcId": "vpc-0065acced4f61c651",  
    "VpcOwnerId": "111122223333",  
    "State": "deleting",  
    "CreationTime": "2019-07-17T16:04:27.000Z"  
  }  
}
```

For more information, see [Delete a VPC attachment](#) in the *Transit Gateways Guide*.

- For API details, see [DeleteTransitGatewayVpcAttachment](#) in *AWS CLI Command Reference*.

delete-transit-gateway

The following code example shows how to use `delete-transit-gateway`.

AWS CLI

To delete a transit gateway

The following `delete-transit-gateway` example deletes the specified transit gateway.

```
aws ec2 delete-transit-gateway \  
  --transit-gateway-id tgw-01f04542b2EXAMPLE
```

Output:

```
{
  "TransitGateway": {
    "TransitGatewayId": "tgw-01f04542b2EXAMPLE",
    "State": "deleting",
    "OwnerId": "123456789012",
    "Description": "Example Transit Gateway",
    "CreationTime": "2019-08-27T15:04:35.000Z",
    "Options": {
      "AmazonSideAsn": 64515,
      "AutoAcceptSharedAttachments": "disable",
      "DefaultRouteTableAssociation": "enable",
      "AssociationDefaultRouteTableId": "tgw-rtb-0ce7a6948fEXAMPLE",
      "DefaultRouteTablePropagation": "enable",
      "PropagationDefaultRouteTableId": "tgw-rtb-0ce7a6948fEXAMPLE",
      "VpnEcmpSupport": "enable",
      "DnsSupport": "enable"
    }
  }
}
```

For more information, see [Delete a transit gateway](#) in the *Transit Gateways Guide*.

- For API details, see [DeleteTransitGateway](#) in *AWS CLI Command Reference*.

delete-verified-access-endpoint

The following code example shows how to use `delete-verified-access-endpoint`.

AWS CLI

To delete a Verified Access endpoint

The following `delete-verified-access-endpoint` example deletes the specified Verified Access endpoint.

```
aws ec2 delete-verified-access-endpoint \
  --verified-access-endpoint-id vae-066fac616d4d546f2
```

Output:

```
{
```

```
"VerifiedAccessEndpoint": {
  "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
  "VerifiedAccessGroupId": "vagr-0dbe967baf14b7235",
  "VerifiedAccessEndpointId": "vae-066fac616d4d546f2",
  "ApplicationDomain": "example.com",
  "EndpointType": "network-interface",
  "AttachmentType": "vpc",
  "DomainCertificateArn": "arn:aws:acm:us-east-2:123456789012:certificate/
eb065ea0-26f9-4e75-a6ce-0a1a7EXAMPLE",
  "EndpointDomain": "my-ava-
app.edge-00c3372d53b1540bb.vai-0ce000c0b7643abea.prod.verified-access.us-
east-2.amazonaws.com",
  "SecurityGroupIds": [
    "sg-004915970c4c8f13a"
  ],
  "NetworkInterfaceOptions": {
    "NetworkInterfaceId": "eni-0aec70418c8d87a0f",
    "Protocol": "https",
    "Port": 443
  },
  "Status": {
    "Code": "deleting"
  },
  "Description": "Testing Verified Access",
  "CreationTime": "2023-08-25T20:54:43",
  "LastUpdatedTime": "2023-08-25T22:46:32"
}
}
```

For more information, see [Verified Access endpoints](#) in the *AWS Verified Access User Guide*.

- For API details, see [DeleteVerifiedAccessEndpoint](#) in *AWS CLI Command Reference*.

delete-verified-access-group

The following code example shows how to use `delete-verified-access-group`.

AWS CLI

To delete a Verified Access group

The following `delete-verified-access-group` example deletes the specified Verified Access group.

```
aws ec2 delete-verified-access-group \  
  --verified-access-group-id vagr-0dbe967baf14b7235
```

Output:

```
{  
  "VerifiedAccessGroup": {  
    "VerifiedAccessGroupId": "vagr-0dbe967baf14b7235",  
    "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",  
    "Description": "Testing Verified Access",  
    "Owner": "123456789012",  
    "VerifiedAccessGroupArn": "arn:aws:ec2:us-east-2:123456789012:verified-  
access-group/vagr-0dbe967baf14b7235",  
    "CreationTime": "2023-08-25T19:55:19",  
    "LastUpdatedTime": "2023-08-25T22:49:03",  
    "DeletionTime": "2023-08-26T00:58:31"  
  }  
}
```

For more information, see [Verified Access groups](#) in the *AWS Verified Access User Guide*.

- For API details, see [DeleteVerifiedAccessGroup](#) in *AWS CLI Command Reference*.

delete-verified-access-instance

The following code example shows how to use `delete-verified-access-instance`.

AWS CLI

To delete a Verified Access instance

The following `delete-verified-access-instance` example deletes the specified Verified Access instance.

```
aws ec2 delete-verified-access-instance \  
  --verified-access-instance-id vai-0ce000c0b7643abea
```

Output:

```
{  
  "VerifiedAccessInstance": {  
    "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
```



```
"Description": "Testing Verified Access",
"VerifiedAccessTrustProviders": [],
"CreationTime": "2023-08-25T18:27:56",
"LastUpdatedTime": "2023-08-26T01:00:18"
}
}
```

For more information, see [Verified Access instances](#) in the *AWS Verified Access User Guide*.

- For API details, see [DeleteVerifiedAccessInstance](#) in *AWS CLI Command Reference*.

delete-verified-access-trust-provider

The following code example shows how to use `delete-verified-access-trust-provider`.

AWS CLI

To delete a Verified Access trust provider

The following `delete-verified-access-trust-provider` example deletes the specified Verified Access trust provider.

```
aws ec2 delete-verified-access-trust-provider \
  --verified-access-trust-provider-id vatp-0bb32de759a3e19e7
```

Output:

```
{
  "VerifiedAccessTrustProvider": {
    "VerifiedAccessTrustProviderId": "vatp-0bb32de759a3e19e7",
    "Description": "Testing Verified Access",
    "TrustProviderType": "user",
    "UserTrustProviderType": "iam-identity-center",
    "PolicyReferenceName": "idc",
    "CreationTime": "2023-08-25T18:40:36",
    "LastUpdatedTime": "2023-08-25T18:40:36"
  }
}
```

For more information, see [Trust providers for Verified Access](#) in the *AWS Verified Access User Guide*.

- For API details, see [DeleteVerifiedAccessTrustProvider](#) in *AWS CLI Command Reference*.

delete-volume

The following code example shows how to use `delete-volume`.

AWS CLI

To delete a volume

This example command deletes an available volume with the volume ID of `vol-049df61146c4d7901`. If the command succeeds, no output is returned.

Command:

```
aws ec2 delete-volume --volume-id vol-049df61146c4d7901
```

- For API details, see [DeleteVolume](#) in *AWS CLI Command Reference*.

delete-vpc-endpoint-connection-notifications

The following code example shows how to use `delete-vpc-endpoint-connection-notifications`.

AWS CLI

To delete an endpoint connection notification

This example deletes the specified endpoint connection notification.

Command:

```
aws ec2 delete-vpc-endpoint-connection-notifications --connection-notification-ids vpce-nfn-008776de7e03f5abc
```

Output:

```
{
  "Unsuccessful": []
}
```

- For API details, see [DeleteVpcEndpointConnectionNotifications](#) in *AWS CLI Command Reference*.

delete-vpc-endpoint-service-configurations

The following code example shows how to use `delete-vpc-endpoint-service-configurations`.

AWS CLI

To delete an endpoint service configuration

This example deletes the specified endpoint service configuration.

Command:

```
aws ec2 delete-vpc-endpoint-service-configurations --service-ids vpce-  
svc-03d5ebb7d9579a2b3
```

Output:

```
{  
  "Unsuccessful": []  
}
```

- For API details, see [DeleteVpcEndpointServiceConfigurations](#) in *AWS CLI Command Reference*.

delete-vpc-endpoints

The following code example shows how to use `delete-vpc-endpoints`.

AWS CLI

To delete an endpoint

This example deletes endpoints `vpce-aa22bb33` and `vpce-1a2b3c4d`. If the command is partially successful or unsuccessful, a list of unsuccessful items is returned. If the command succeeds, the returned list is empty.

Command:

```
aws ec2 delete-vpc-endpoints --vpc-endpoint-ids vpce-aa22bb33 vpce-1a2b3c4d
```

Output:

```
{
  "Unsuccessful": []
}
```

- For API details, see [DeleteVpcEndpoints](#) in *AWS CLI Command Reference*.

delete-vpc-peering-connection

The following code example shows how to use `delete-vpc-peering-connection`.

AWS CLI

To delete a VPC peering connection

This example deletes the specified VPC peering connection.

Command:

```
aws ec2 delete-vpc-peering-connection --vpc-peering-connection-id pcx-1a2b3c4d
```

Output:

```
{
  "Return": true
}
```

- For API details, see [DeleteVpcPeeringConnection](#) in *AWS CLI Command Reference*.

delete-vpc

The following code example shows how to use `delete-vpc`.

AWS CLI

To delete a VPC

This example deletes the specified VPC. If the command succeeds, no output is returned.

Command:

```
aws ec2 delete-vpc --vpc-id vpc-a01106c2
```

- For API details, see [DeleteVpc](#) in *AWS CLI Command Reference*.

delete-vpn-connection-route

The following code example shows how to use `delete-vpn-connection-route`.

AWS CLI

To delete a static route from a VPN connection

This example deletes the specified static route from the specified VPN connection. If the command succeeds, no output is returned.

Command:

```
aws ec2 delete-vpn-connection-route --vpn-connection-id vpn-40f41529 --destination-cidr-block 11.12.0.0/16
```

- For API details, see [DeleteVpnConnectionRoute](#) in *AWS CLI Command Reference*.

delete-vpn-connection

The following code example shows how to use `delete-vpn-connection`.

AWS CLI

To delete a VPN connection

This example deletes the specified VPN connection. If the command succeeds, no output is returned.

Command:

```
aws ec2 delete-vpn-connection --vpn-connection-id vpn-40f41529
```

- For API details, see [DeleteVpnConnection](#) in *AWS CLI Command Reference*.

delete-vpn-gateway

The following code example shows how to use `delete-vpn-gateway`.

AWS CLI

To delete a virtual private gateway

This example deletes the specified virtual private gateway. If the command succeeds, no output is returned.

Command:

```
aws ec2 delete-vpn-gateway --vpn-gateway-id vgw-9a4cacf3
```

- For API details, see [DeleteVpnGateway](#) in *AWS CLI Command Reference*.

deprovision-byoip-cidr

The following code example shows how to use `deprovision-byoip-cidr`.

AWS CLI

To remove an IP address range from use

The following example removes the specified address range from use with AWS.

```
aws ec2 deprovision-byoip-cidr \  
  --cidr 203.0.113.25/24
```

Output:

```
{  
  "ByoipCidr": {  
    "Cidr": "203.0.113.25/24",  
    "State": "pending-deprovision"  
  }  
}
```

- For API details, see [DeprovisionByoipCidr](#) in *AWS CLI Command Reference*.

deprovision-ipam-pool-cidr

The following code example shows how to use `deprovision-ipam-pool-cidr`.

AWS CLI

To deprovision an IPAM pool CIDR

The following `deprovision-ipam-pool-cidr` example deprovisions a CIDR provisioned to an IPAM pool.

(Linux):

```
aws ec2 deprovision-ipam-pool-cidr \  
  --ipam-pool-id ipam-pool-02ec043a19bbe5d08 \  
  --cidr 11.0.0.0/16
```

(Windows):

```
aws ec2 deprovision-ipam-pool-cidr ^  
  --ipam-pool-id ipam-pool-02ec043a19bbe5d08 ^  
  --cidr 11.0.0.0/16
```

Output:

```
{  
  "IpamPoolCidr": {  
    "Cidr": "11.0.0.0/16",  
    "State": "pending-deprovision"  
  }  
}
```

For more information, see [Deprovision pool CIDRs](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [DeprovisionIpamPoolCidr](#) in *AWS CLI Command Reference*.

deregister-image

The following code example shows how to use `deregister-image`.

AWS CLI

To deregister an AMI

This example deregisters the specified AMI. If the command succeeds, no output is returned.

Command:

```
aws ec2 deregister-image --image-id ami-4fa54026
```

- For API details, see [DeregisterImage](#) in *AWS CLI Command Reference*.

deregister-instance-event-notification-attributes

The following code example shows how to use `deregister-instance-event-notification-attributes`.

AWS CLI**Example 1: To remove all tags from event notifications**

The following `deregister-instance-event-notification-attributes` example removes `IncludeAllTagsOfInstance=true`, which has the effect of setting `IncludeAllTagsOfInstance` to `false`.

```
aws ec2 deregister-instance-event-notification-attributes \  
  --instance-tag-attribute IncludeAllTagsOfInstance=true
```

Output:

```
{  
  "InstanceTagAttribute": {  
    "InstanceTagKeys": [],  
    "IncludeAllTagsOfInstance": true  
  }  
}
```

For more information, see [Scheduled events for your instances](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

Example 2: To remove specific tags from event notifications

The following `deregister-instance-event-notification-attributes` example removes the specified tag from the tags included in event notifications. To describe the remaining tags included in event notifications, use `describe-instance-event-notification-attributes`.


```
aws ec2 deregister-instance-event-notification-attributes \  
  --instance-tag-attribute InstanceTagKeys="tag-key2"
```

Output:

```
{  
  "InstanceTagAttribute": {  
    "InstanceTagKeys": [  
      "tag-key2"  
    ],  
    "IncludeAllTagsOfInstance": false  
  }  
}
```

For more information, see [Scheduled events for your instances](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [DeregisterInstanceEventNotificationAttributes](#) in *AWS CLI Command Reference*.

deregister-transit-gateway-multicast-group-members

The following code example shows how to use `deregister-transit-gateway-multicast-group-members`.

AWS CLI**To deregister group members from a multicast group**

This example deregisters the specified network interface group member from the transit gateway multicast group.

```
aws ec2 deregister-transit-gateway-multicast-group-members \  
  --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef7EXAMPLE \  
  --group-ip-address 224.0.1.0 \  
  --network-interface-ids eni-0e246d3269EXAMPLE
```

Output:

```
{
```

```

    "DeregisteredMulticastGroupMembers": {
      "TransitGatewayMulticastDomainId": "tgw-mcast-domain-0c4905cef7EXAMPLE",
      "RegisteredNetworkInterfaceIds": [
        "eni-0e246d3269EXAMPLE"
      ],
      "GroupIpAddress": "224.0.1.0"
    }
  }
}

```

For more information, see [Deregister Members from a Multicast Group](#) in the *AWS Transit Gateways Users Guide*.

- For API details, see [DeregisterTransitGatewayMulticastGroupMembers](#) in *AWS CLI Command Reference*.

deregister-transit-gateway-multicast-group-source

The following code example shows how to use `deregister-transit-gateway-multicast-group-source`.

AWS CLI

To deregister a source from the transit gateway multicast group

This example deregisters the specified network interface group source from the multicast group.

```

aws ec2 register-transit-gateway-multicast-group-sources \
  --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef79d6e597 \
  --group-ip-address 224.0.1.0 \
  --network-interface-ids eni-07f290fc3c090cbae

```

Output:

```

{
  "DeregisteredMulticastGroupSources": {
    "TransitGatewayMulticastDomainId": "tgw-mcast-domain-0c4905cef79d6e597",
    "DeregisteredNetworkInterfaceIds": [
      "eni-07f290fc3c090cbae"
    ],
    "GroupIpAddress": "224.0.1.0"
  }
}

```

```
}  
}
```

For more information, see [Deregister Sources from a Multicast Group](#) in the *AWS Transit Gateways User Guide*.

- For API details, see [DeregisterTransitGatewayMulticastGroupSource](#) in *AWS CLI Command Reference*.

describe-account-attributes

The following code example shows how to use `describe-account-attributes`.

AWS CLI

To describe all the attributes for your AWS account

This example describes the attributes for your AWS account.

Command:

```
aws ec2 describe-account-attributes
```

Output:

```
{  
  "AccountAttributes": [  
    {  
      "AttributeName": "vpc-max-security-groups-per-interface",  
      "AttributeValues": [  
        {  
          "AttributeValue": "5"  
        }  
      ]  
    },  
    {  
      "AttributeName": "max-instances",  
      "AttributeValues": [  
        {  
          "AttributeValue": "20"  
        }  
      ]  
    }  
  ]  
}
```

```
    },
    {
      "AttributeName": "supported-platforms",
      "AttributeValues": [
        {
          "AttributeValue": "EC2"
        },
        {
          "AttributeValue": "VPC"
        }
      ]
    },
    {
      "AttributeName": "default-vpc",
      "AttributeValues": [
        {
          "AttributeValue": "none"
        }
      ]
    },
    {
      "AttributeName": "max-elastic-ips",
      "AttributeValues": [
        {
          "AttributeValue": "5"
        }
      ]
    },
    {
      "AttributeName": "vpc-max-elastic-ips",
      "AttributeValues": [
        {
          "AttributeValue": "5"
        }
      ]
    }
  ]
}
```

To describe a single attribute for your AWS account

This example describes the supported-platforms attribute for your AWS account.

Command:

```
aws ec2 describe-account-attributes --attribute-names supported-platforms
```

Output:

```
{
  "AccountAttributes": [
    {
      "AttributeName": "supported-platforms",
      "AttributeValues": [
        {
          "AttributeValue": "EC2"
        },
        {
          "AttributeValue": "VPC"
        }
      ]
    }
  ]
}
```

- For API details, see [DescribeAccountAttributes](#) in *AWS CLI Command Reference*.

describe-address-transfers

The following code example shows how to use `describe-address-transfers`.

AWS CLI

To describe an Elastic IP address transfer

The following `describe-address-transfers` example describes the Elastic IP address transfer for the specified Elastic IP address.

```
aws ec2 describe-address-transfers \
  --allocation-ids eipalloc-09ad461b0d03f6aaf
```

Output:

```
{
  "AddressTransfers": [
    {
```

```
        "PublicIp": "100.21.184.216",
        "AllocationId": "eipalloc-09ad461b0d03f6aaf",
        "TransferAccountId": "123456789012",
        "TransferOfferExpirationTimestamp": "2023-02-22T22:51:01.000Z",
        "AddressTransferStatus": "pending"
    }
]
}
```

For more information, see [Transfer Elastic IP addresses](#) in the *Amazon VPC User Guide*.

- For API details, see [DescribeAddressTransfers](#) in *AWS CLI Command Reference*.

describe-addresses-attribute

The following code example shows how to use `describe-addresses-attribute`.

AWS CLI

To view the attributes of the domain name associated with an elastic IP address

The following `describe-addresses-attribute` examples return the attributes of the domain name associated with the elastic IP address.

Linux:

```
aws ec2 describe-addresses-attribute \
  --allocation-ids eipalloc-abcdef01234567890 \
  --attribute domain-name
```

Windows:

```
aws ec2 describe-addresses-attribute ^
  --allocation-ids eipalloc-abcdef01234567890 ^
  --attribute domain-name
```

Output:

```
{
  "Addresses": [
    {
      "PublicIp": "192.0.2.0",
```

```
        "AllocationId": "eipalloc-abcdef01234567890",
        "PtrRecord": "example.com."
    }
]
}
```

To view the attributes of an elastic IP address, you must have first associated a domain name with the elastic IP address. For more information, see [Use reverse DNS for email applications](#) in the *Amazon EC2 User Guide* or [modify-address-attribute](#) in the *AWS CLI Command Reference*.

- For API details, see [DescribeAddressesAttribute](#) in *AWS CLI Command Reference*.

describe-addresses

The following code example shows how to use `describe-addresses`.

AWS CLI

Example 1: To retrieve details about all of your Elastic IP addresses

The following `describe addresses` example displays details about your Elastic IP addresses.

```
aws ec2 describe-addresses
```

Output:

```
{
  "Addresses": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "PublicIp": "198.51.100.0",
      "PublicIpv4Pool": "amazon",
      "Domain": "standard"
    },
    {
      "Domain": "vpc",
      "PublicIpv4Pool": "amazon",
      "InstanceId": "i-1234567890abcdef0",
      "NetworkInterfaceId": "eni-12345678",
      "AssociationId": "eipassoc-12345678",
      "NetworkInterfaceOwnerId": "123456789012",
      "PublicIp": "203.0.113.0",
    }
  ]
}
```

```
        "AllocationId": "eipalloc-12345678",
        "PrivateIpAddress": "10.0.1.241"
    }
]
}
```

Example 2: To retrieve details your Elastic IP addresses for EC2-VPC

The following `describe-addresses` example displays details about your Elastic IP addresses for use with instances in a VPC.

```
aws ec2 describe-addresses \
  --filters "Name=domain,Values=vpc"
```

Output:

```
{
  "Addresses": [
    {
      "Domain": "vpc",
      "PublicIpv4Pool": "amazon",
      "InstanceId": "i-1234567890abcdef0",
      "NetworkInterfaceId": "eni-12345678",
      "AssociationId": "eipassoc-12345678",
      "NetworkInterfaceOwnerId": "123456789012",
      "PublicIp": "203.0.113.0",
      "AllocationId": "eipalloc-12345678",
      "PrivateIpAddress": "10.0.1.241"
    }
  ]
}
```

Example 3: To retrieve details about an Elastic IP address specified by allocation ID

The following `describe-addresses` example displays details about the Elastic IP address with the specified allocation ID, which is associated with an instance in EC2-VPC.

```
aws ec2 describe-addresses \
  --allocation-ids eipalloc-282d9641
```

Output:


```
{
  "Addresses": [
    {
      "Domain": "vpc",
      "PublicIpv4Pool": "amazon",
      "InstanceId": "i-1234567890abcdef0",
      "NetworkInterfaceId": "eni-1a2b3c4d",
      "AssociationId": "eipassoc-123abc12",
      "NetworkInterfaceOwnerId": "1234567891012",
      "PublicIp": "203.0.113.25",
      "AllocationId": "eipalloc-282d9641",
      "PrivateIpAddress": "10.251.50.12"
    }
  ]
}
```

Example 4: To retrieve details about an Elastic IP address specified by its VPC private IP address

The following describe-addresses example displays details about the Elastic IP address associated with a particular private IP address in EC2-VPC.

```
aws ec2 describe-addresses \
  --filters "Name=private-ip-address,Values=10.251.50.12"
```

Example 5: To retrieve details about Elastic IP addresses in EC2-Classic

The following describe-addresses example displays details about your Elastic IP addresses for use in EC2-Classic.

```
aws ec2 describe-addresses \
  --filters "Name=domain,Values=standard"
```

Output:

```
{
  "Addresses": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "PublicIp": "203.0.110.25",
      "PublicIpv4Pool": "amazon",

```

```

        "Domain": "standard"
    }
]
}

```

Example 6: To retrieve details about an Elastic IP addresses specified by its public IP address

The following `describe-addresses` example displays details about the Elastic IP address with the value `203.0.110.25`, which is associated with an instance in EC2-Classic.

```

aws ec2 describe-addresses \
  --public-ips 203.0.110.25

```

Output:

```

{
  "Addresses": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "PublicIp": "203.0.110.25",
      "PublicIpv4Pool": "amazon",
      "Domain": "standard"
    }
  ]
}

```

- For API details, see [DescribeAddresses](#) in *AWS CLI Command Reference*.

`describe-aggregate-id-format`

The following code example shows how to use `describe-aggregate-id-format`.

AWS CLI

To describe the longer ID format settings for all resource types in a Region

The following `describe-aggregate-id-format` example describes the overall long ID format status for the current Region. The `Deadline` value indicates that the deadlines for these resources to permanently switch from the short ID format to the long ID format expired. The `UseLongIdsAggregated` value indicates that all IAM users and IAM roles are configured to use long ID format for all resource types.

```
aws ec2 describe-aggregate-id-format
```

Output:

```
{
  "UseLongIdsAggregated": true,
  "Statuses": [
    {
      "Deadline": "2018-08-13T02:00:00.000Z",
      "Resource": "network-interface-attachment",
      "UseLongIds": true
    },
    {
      "Deadline": "2016-12-13T02:00:00.000Z",
      "Resource": "instance",
      "UseLongIds": true
    },
    {
      "Deadline": "2018-08-13T02:00:00.000Z",
      "Resource": "elastic-ip-association",
      "UseLongIds": true
    },
    ...
  ]
}
```

- For API details, see [DescribeAggregateIdFormat](#) in *AWS CLI Command Reference*.

describe-availability-zones

The following code example shows how to use `describe-availability-zones`.

AWS CLI

To describe your Availability Zones

The following example `describe-availability-zones` displays details for the Availability Zones that are available to you. The response includes Availability Zones only for the current Region. In this example, it uses the profiles default `us-west-2` (Oregon) Region.

```
aws ec2 describe-availability-zones
```

Output:

```
{
  "AvailabilityZones": [
    {
      "State": "available",
      "OptInStatus": "opt-in-not-required",
      "Messages": [],
      "RegionName": "us-west-2",
      "ZoneName": "us-west-2a",
      "ZoneId": "usw2-az1",
      "GroupName": "us-west-2",
      "NetworkBorderGroup": "us-west-2"
    },
    {
      "State": "available",
      "OptInStatus": "opt-in-not-required",
      "Messages": [],
      "RegionName": "us-west-2",
      "ZoneName": "us-west-2b",
      "ZoneId": "usw2-az2",
      "GroupName": "us-west-2",
      "NetworkBorderGroup": "us-west-2"
    },
    {
      "State": "available",
      "OptInStatus": "opt-in-not-required",
      "Messages": [],
      "RegionName": "us-west-2",
      "ZoneName": "us-west-2c",
      "ZoneId": "usw2-az3",
      "GroupName": "us-west-2",
      "NetworkBorderGroup": "us-west-2"
    },
    {
      "State": "available",
      "OptInStatus": "opt-in-not-required",
      "Messages": [],
      "RegionName": "us-west-2",
      "ZoneName": "us-west-2d",
      "ZoneId": "usw2-az4",
      "GroupName": "us-west-2",
      "NetworkBorderGroup": "us-west-2"
    },
  ],
}
```

```
    {
      "State": "available",
      "OptInStatus": "opted-in",
      "Messages": [],
      "RegionName": "us-west-2",
      "ZoneName": "us-west-2-lax-1a",
      "ZoneId": "usw2-lax1-az1",
      "GroupName": "us-west-2-lax-1",
      "NetworkBorderGroup": "us-west-2-lax-1"
    }
  ]
}
```

- For API details, see [DescribeAvailabilityZones](#) in *AWS CLI Command Reference*.

describe-aws-network-performance-metric-subscription

The following code example shows how to use `describe-aws-network-performance-metric-subscription`.

AWS CLI

To describe your metric subscriptions

The following `describe-aws-network-performance-metric-subscriptions` example describes your metric subscriptions.

```
aws ec2 describe-aws-network-performance-metric-subscriptions
```

Output:

```
{
  "Subscriptions": [
    {
      "Source": "us-east-1",
      "Destination": "eu-west-1",
      "Metric": "aggregate-latency",
      "Statistic": "p50",
      "Period": "five-minutes"
    }
  ]
}
```

For more information, see [Manage subscriptions](#) in the *Infrastructure Performance User Guide*.

- For API details, see [DescribeAwsNetworkPerformanceMetricSubscription](#) in *AWS CLI Command Reference*.

describe-aws-network-performance-metric-subscriptions

The following code example shows how to use `describe-aws-network-performance-metric-subscriptions`.

AWS CLI

To describe your metric subscriptions

The following `describe-aws-network-performance-metric-subscriptions` example describes your metric subscriptions.

```
aws ec2 describe-aws-network-performance-metric-subscriptions
```

Output:

```
{
  "Subscriptions": [
    {
      "Source": "us-east-1",
      "Destination": "eu-west-1",
      "Metric": "aggregate-latency",
      "Statistic": "p50",
      "Period": "five-minutes"
    }
  ]
}
```

For more information, see [Manage subscriptions](#) in the *Infrastructure Performance User Guide*.

- For API details, see [DescribeAwsNetworkPerformanceMetricSubscriptions](#) in *AWS CLI Command Reference*.

describe-bundle-tasks

The following code example shows how to use `describe-bundle-tasks`.

AWS CLI

To describe your bundle tasks

This example describes all of your bundle tasks.

Command:

```
aws ec2 describe-bundle-tasks
```

Output:

```
{
  "BundleTasks": [
    {
      "UpdateTime": "2015-09-15T13:26:54.000Z",
      "InstanceId": "i-1234567890abcdef0",
      "Storage": {
        "S3": {
          "Prefix": "winami",
          "Bucket": "bundletasks"
        }
      },
      "State": "bundling",
      "StartTime": "2015-09-15T13:24:35.000Z",
      "Progress": "3%",
      "BundleId": "bun-2a4e041c"
    }
  ]
}
```

- For API details, see [DescribeBundleTasks](#) in *AWS CLI Command Reference*.

describe-byoip-cidrs

The following code example shows how to use `describe-byoip-cidrs`.

AWS CLI

To describe your provisioned address ranges

The following `describe-byoip-cidrs` example displays details about the public IPv4 address ranges that you provisioned for use by AWS.

```
aws ec2 describe-byoip-cidrs
```

Output:

```
{
  "ByoipCidrs": [
    {
      "Cidr": "203.0.113.25/24",
      "StatusMessage": "ipv4pool-ec2-1234567890abcdef0",
      "State": "provisioned"
    }
  ]
}
```

- For API details, see [DescribeByoipCidrs](#) in *AWS CLI Command Reference*.

describe-capacity-reservation-fleets

The following code example shows how to use `describe-capacity-reservation-fleets`.

AWS CLI

To view a Capacity Reservation Fleet

The following `describe-capacity-reservation-fleets` example lists configuration and capacity information for the specified Capacity Reservation Fleet. It also lists details about the individual Capacity Reservations that are inside the Fleet.:

```
aws ec2 describe-capacity-reservation-fleets \
  --capacity-reservation-fleet-ids crf-abcdef01234567890
```

Output:

```
{
  "CapacityReservationFleets": [
    {
      "Status": "active",
      "EndDate": "2022-12-31T23:59:59.000Z",
      "InstanceMatchCriteria": "open",
      "Tags": [],
      "CapacityReservationFleetId": "crf-abcdef01234567890",

```



```
    "Tenancy": "default",
    "InstanceTypeSpecifications": [
      {
        "CapacityReservationId": "cr-1234567890abcdef0",
        "AvailabilityZone": "us-east-1a",
        "FulfilledCapacity": 5.0,
        "Weight": 1.0,
        "CreateDate": "2022-07-02T08:34:33.398Z",
        "InstancePlatform": "Linux/UNIX",
        "TotalInstanceCount": 5,
        "Priority": 1,
        "EbsOptimized": true,
        "InstanceType": "m5.xlarge"
      }
    ],
    "TotalTargetCapacity": 5,
    "TotalFulfilledCapacity": 5.0,
    "CreateTime": "2022-07-02T08:34:33.397Z",
    "AllocationStrategy": "prioritized"
  }
]
```

For more information about Capacity Reservation Fleets, see [Capacity Reservation Fleets](#) in the *Amazon EC2 User Guide*.

- For API details, see [DescribeCapacityReservationFleets](#) in *AWS CLI Command Reference*.

describe-capacity-reservations

The following code example shows how to use `describe-capacity-reservations`.

AWS CLI

Example 1: To describe one or more of your capacity reservations

The following `describe-capacity-reservations` example displays details about all of your capacity reservations in the current AWS Region.

```
aws ec2 describe-capacity-reservations
```

Output:

```
{
  "CapacityReservations": [
    {
      "CapacityReservationId": "cr-1234abcd56EXAMPLE ",
      "EndDateType": "unlimited",
      "AvailabilityZone": "eu-west-1a",
      "InstanceMatchCriteria": "open",
      "Tags": [],
      "EphemeralStorage": false,
      "CreateDate": "2019-08-16T09:03:18.000Z",
      "AvailableInstanceCount": 1,
      "InstancePlatform": "Linux/UNIX",
      "TotalInstanceCount": 1,
      "State": "active",
      "Tenancy": "default",
      "EbsOptimized": true,
      "InstanceType": "a1.medium"
    },
    {
      "CapacityReservationId": "cr-abcdEXAMPLE9876ef ",
      "EndDateType": "unlimited",
      "AvailabilityZone": "eu-west-1a",
      "InstanceMatchCriteria": "open",
      "Tags": [],
      "EphemeralStorage": false,
      "CreateDate": "2019-08-07T11:34:19.000Z",
      "AvailableInstanceCount": 3,
      "InstancePlatform": "Linux/UNIX",
      "TotalInstanceCount": 3,
      "State": "cancelled",
      "Tenancy": "default",
      "EbsOptimized": true,
      "InstanceType": "m5.large"
    }
  ]
}
```

Example 2: To describe one or more of your capacity reservations

The following describe-capacity-reservations example displays details about the specified capacity reservation.

```
aws ec2 describe-capacity-reservations \
```

```
--capacity-reservation-ids cr-1234abcd56EXAMPLE
```

Output:

```
{
  "CapacityReservations": [
    {
      "CapacityReservationId": "cr-1234abcd56EXAMPLE",
      "EndDateType": "unlimited",
      "AvailabilityZone": "eu-west-1a",
      "InstanceMatchCriteria": "open",
      "Tags": [],
      "EphemeralStorage": false,
      "CreateDate": "2019-08-16T09:03:18.000Z",
      "AvailableInstanceCount": 1,
      "InstancePlatform": "Linux/UNIX",
      "TotalInstanceCount": 1,
      "State": "active",
      "Tenancy": "default",
      "EbsOptimized": true,
      "InstanceType": "a1.medium"
    }
  ]
}
```

For more information, see [Viewing a Capacity Reservation](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [DescribeCapacityReservations](#) in *AWS CLI Command Reference*.

describe-carrier-gateways

The following code example shows how to use describe-carrier-gateways.

AWS CLI

To describe all carrier gateways

The following describe-carrier-gateways example lists all your carrier gateways.

```
aws ec2 describe-carrier-gateways
```

Output:

```
{
  "CarrierGateways": [
    {
      "CarrierGatewayId": "cagw-0465cdEXAMPLE1111",
      "VpcId": "vpc-0c529aEXAMPLE",
      "State": "available",
      "OwnerId": "123456789012",
      "Tags": [
        {
          "Key": "example",
          "Value": "tag"
        }
      ]
    }
  ]
}
```

For more information, see Carrier gateways<https://docs.aws.amazon.com/vpc/latest/userguide/Carrier_Gateway.html> in the *Amazon Virtual Private Cloud User Guide*.

- For API details, see [DescribeCarrierGateways](#) in *AWS CLI Command Reference*.

describe-classic-link-instances

The following code example shows how to use describe-classic-link-instances.

AWS CLI

To describe linked EC2-Classic instances

This example lists all of your linked EC2-Classic instances.

Command:

```
aws ec2 describe-classic-link-instances
```

Output:

```
{
  "Instances": [
    {
      "InstanceId": "i-1234567890abcdef0",
```

```

        "VpcId": "vpc-88888888",
        "Groups": [
            {
                "GroupId": "sg-11122233"
            }
        ],
        "Tags": [
            {
                "Value": "ClassicInstance",
                "Key": "Name"
            }
        ]
    },
    {
        "InstanceId": "i-0598c7d356eba48d7",
        "VpcId": "vpc-12312312",
        "Groups": [
            {
                "GroupId": "sg-aabbccdd"
            }
        ],
        "Tags": [
            {
                "Value": "ClassicInstance2",
                "Key": "Name"
            }
        ]
    }
]
}

```

This example lists all of your linked EC2-Classic instances, and filters the response to include only instances that are linked to VPC vpc-88888888.

Command:

```
aws ec2 describe-classic-link-instances --filter "Name=vpc-id,Values=vpc-88888888"
```

Output:

```

{
    "Instances": [
        {

```

```

    "InstanceId": "i-1234567890abcdef0",
    "VpcId": "vpc-88888888",
    "Groups": [
      {
        "GroupId": "sg-11122233"
      }
    ],
    "Tags": [
      {
        "Value": "ClassicInstance",
        "Key": "Name"
      }
    ]
  }
]
}

```

- For API details, see [DescribeClassicLinkInstances](#) in *AWS CLI Command Reference*.

describe-client-vpn-authorization-rules

The following code example shows how to use `describe-client-vpn-authorization-rules`.

AWS CLI

To describe the authorization rules for a Client VPN endpoint

The following `describe-client-vpn-authorization-rules` example displays details about the authorization rules for the specified Client VPN endpoint.

```
aws ec2 describe-client-vpn-authorization-rules \
  --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde
```

Output:

```

{
  "AuthorizationRules": [
    {
      "ClientVpnEndpointId": "cvpn-endpoint-123456789123abcde",
      "GroupId": "",
      "AccessAll": true,
      "DestinationCidr": "0.0.0.0/0",
    }
  ]
}

```

```
        "Status": {
            "Code": "active"
        }
    ]
}
```

For more information, see [Authorization Rules](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [DescribeClientVpnAuthorizationRules](#) in *AWS CLI Command Reference*.

describe-client-vpn-connections

The following code example shows how to use `describe-client-vpn-connections`.

AWS CLI

To describe the connections to a Client VPN endpoint

The following `describe-client-vpn-connections` example displays details about the client connections to the specified Client VPN endpoint.

```
aws ec2 describe-client-vpn-connections \
    --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde
```

Output:

```
{
  "Connections": [
    {
      "ClientVpnEndpointId": "cvpn-endpoint-123456789123abcde",
      "Timestamp": "2019-08-12 07:58:34",
      "ConnectionId": "cvpn-connection-0e03eb24267165acd",
      "ConnectionEstablishedTime": "2019-08-12 07:57:14",
      "IngressBytes": "32302",
      "EgressBytes": "5696",
      "IngressPackets": "332",
      "EgressPackets": "67",
      "ClientIp": "172.31.0.225",
      "CommonName": "client1.domain.tld",
      "Status": {
        "Code": "terminated"
      }
    },
  ],
}
```

```

        "ConnectionEndTime": "2019-08-12 07:58:34"
    },
    {
        "ClientVpnEndpointId": "cvpn-endpoint-123456789123abcde",
        "Timestamp": "2019-08-12 08:02:54",
        "ConnectionId": "cvpn-connection-00668867a40f18253",
        "ConnectionEstablishedTime": "2019-08-12 08:02:53",
        "IngressBytes": "2951",
        "EgressBytes": "2611",
        "IngressPackets": "9",
        "EgressPackets": "6",
        "ClientIp": "172.31.0.226",
        "CommonName": "client1.domain.tld",
        "Status": {
            "Code": "active"
        },
        "ConnectionEndTime": "-"
    }
]
}

```

For more information, see [Client Connections](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [DescribeClientVpnConnections](#) in *AWS CLI Command Reference*.

describe-client-vpn-endpoints

The following code example shows how to use `describe-client-vpn-endpoints`.

AWS CLI

To describe your Client VPN endpoints

The following `describe-client-vpn-endpoints` example displays details about all of your Client VPN endpoints.

```
aws ec2 describe-client-vpn-endpoints
```

Output:

```
{
  "ClientVpnEndpoints": [
    {
```



```
"ClientVpnEndpointId": "cvpn-endpoint-123456789123abcde",
"Description": "Endpoint for Admin access",
"Status": {
  "Code": "available"
},
"CreationTime": "2020-11-13T11:37:27",
"DnsName": "*.cvpn-endpoint-123456789123abcde.prod.clientvpn.ap-
south-1.amazonaws.com",
"ClientCidrBlock": "172.31.0.0/16",
"DnsServers": [
  "8.8.8.8"
],
"SplitTunnel": false,
"VpnProtocol": "openvpn",
"TransportProtocol": "udp",
"VpnPort": 443,
"ServerCertificateArn": "arn:aws:acm:ap-
south-1:123456789012:certificate/a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
"AuthenticationOptions": [
  {
    "Type": "certificate-authentication",
    "MutualAuthentication": {
      "ClientRootCertificateChain": "arn:aws:acm:ap-
south-1:123456789012:certificate/a1b2c3d4-5678-90ab-cdef-22222EXAMPLE"
    }
  }
],
"ConnectionLogOptions": {
  "Enabled": true,
  "CloudwatchLogGroup": "Client-vpn-connection-logs",
  "CloudwatchLogStream": "cvpn-endpoint-123456789123abcde-ap-
south-1-2020/11/13-FCD8HEMVAccw"
},
"Tags": [
  {
    "Key": "Name",
    "Value": "Client VPN"
  }
],
"SecurityGroupIds": [
  "sg-aabbcc11223344567"
],
"VpcId": "vpc-a87f92c1",
```

```

        "SelfServicePortalUrl": "https://self-service.clientvpn.amazonaws.com/
endpoints/cvpn-endpoint-123456789123abcde",
        "ClientConnectOptions": {
            "Enabled": false
        }
    }
]
}

```

For more information, see [Client VPN Endpoints](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [DescribeClientVpnEndpoints](#) in *AWS CLI Command Reference*.

describe-client-vpn-routes

The following code example shows how to use `describe-client-vpn-routes`.

AWS CLI

To describe the routes for a Client VPN endpoint

The following `describe-client-vpn-routes` example displays details about the routes for the specified Client VPN endpoint.

```

aws ec2 describe-client-vpn-routes \
  --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde

```

Output:

```

{
  "Routes": [
    {
      "ClientVpnEndpointId": "cvpn-endpoint-123456789123abcde",
      "DestinationCidr": "10.0.0.0/16",
      "TargetSubnet": "subnet-0123456789abcabca",
      "Type": "Nat",
      "Origin": "associate",
      "Status": {
        "Code": "active"
      },
      "Description": "Default Route"
    },
  ],
}

```

```

    {
      "ClientVpnEndpointId": "cvpn-endpoint-123456789123abcde",
      "DestinationCidr": "0.0.0.0/0",
      "TargetSubnet": "subnet-0123456789abcabca",
      "Type": "Nat",
      "Origin": "add-route",
      "Status": {
        "Code": "active"
      }
    }
  ]
}

```

For more information, see [Routes](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [DescribeClientVpnRoutes](#) in *AWS CLI Command Reference*.

describe-client-vpn-target-networks

The following code example shows how to use `describe-client-vpn-target-networks`.

AWS CLI

To describe the target networks for a Client VPN endpoint

The following `describe-client-vpn-target-networks` example displays details about the target networks for the specified Client VPN endpoint.

```

aws ec2 describe-client-vpn-target-networks \
  --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde

```

Output:

```

{
  "ClientVpnTargetNetworks": [
    {
      "AssociationId": "cvpn-assoc-012e837060753dc3d",
      "VpcId": "vpc-11111222222333333",
      "TargetNetworkId": "subnet-0123456789abcabca",
      "ClientVpnEndpointId": "cvpn-endpoint-123456789123abcde",
      "Status": {
        "Code": "associating"
      }
    },
  ],
}

```

```

        "SecurityGroups": [
            "sg-012345678910abcab"
        ]
    }
]
}

```

For more information, see [Target Networks](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [DescribeClientVpnTargetNetworks](#) in *AWS CLI Command Reference*.

describe-coip-pools

The following code example shows how to use `describe-coip-pools`.

AWS CLI

To describe customer-owned IP address pools

The following `describe-coip-pools` example describes the customer-owned IP address pools in your AWS account.

```
aws ec2 describe-coip-pools
```

Output:

```

{
  "CoipPools": [
    {
      "PoolId": "ipv4pool-coip-123a45678bEXAMPLE",
      "PoolCidrs": [
        "0.0.0.0/0"
      ],
      "LocalGatewayRouteTableId": "lgw-rtb-059615ef7dEXAMPLE",
      "PoolArn": "arn:aws:ec2:us-west-2:123456789012:coip-pool/ipv4pool-coip-123a45678bEXAMPLE"
    }
  ]
}

```

For more information, see [Customer-owned IP addresses](#) in the *AWS Outposts User Guide*.

- For API details, see [DescribeCoipPools](#) in *AWS CLI Command Reference*.

describe-conversion-tasks

The following code example shows how to use describe-conversion-tasks.

AWS CLI

To view the status of a conversion task

This example returns the status of a conversion task with the ID import-i-ffvko9js.

Command:

```
aws ec2 describe-conversion-tasks --conversion-task-ids import-i-ffvko9js
```

Output:

```
{
  "ConversionTasks": [
    {
      "ConversionTaskId": "import-i-ffvko9js",
      "ImportInstance": {
        "InstanceId": "i-1234567890abcdef0",
        "Volumes": [
          {
            "Volume": {
              "Id": "vol-049df61146c4d7901",
              "Size": 16
            },
            "Status": "completed",
            "Image": {
              "Size": 1300687360,
              "ImportManifestUrl": "https://s3.amazonaws.com/myimportbucket/411443cd-d620-4f1c-9d66-13144EXAMPLE/RHEL5.vmdkmanifest.xml?AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&Expires=140EXAMPLE&Signature=XYNhznHNgCqsjDxL9wRL%2FJvEXAMPLE",
              "Format": "VMDK"
            },
            "BytesConverted": 1300682960,
            "AvailabilityZone": "us-east-1d"
          }
        ]
      },
      "ExpirationTime": "2014-05-14T22:06:23Z",
      "State": "completed"
    }
  ]
}
```

```
    }  
  ]  
}
```

- For API details, see [DescribeConversionTasks](#) in *AWS CLI Command Reference*.

describe-customer-gateways

The following code example shows how to use `describe-customer-gateways`.

AWS CLI

To describe your customer gateways

This example describes your customer gateways.

Command:

```
aws ec2 describe-customer-gateways
```

Output:

```
{  
  "CustomerGateways": [  
    {  
      "CustomerGatewayId": "cgw-b4dc3961",  
      "IpAddress": "203.0.113.12",  
      "State": "available",  
      "Type": "ipsec.1",  
      "BgpAsn": "65000"  
    },  
    {  
      "CustomerGatewayId": "cgw-0e11f167",  
      "IpAddress": "12.1.2.3",  
      "State": "available",  
      "Type": "ipsec.1",  
      "BgpAsn": "65534"  
    }  
  ]  
}
```

To describe a specific customer gateway

This example describes the specified customer gateway.

Command:

```
aws ec2 describe-customer-gateways --customer-gateway-ids cgw-0e11f167
```

Output:

```
{
  "CustomerGateways": [
    {
      "CustomerGatewayId": "cgw-0e11f167",
      "IpAddress": "12.1.2.3",
      "State": "available",
      "Type": "ipsec.1",
      "BgpAsn": "65534"
    }
  ]
}
```

- For API details, see [DescribeCustomerGateways](#) in *AWS CLI Command Reference*.

describe-dhcp-options

The following code example shows how to use describe-dhcp-options.

AWS CLI

Example 1: To describe your DHCP options

The following describe-dhcp-options example retrieves details about your DHCP options.

```
aws ec2 describe-dhcp-options
```

Output:

```
{
  "DhcpOptions": [
    {
      "DhcpConfigurations": [
        {
          "Key": "domain-name",
```

```
        "Values": [
            {
                "Value": "us-east-2.compute.internal"
            }
        ]
    },
    {
        "Key": "domain-name-servers",
        "Values": [
            {
                "Value": "AmazonProvidedDNS"
            }
        ]
    }
],
"DhcpOptionsId": "dopt-19edf471",
"OwnerId": "111122223333"
},
{
    "DhcpConfigurations": [
        {
            "Key": "domain-name",
            "Values": [
                {
                    "Value": "us-east-2.compute.internal"
                }
            ]
        },
        {
            "Key": "domain-name-servers",
            "Values": [
                {
                    "Value": "AmazonProvidedDNS"
                }
            ]
        }
    ]
},
"DhcpOptionsId": "dopt-fEXAMPLE",
"OwnerId": "111122223333"
}
]
}
```


For more information, see [Working with DHCP Option Sets](#) in the *AWS VPC User Guide*.

Example 2: To describe your DHCP options and filter the output

The following `describe-dhcp-options` example describes your DHCP options and uses a filter to return only DHCP options that have `example.com` for the domain name server. The example uses the `--query` parameter to display only the configuration information and ID in the output.

```
aws ec2 describe-dhcp-options \
  --filters Name=key,Values=domain-name-servers Name=value,Values=example.com \
  --query "DhcpOptions[*].[DhcpConfigurations,DhcpOptionsId]"
```

Output:

```
[
  [
    [
      {
        "Key": "domain-name",
        "Values": [
          {
            "Value": "example.com"
          }
        ]
      },
      {
        "Key": "domain-name-servers",
        "Values": [
          {
            "Value": "172.16.16.16"
          }
        ]
      }
    ],
    "dopt-001122334455667ab"
  ]
]
```

For more information, see [Working with DHCP Option Sets](#) in the *AWS VPC User Guide*.

- For API details, see [DescribeDhcpOptions](#) in *AWS CLI Command Reference*.

describe-egress-only-internet-gateways

The following code example shows how to use `describe-egress-only-internet-gateways`.

AWS CLI

To describe your egress-only Internet gateways

This example describes your egress-only Internet gateways.

Command:

```
aws ec2 describe-egress-only-internet-gateways
```

Output:

```
{
  "EgressOnlyInternetGateways": [
    {
      "EgressOnlyInternetGatewayId": "eigw-015e0e244e24dfe8a",
      "Attachments": [
        {
          "State": "attached",
          "VpcId": "vpc-0c62a468"
        }
      ]
    }
  ]
}
```

- For API details, see [DescribeEgressOnlyInternetGateways](#) in *AWS CLI Command Reference*.

describe-elastic-gpus

The following code example shows how to use `describe-elastic-gpus`.

AWS CLI

To describe an Elastic GPU

Command:

```
aws ec2 describe-elastic-gpus --elastic-gpu-ids
egpu-12345678901234567890abcdefghijkl
```

- For API details, see [DescribeElasticGpus](#) in *AWS CLI Command Reference*.

describe-export-image-tasks

The following code example shows how to use `describe-export-image-tasks`.

AWS CLI

To monitor an export image task

The following `describe-export-image-tasks` example checks the status of the specified export image task. The resulting image file in Amazon S3 is `my-export-bucket/exports/export-ami-1234567890abcdef0.vmdk`.

```
aws ec2 describe-export-image-tasks \
--export-image-task-ids export-ami-1234567890abcdef0
```

Output for an export image task that is in progress.

```
{
  "ExportImageTasks": [
    {
      "ExportImageTaskId": "export-ami-1234567890abcdef0"
      "Progress": "21",
      "S3ExportLocation": {
        "S3Bucket": "my-export-bucket",
        "S3Prefix": "exports/"
      },
      "Status": "active",
      "StatusMessage": "updating"
    }
  ]
}
```

Output for an export image task that is completed.

```
{
  "ExportImageTasks": [
```

```
{
  "ExportImageTaskId": "export-ami-1234567890abcdef0"
  "S3ExportLocation": {
    "S3Bucket": "my-export-bucket",
    "S3Prefix": "exports/"
  },
  "Status": "completed"
}
]
```

For more information, see [Export a VM from an AMI](#) in the *VM Import/Export User Guide*.

- For API details, see [DescribeExportImageTasks](#) in *AWS CLI Command Reference*.

describe-export-tasks

The following code example shows how to use describe-export-tasks.

AWS CLI

To list details about an instance export task

This example describes the export task with ID export-i-fh8sjjsq.

Command:

```
aws ec2 describe-export-tasks --export-task-ids export-i-fh8sjjsq
```

Output:

```
{
  "ExportTasks": [
    {
      "State": "active",
      "InstanceExportDetails": {
        "InstanceId": "i-1234567890abcdef0",
        "TargetEnvironment": "vmware"
      },
      "ExportToS3Task": {
        "S3Bucket": "myexportbucket",
        "S3Key": "RHEL5export-i-fh8sjjsq.ova",
        "DiskImageFormat": "vmdk",

```

```
        "ContainerFormat": "ova"
      },
      "Description": "RHEL5 instance",
      "ExportTaskId": "export-i-fh8sjjsq"
    }
  ]
}
```

- For API details, see [DescribeExportTasks](#) in *AWS CLI Command Reference*.

describe-fast-launch-images

The following code example shows how to use `describe-fast-launch-images`.

AWS CLI

To describe the details for Windows AMIs that are configured for faster launching

The following `describe-fast-launch-images` example describes the details for each of the AMIs in your account that are configured for faster launching, including the resource type, the snapshot configuration, the launch template details, the maximum number of parallel launches, the AMI owner ID, the state of the fast launch configuration, the reason the state was changed, and the time that the state change occurred.

```
aws ec2 describe-fast-launch-images
```

Output:

```
{
  "FastLaunchImages": [
    {
      "ImageId": "ami-01234567890abcdef",
      "ResourceType": "snapshot",
      "SnapshotConfiguration": {},
      "LaunchTemplate": {
        "LaunchTemplateId": "lt-01234567890abcdef",
        "LaunchTemplateName": "EC2FastLaunchDefaultResourceCreation-
a8c6215d-94e6-441b-9272-dbd1f87b07e2",
        "Version": "1"
      },
      "MaxParallelLaunches": 6,
      "OwnerId": "0123456789123",
    }
  ]
}
```

```

        "State": "enabled",
        "StateTransitionReason": "Client.UserInitiated",
        "StateTransitionTime": "2022-01-27T22:20:06.552000+00:00"
    }
]
}

```

For more information about configuring a Windows AMI for faster launching, see [Configure your AMI for faster launching](#) in the *Amazon EC2 User Guide*.

- For API details, see [DescribeFastLaunchImages](#) in *AWS CLI Command Reference*.

describe-fast-snapshot-restores

The following code example shows how to use `describe-fast-snapshot-restores`.

AWS CLI

To describe fast snapshot restores

The following `describe-fast-snapshot-restores` example displays details for all fast snapshot restores with a state of disabled.

```
aws ec2 describe-fast-snapshot-restores \
  --filters Name=state,Values=disabled
```

Output:

```

{
  "FastSnapshotRestores": [
    {
      "SnapshotId": "snap-1234567890abcdef0",
      "AvailabilityZone": "us-west-2c",
      "State": "disabled",
      "StateTransitionReason": "Client.UserInitiated - Lifecycle state
transition",
      "OwnerId": "123456789012",
      "EnablingTime": "2020-01-25T23:57:49.596Z",
      "OptimizingTime": "2020-01-25T23:58:25.573Z",
      "EnabledTime": "2020-01-25T23:59:29.852Z",
      "DisablingTime": "2020-01-26T00:40:56.069Z",
      "DisabledTime": "2020-01-26T00:41:27.390Z"
    }
  ]
}

```

```
    }
  ]
}
```

The following `describe-fast-snapshot-restores` example describes all fast snapshot restores.

```
aws ec2 describe-fast-snapshot-restores
```

- For API details, see [DescribeFastSnapshotRestores](#) in *AWS CLI Command Reference*.

describe-fleet-history

The following code example shows how to use `describe-fleet-history`.

AWS CLI

To describe EC2 Fleet history

The following `describe-fleet-history` example returns the history for the specified EC2 Fleet starting at the specified time. The output is for an EC2 Fleet with two running instances.

```
aws ec2 describe-fleet-history \
  --fleet-id fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE \
  --start-time 2020-09-01T00:00:00Z
```

Output:

```
{
  "HistoryRecords": [
    {
      "EventInformation": {
        "EventSubType": "submitted"
      },
      "EventType": "fleetRequestChange",
      "Timestamp": "2020-09-01T18:26:05.000Z"
    },
    {
      "EventInformation": {
        "EventSubType": "active"
      },
    }
  ]
}
```

```

    "EventType": "fleetRequestChange",
    "Timestamp": "2020-09-01T18:26:15.000Z"
  },
  {
    "EventInformation": {
      "EventDescription": "t2.small, ami-07c8bc5c1ce9598c3, ...",
      "EventSubType": "progress"
    },
    "EventType": "fleetRequestChange",
    "Timestamp": "2020-09-01T18:26:17.000Z"
  },
  {
    "EventInformation": {
      "EventDescription": "{\"instanceType\": \"t2.small\", ...}",
      "EventSubType": "launched",
      "InstanceId": "i-083a1c446e66085d2"
    },
    "EventType": "instanceChange",
    "Timestamp": "2020-09-01T18:26:17.000Z"
  },
  {
    "EventInformation": {
      "EventDescription": "{\"instanceType\": \"t2.small\", ...}",
      "EventSubType": "launched",
      "InstanceId": "i-090db02406cc3c2d6"
    },
    "EventType": "instanceChange",
    "Timestamp": "2020-09-01T18:26:17.000Z"
  }
],
"LastEvaluatedTime": "2020-09-01T19:10:19.000Z",
"FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE",
"StartTime": "2020-08-31T23:53:20.000Z"
}

```

For more information, see [Managing an EC2 Fleet](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [DescribeFleetHistory](#) in *AWS CLI Command Reference*.

describe-fleet-instances

The following code example shows how to use `describe-fleet-instances`.

AWS CLI

To describe the running instances for an EC2 Fleet

The following `describe-fleet-instances` example describes the running instances for the specified EC2 Fleet.

```
aws ec2 describe-fleet-instances \
  --fleet-id 12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE
```

Output:

```
{
  "ActiveInstances": [
    {
      "InstanceId": "i-090db02406cc3c2d6",
      "InstanceType": "t2.small",
      "SpotInstanceRequestId": "sir-a43gtpfk",
      "InstanceHealth": "healthy"
    },
    {
      "InstanceId": "i-083a1c446e66085d2",
      "InstanceType": "t2.small",
      "SpotInstanceRequestId": "sir-iwcit2nj",
      "InstanceHealth": "healthy"
    }
  ],
  "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE"
}
```

For more information, see [Managing an EC2 Fleet](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [DescribeFleetInstances](#) in *AWS CLI Command Reference*.

describe-fleets

The following code example shows how to use `describe-fleets`.

AWS CLI

To describe an EC2 Fleet

The following describe-fleets example describes the specified EC2 Fleet.

```
aws ec2 describe-fleets \  
  --fleet-ids fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE
```

Output:

```
{  
  "Fleets": [  
    {  
      "ActivityStatus": "pending_fulfillment",  
      "CreateTime": "2020-09-01T18:26:05.000Z",  
      "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE",  
      "FleetState": "active",  
      "ExcessCapacityTerminationPolicy": "termination",  
      "FulfilledCapacity": 0.0,  
      "FulfilledOnDemandCapacity": 0.0,  
      "LaunchTemplateConfigs": [  
        {  
          "LaunchTemplateSpecification": {  
            "LaunchTemplateId": "lt-0e632f2855a979cd5",  
            "Version": "1"  
          }  
        }  
      ],  
      "TargetCapacitySpecification": {  
        "TotalTargetCapacity": 2,  
        "OnDemandTargetCapacity": 0,  
        "SpotTargetCapacity": 2,  
        "DefaultTargetCapacityType": "spot"  
      },  
      "TerminateInstancesWithExpiration": false,  
      "Type": "maintain",  
      "ReplaceUnhealthyInstances": false,  
      "SpotOptions": {  
        "AllocationStrategy": "lowestPrice",  
        "InstanceInterruptionBehavior": "terminate",  
        "InstancePoolsToUseCount": 1  
      },  
      "OnDemandOptions": {  
        "AllocationStrategy": "lowestPrice"  
      }  
    }  
  ]  
}
```

```
]
}
```

For more information, see [Managing an EC2 Fleet](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [DescribeFleets](#) in *AWS CLI Command Reference*.

describe-flow-logs

The following code example shows how to use `describe-flow-logs`.

AWS CLI

Example 1: To describe all of your flow logs

The following `describe-flow-logs` example displays details for all of your flow logs.

```
aws ec2 describe-flow-logs
```

Output:

```
{
  "FlowLogs": [
    {
      "CreationTime": "2018-02-21T13:22:12.644Z",
      "DeliverLogsPermissionArn": "arn:aws:iam::123456789012:role/flow-logs-
role",
      "DeliverLogsStatus": "SUCCESS",
      "FlowLogId": "fl-aabbccdd112233445",
      "MaxAggregationInterval": 600,
      "FlowLogStatus": "ACTIVE",
      "LogGroupName": "FlowLogGroup",
      "ResourceId": "subnet-12345678901234567",
      "TrafficType": "ALL",
      "LogDestinationType": "cloud-watch-logs",
      "LogFormat": "${version} ${account-id} ${interface-id} ${srcaddr}
${dstaddr} ${srcport} ${dstport} ${protocol} ${packets} ${bytes} ${start} ${end}
${action} ${log-status}"
    },
    {
      "CreationTime": "2020-02-04T15:22:29.986Z",
```

```

    "DeliverLogsStatus": "SUCCESS",
    "FlowLogId": "fl-01234567890123456",
    "MaxAggregationInterval": 60,
    "FlowLogStatus": "ACTIVE",
    "ResourceId": "vpc-00112233445566778",
    "TrafficType": "ACCEPT",
    "LogDestinationType": "s3",
    "LogDestination": "arn:aws:s3:::my-flow-log-bucket/custom",
    "LogFormat": "${version} ${vpc-id} ${subnet-id} ${instance-id}
    ${interface-id} ${account-id} ${type} ${srcaddr} ${dstaddr} ${srcport} ${dstport}
    ${pkt-srcaddr} ${pkt-dstaddr} ${protocol} ${bytes} ${packets} ${start} ${end}
    ${action} ${tcp-flags} ${log-status}"
  }
]
}

```

Example 2: To describe a subset of your flow logs

The following `describe-flow-logs` example uses a filter to display details for only those flow logs that are in the specified log group in Amazon CloudWatch Logs.

```

aws ec2 describe-flow-logs \
  --filter "Name=log-group-name,Values=MyFlowLogs"

```

- For API details, see [DescribeFlowLogs](#) in *AWS CLI Command Reference*.

`describe-fpga-image-attribute`

The following code example shows how to use `describe-fpga-image-attribute`.

AWS CLI

To describe the attributes of an Amazon FPGA image

This example describes the load permissions for the specified AFI.

Command:

```

aws ec2 describe-fpga-image-attribute --fpga-image-id afi-0d123e123bfc85abc --
attribute loadPermission

```

Output:

```
{
  "FpgaImageAttribute": {
    "FpgaImageId": "afi-0d123e123bfc85abc",
    "LoadPermissions": [
      {
        "UserId": "123456789012"
      }
    ]
  }
}
```

- For API details, see [DescribeFpgaImageAttribute](#) in *AWS CLI Command Reference*.

describe-fpga-images

The following code example shows how to use `describe-fpga-images`.

AWS CLI

To describe Amazon FPGA images

This example describes AFIs that are owned by account 123456789012.

Command:

```
aws ec2 describe-fpga-images --filters Name=owner-id,Values=123456789012
```

Output:

```
{
  "FpgaImages": [
    {
      "UpdateTime": "2017-12-22T12:09:14.000Z",
      "Name": "my-afi",
      "PciId": {
        "SubsystemVendorId": "0xfedd",
        "VendorId": "0x1d0f",
        "DeviceId": "0xf000",
        "SubsystemId": "0x1d51"
      },
      "FpgaImageGlobalId": "agfi-123cb27b5e84a0abc",
    }
  ]
}
```

```
    "Public": false,
    "State": {
      "Code": "available"
    },
    "ShellVersion": "0x071417d3",
    "OwnerId": "123456789012",
    "FpgaImageId": "afi-0d123e123bfc85abc",
    "CreateTime": "2017-12-22T11:43:33.000Z",
    "Description": "my-afi"
  }
]
}
```

- For API details, see [DescribeFpgaImages](#) in *AWS CLI Command Reference*.

describe-host-reservation-offerings

The following code example shows how to use `describe-host-reservation-offerings`.

AWS CLI

To describe Dedicated Host Reservation offerings

This example describes the Dedicated Host Reservations for the M4 instance family that are available to purchase.

Command:

```
aws ec2 describe-host-reservation-offerings --filter Name=instance-family,Values=m4
```

Output:

```
{
  "OfferingSet": [
    {
      "HourlyPrice": "1.499",
      "OfferingId": "hro-03f707bf363b6b324",
      "InstanceFamily": "m4",
      "PaymentOption": "NoUpfront",
      "UpfrontPrice": "0.000",
      "Duration": 31536000
    },
  ],
}
```

```
{
  "HourlyPrice": "1.045",
  "OfferingId": "hro-0ef9181cabdef7a02",
  "InstanceFamily": "m4",
  "PaymentOption": "NoUpfront",
  "UpfrontPrice": "0.000",
  "Duration": 94608000
},
{
  "HourlyPrice": "0.714",
  "OfferingId": "hro-04567a15500b92a51",
  "InstanceFamily": "m4",
  "PaymentOption": "PartialUpfront",
  "UpfrontPrice": "6254.000",
  "Duration": 31536000
},
{
  "HourlyPrice": "0.484",
  "OfferingId": "hro-0d5d7a9d23ed7fbfe",
  "InstanceFamily": "m4",
  "PaymentOption": "PartialUpfront",
  "UpfrontPrice": "12720.000",
  "Duration": 94608000
},
{
  "HourlyPrice": "0.000",
  "OfferingId": "hro-05da4108ca998c2e5",
  "InstanceFamily": "m4",
  "PaymentOption": "AllUpfront",
  "UpfrontPrice": "23913.000",
  "Duration": 94608000
},
{
  "HourlyPrice": "0.000",
  "OfferingId": "hro-0a9f9be3b95a3dc8f",
  "InstanceFamily": "m4",
  "PaymentOption": "AllUpfront",
  "UpfrontPrice": "12257.000",
  "Duration": 31536000
}
]
```

- For API details, see [DescribeHostReservationOfferings](#) in *AWS CLI Command Reference*.

describe-host-reservations

The following code example shows how to use `describe-host-reservations`.

AWS CLI

To describe Dedicated Host Reservations in your account

This example describes the Dedicated Host Reservations in your account.

Command:

```
aws ec2 describe-host-reservations
```

Output:

```
{
  "HostReservationSet": [
    {
      "Count": 1,
      "End": "2019-01-10T12:14:09Z",
      "HourlyPrice": "1.499",
      "InstanceFamily": "m4",
      "OfferingId": "hro-03f707bf363b6b324",
      "PaymentOption": "NoUpfront",
      "State": "active",
      "HostIdSet": [
        "h-013abcd2a00cbd123"
      ],
      "Start": "2018-01-10T12:14:09Z",
      "HostReservationId": "hr-0d418a3a4ffc669ae",
      "UpfrontPrice": "0.000",
      "Duration": 31536000
    }
  ]
}
```

- For API details, see [DescribeHostReservations](#) in *AWS CLI Command Reference*.

describe-hosts

The following code example shows how to use `describe-hosts`.

AWS CLI

To view details about Dedicated Hosts

The following describe-hosts example displays details for the available Dedicated Hosts in your AWS account.

```
aws ec2 describe-hosts --filter "Name=state,Values=available"
```

Output:

```
{
  "Hosts": [
    {
      "HostId": "h-07879acf49EXAMPLE",
      "Tags": [
        {
          "Value": "production",
          "Key": "purpose"
        }
      ],
      "HostProperties": {
        "Cores": 48,
        "TotalVCpus": 96,
        "InstanceType": "m5.large",
        "Sockets": 2
      },
      "Instances": [],
      "State": "available",
      "AvailabilityZone": "eu-west-1a",
      "AvailableCapacity": {
        "AvailableInstanceCapacity": [
          {
            "AvailableCapacity": 48,
            "InstanceType": "m5.large",
            "TotalCapacity": 48
          }
        ],
        "AvailableVCpus": 96
      },
      "HostRecovery": "on",
      "AllocationTime": "2019-08-19T08:57:44.000Z",
      "AutoPlacement": "off"
    }
  ]
}
```

```
    }  
  ]  
}
```

For more information, see [Viewing Dedicated Hosts](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [DescribeHosts](#) in *AWS CLI Command Reference*.

describe-iam-instance-profile-associations

The following code example shows how to use `describe-iam-instance-profile-associations`.

AWS CLI

To describe IAM instance profile associations

This example describes all of your IAM instance profile associations.

Command:

```
aws ec2 describe-iam-instance-profile-associations
```

Output:

```
{  
  "IamInstanceProfileAssociations": [  
    {  
      "InstanceId": "i-09eb09efa73ec1dee",  
      "State": "associated",  
      "AssociationId": "iip-assoc-0db249b1f25fa24b8",  
      "IamInstanceProfile": {  
        "Id": "AIPAJVQN4F5WVLGCJDRGM",  
        "Arn": "arn:aws:iam::123456789012:instance-profile/admin-role"  
      }  
    },  
    {  
      "InstanceId": "i-0402909a2f4dff14",  
      "State": "associating",  
      "AssociationId": "iip-assoc-0d1ec06278d29f44a",  
      "IamInstanceProfile": {  
        "Id": "AGJAJVQN4F5WVLGCJABCM",  
        "Arn": "arn:aws:iam::123456789012:instance-profile/admin-role"  
      }  
    }  
  ]  
}
```

```

    "Arn": "arn:aws:iam::123456789012:instance-profile/user1-role"
  }
}
]
}

```

- For API details, see [DescribeIAMInstanceProfileAssociations](#) in *AWS CLI Command Reference*.

describe-id-format

The following code example shows how to use `describe-id-format`.

AWS CLI

Example 1: To describe the ID format of a resource

The following `describe-id-format` example describes the ID format for security groups.

```
aws ec2 describe-id-format \
  --resource security-group
```

In the following example output, the `Deadline` value indicates that the deadline for this resource type to permanently switch from the short ID format to the long ID format expired at 00:00 UTC on August 15, 2018.

```
{
  "Statuses": [
    {
      "Deadline": "2018-08-15T00:00:00.000Z",
      "Resource": "security-group",
      "UseLongIds": true
    }
  ]
}
```

Example 2: To describe the ID format for all resources

The following `describe-id-format` example describes the ID format for all resource types. All resource types that supported the short ID format were switched to use the long ID format.

```
aws ec2 describe-id-format
```

- For API details, see [DescribeIdFormat](#) in *AWS CLI Command Reference*.

describe-identity-id-format

The following code example shows how to use `describe-identity-id-format`.

AWS CLI

To describe the ID format for an IAM role

The following `describe-identity-id-format` example describes the ID format received by instances created by the IAM role `EC2Role` in your AWS account.

```
aws ec2 describe-identity-id-format \
  --principal-arn arn:aws:iam::123456789012:role/my-iam-role \
  --resource instance
```

The following output indicates that instances created by this role receive IDs in long ID format.

```
{
  "Statuses": [
    {
      "Deadline": "2016-12-15T00:00:00Z",
      "Resource": "instance",
      "UseLongIds": true
    }
  ]
}
```

To describe the ID format for an IAM user

The following `describe-identity-id-format` example describes the ID format received by snapshots created by the IAM user `AdminUser` in your AWS account.

```
aws ec2 describe-identity-id-format \
  --principal-arn arn:aws:iam::123456789012:user/AdminUser \
  --resource snapshot
```

The output indicates that snapshots created by this user receive IDs in long ID format.

```
{
```

```
"Statuses": [  
  {  
    "Deadline": "2016-12-15T00:00:00Z",  
    "Resource": "snapshot",  
    "UseLongIds": true  
  }  
]  
}
```

- For API details, see [DescribeIdentityIdFormat](#) in *AWS CLI Command Reference*.

describe-image-attribute

The following code example shows how to use `describe-image-attribute`.

AWS CLI

To describe the launch permissions for an AMI

This example describes the launch permissions for the specified AMI.

Command:

```
aws ec2 describe-image-attribute --image-id ami-5731123e --attribute  
launchPermission
```

Output:

```
{  
  "LaunchPermissions": [  
    {  
      "UserId": "123456789012"  
    }  
  ],  
  "ImageId": "ami-5731123e",  
}
```

To describe the product codes for an AMI

This example describes the product codes for the specified AMI. Note that this AMI has no product codes.

Command:

```
aws ec2 describe-image-attribute --image-id ami-5731123e --attribute productCodes
```

Output:

```
{
  "ProductCodes": [],
  "ImageId": "ami-5731123e",
}
```

- For API details, see [DescribeImageAttribute](#) in *AWS CLI Command Reference*.

describe-images

The following code example shows how to use `describe-images`.

AWS CLI**Example 1: To describe an AMI**

The following `describe-images` example describes the specified AMI in the specified Region.

```
aws ec2 describe-images \
  --region us-east-1 \
  --image-ids ami-1234567890EXAMPLE
```

Output:

```
{
  "Images": [
    {
      "VirtualizationType": "hvm",
      "Description": "Provided by Red Hat, Inc.",
      "PlatformDetails": "Red Hat Enterprise Linux",
      "EnaSupport": true,
      "Hypervisor": "xen",
      "State": "available",
      "SriovNetSupport": "simple",
      "ImageId": "ami-1234567890EXAMPLE",
      "UsageOperation": "RunInstances:0010",
      "BlockDeviceMappings": [
        {
          "DeviceName": "/dev/sda1",
```

```

        "Ebs": {
            "SnapshotId": "snap-111222333444aaabb",
            "DeleteOnTermination": true,
            "VolumeType": "gp2",
            "VolumeSize": 10,
            "Encrypted": false
        }
    ],
    "Architecture": "x86_64",
    "ImageLocation": "123456789012/RHEL-8.0.0_HVM-20190618-x86_64-1-Hourly2-
GP2",
    "RootDeviceType": "ebs",
    "OwnerId": "123456789012",
    "RootDeviceName": "/dev/sda1",
    "CreationDate": "2019-05-10T13:17:12.000Z",
    "Public": true,
    "ImageType": "machine",
    "Name": "RHEL-8.0.0_HVM-20190618-x86_64-1-Hourly2-GP2"
}
]
}

```

For more information, see [Amazon Machine Images \(AMI\)](#) in the *Amazon EC2 User Guide*.

Example 2: To describe AMIs based on filters

The following `describe-images` example describes Windows AMIs provided by Amazon that are backed by Amazon EBS.

```

aws ec2 describe-images \
  --owners amazon \
  --filters "Name=platform,Values=windows" "Name=root-device-type,Values=ebs"

```

For an example of the output for `describe-images`, see Example 1.

For additional examples using filters, see [Listing and filtering your resources](#) in the *Amazon EC2 User Guide*.

Example 3: To describe AMIs based on tags

The following `describe-images` example describes all AMIs that have the tag `Type=Custom`. The example uses the `--query` parameter to display only the AMI IDs.

```
aws ec2 describe-images \  
  --filters "Name=tag:Type,Values=Custom" \  
  --query 'Images[*].[ImageId]' \  
  --output text
```

Output:

```
ami-1234567890EXAMPLE  
ami-0abcdef1234567890
```

For additional examples using tag filters, see [Working with tags](#) in the *Amazon EC2 User Guide*.

- For API details, see [DescribeImages](#) in *AWS CLI Command Reference*.

describe-import-image-tasks

The following code example shows how to use `describe-import-image-tasks`.

AWS CLI

To monitor an import image task

The following `describe-import-image-tasks` example checks the status of the specified import image task.

```
aws ec2 describe-import-image-tasks \  
  --import-task-ids import-ami-1234567890abcdef0
```

Output for an import image task that is in progress.

```
{  
  "ImportImageTasks": [  
    {  
      "ImportTaskId": "import-ami-1234567890abcdef0",  
      "Progress": "28",  
      "SnapshotDetails": [  
        {  
          "DiskImageSize": 705638400.0,  
          "Format": "ova",  
          "Status": "completed",
```



```

        "UserBucket": {
            "S3Bucket": "my-import-bucket",
            "S3Key": "vms/my-server-vm.ova"
        }
    ],
    "Status": "active",
    "StatusMessage": "converting"
}
]
}

```

Output for an import image task that is completed. The ID of the resulting AMI is provided by `ImageId`.

```

{
  "ImportImageTasks": [
    {
      "ImportTaskId": "import-ami-1234567890abcdef0",
      "ImageId": "ami-1234567890abcdef0",
      "SnapshotDetails": [
        {
          "DiskImageSize": 705638400.0,
          "Format": "ova",
          "SnapshotId": "snap-1234567890abcdef0",
          "Status": "completed",
          "UserBucket": {
            "S3Bucket": "my-import-bucket",
            "S3Key": "vms/my-server-vm.ova"
          }
        }
      ],
      "Status": "completed"
    }
  ]
}

```

- For API details, see [DescribeImportImageTasks](#) in *AWS CLI Command Reference*.

describe-import-snapshot-tasks

The following code example shows how to use `describe-import-snapshot-tasks`.

AWS CLI

To monitor an import snapshot task

The following `describe-import-snapshot-tasks` example checks the status of the specified import snapshot task.

```
aws ec2 describe-import-snapshot-tasks \  
  --import-task-ids import-snap-1234567890abcdef0
```

Output for an import snapshot task that is in progress:

```
{  
  "ImportSnapshotTasks": [  
    {  
      "Description": "My server VMDK",  
      "ImportTaskId": "import-snap-1234567890abcdef0",  
      "SnapshotTaskDetail": {  
        "Description": "My server VMDK",  
        "DiskImageSize": "705638400.0",  
        "Format": "VMDK",  
        "Progress": "42",  
        "Status": "active",  
        "StatusMessage": "downloading/converting",  
        "UserBucket": {  
          "S3Bucket": "my-import-bucket",  
          "S3Key": "vms/my-server-vm.vmdk"  
        }  
      }  
    }  
  ]  
}
```

Output for an import snapshot task that is completed. The ID of the resulting snapshot is provided by `SnapshotId`.

```
{  
  "ImportSnapshotTasks": [  
    {  
      "Description": "My server VMDK",  
      "ImportTaskId": "import-snap-1234567890abcdef0",  
      "SnapshotTaskDetail": {
```

```

    "Description": "My server VMDK",
    "DiskImageSize": "705638400.0",
    "Format": "VMDK",
    "SnapshotId": "snap-1234567890abcdef0"
    "Status": "completed",
    "UserBucket": {
      "S3Bucket": "my-import-bucket",
      "S3Key": "vms/my-server-vm.vmdk"
    }
  }
}
]
}

```

- For API details, see [DescribeImportSnapshotTasks](#) in *AWS CLI Command Reference*.

describe-instance-attribute

The following code example shows how to use `describe-instance-attribute`.

AWS CLI

To describe the instance type

This example describes the instance type of the specified instance.

Command:

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --attribute
instanceType
```

Output:

```
{
  "InstanceId": "i-1234567890abcdef0"
  "InstanceType": {
    "Value": "t1.micro"
  }
}
```

To describe the `disableApiTermination` attribute

This example describes the `disableApiTermination` attribute of the specified instance.

Command:

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --attribute
disableApiTermination
```

Output:

```
{
  "InstanceId": "i-1234567890abcdef0"
  "DisableApiTermination": {
    "Value": "false"
  }
}
```

To describe the block device mapping for an instance

This example describes the `blockDeviceMapping` attribute of the specified instance.

Command:

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --attribute
blockDeviceMapping
```

Output:

```
{
  "InstanceId": "i-1234567890abcdef0"
  "BlockDeviceMappings": [
    {
      "DeviceName": "/dev/sda1",
      "Ebs": {
        "Status": "attached",
        "DeleteOnTermination": true,
        "VolumeId": "vol-049df61146c4d7901",
        "AttachTime": "2013-05-17T22:42:34.000Z"
      }
    },
    {
      "DeviceName": "/dev/sdf",
      "Ebs": {
```

```

        "Status": "attached",
        "DeleteOnTermination": false,
        "VolumeId": "vol-049df61146c4d7901",
        "AttachTime": "2013-09-10T23:07:00.000Z"
    }
  ],
}

```

- For API details, see [DescribeInstanceAttribute](#) in *AWS CLI Command Reference*.

describe-instance-connect-endpoints

The following code example shows how to use `describe-instance-connect-endpoints`.

AWS CLI

To describe an EC2 Instance Connect Endpoint

The following `describe-instance-connect-endpoints` example describes the specified EC2 Instance Connect Endpoint.

```

aws ec2 describe-instance-connect-endpoints \
  --region us-east-1 \
  --instance-connect-endpoint-ids eice-0123456789example

```

Output:

```

{
  "InstanceConnectEndpoints": [
    {
      "OwnerId": "111111111111",
      "InstanceConnectEndpointId": "eice-0123456789example",
      "InstanceConnectEndpointArn": "arn:aws:ec2:us-east-1:111111111111:instance-connect-endpoint/eice-0123456789example",
      "State": "create-complete",
      "StateMessage": "",
      "DnsName": "eice-0123456789example.b67b86ba.ec2-instance-connect-endpoint.us-east-1.amazonaws.com",
      "NetworkInterfaceIds": [
        "eni-0123456789example"
      ],
    }
  ],
}

```

```
        "VpcId": "vpc-0123abcd",
        "AvailabilityZone": "us-east-1d",
        "CreatedAt": "2023-02-07T12:05:37+00:00",
        "SubnetId": "subnet-0123abcd",
        "Tags": []
    }
]
```

For more information, see [Create an EC2 Instance Connect Endpoint](#) in the *Amazon EC2 User Guide*.

- For API details, see [DescribeInstanceConnectEndpoints](#) in *AWS CLI Command Reference*.

describe-instance-credit-specifications

The following code example shows how to use `describe-instance-credit-specifications`.

AWS CLI

To describe the credit option for CPU usage of one or more instances

The following `describe-instance-credit-specifications` example describes the CPU credit option for the specified instance.

```
aws ec2 describe-instance-credit-specifications \
  --instance-ids i-1234567890abcdef0
```

Output:

```
{
  "InstanceCreditSpecifications": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "CpuCredits": "unlimited"
    }
  ]
}
```

For more information, see [Work with burstable performance instances](#) in the *Amazon EC2 User Guide*.

- For API details, see [DescribeInstanceCreditSpecifications](#) in *AWS CLI Command Reference*.

describe-instance-event-notification-attributes

The following code example shows how to use `describe-instance-event-notification-attributes`.

AWS CLI

To describe the tags for scheduled event notifications

The following `describe-instance-event-notification-attributes` example describes the tags to appear in scheduled event notifications.

```
aws ec2 describe-instance-event-notification-attributes
```

Output:

```
{
  "InstanceTagAttribute": {
    "InstanceTagKeys": [],
    "IncludeAllTagsOfInstance": true
  }
}
```

For more information, see [Scheduled events for your instances](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [DescribeInstanceEventNotificationAttributes](#) in *AWS CLI Command Reference*.

describe-instance-event-windows

The following code example shows how to use `describe-instance-event-windows`.

AWS CLI

Example 1: To describe all event windows

The following `describe-instance-event-windows` example describes all event windows in the specified Region.

```
aws ec2 describe-instance-event-windows \
  --region us-east-1
```

Output:

```
{
  "InstanceEventWindows": [
    {
      "InstanceEventWindowId": "iew-0abcdef1234567890",
      "Name": "myEventWindowName",
      "CronExpression": "* 21-23 * * 2,3",
      "AssociationTarget": {
        "InstanceIds": [
          "i-1234567890abcdef0",
          "i-0598c7d356eba48d7"
        ],
        "Tags": [],
        "DedicatedHostIds": []
      },
      "State": "active",
      "Tags": []
    }
    ...
  ],
  "NextToken": "9d624e0c-388b-4862-a31e-a85c64fc1d4a"
}
```

Example 2: To describe a specific event window

The following describe-instance-event-windows example describes a specific event by using the instance-event-window parameter to describe a specific event window.

```
aws ec2 describe-instance-event-windows \
  --region us-east-1 \
  --instance-event-window-ids iew-0abcdef1234567890
```

Output:

```
{
  "InstanceEventWindows": [
    {
      "InstanceEventWindowId": "iew-0abcdef1234567890",
      "Name": "myEventWindowName",
```



```

    "CronExpression": "* 21-23 * * 2,3",
    "AssociationTarget": {
      "InstanceIds": [
        "i-1234567890abcdef0",
        "i-0598c7d356eba48d7"
      ],
      "Tags": [],
      "DedicatedHostIds": []
    },
    "State": "active",
    "Tags": []
  }
}

```

Example 3: To describe event windows that match one or more filters

The following `describe-instance-event-windows` example describes event windows that match one or more filters using the `filter` parameter. The `instance-id` filter is used to describe all of the event windows that are associated with the specified instance. When a filter is used, it performs a direct match. However, the `instance-id` filter is different. If there is no direct match to the instance ID, then it falls back to indirect associations with the event window, such as the tags of the instance or Dedicated Host ID (if the instance is a Dedicated Host).

```

aws ec2 describe-instance-event-windows \
  --region us-east-1 \
  --filters Name=instance-id,Values=i-1234567890abcdef0 \
  --max-results 100 \
  --next-token <next-token-value>

```

Output:

```

{
  "InstanceEventWindows": [
    {
      "InstanceEventWindowId": "iew-0dbc0adb66f235982",
      "TimeRanges": [
        {
          "StartWeekDay": "sunday",
          "StartHour": 2,
          "EndWeekDay": "sunday",
          "EndHour": 8
        }
      ]
    }
  ]
}

```

```

    ],
    "Name": "myEventWindowName",
    "AssociationTarget": {
      "InstanceIds": [],
      "Tags": [],
      "DedicatedHostIds": [
        "h-0140d9a7ecbd102dd"
      ]
    },
    "State": "active",
    "Tags": []
  }
]
}

```

In the example output, the instance is on a Dedicated Host, which is associated with the event window.

For event window constraints, see [Considerations](#) in the *Amazon EC2 User Guide*.

- For API details, see [DescribeInstanceEventWindows](#) in *AWS CLI Command Reference*.

describe-instance-status

The following code example shows how to use `describe-instance-status`.

AWS CLI

To describe the status of an instance

The following `describe-instance-status` example describes the current status of the specified instance.

```
aws ec2 describe-instance-status \
  --instance-ids i-1234567890abcdef0
```

Output:

```
{
  "InstanceStatuses": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "InstanceState": {
```

```
        "Code": 16,
        "Name": "running"
    },
    "AvailabilityZone": "us-east-1d",
    "SystemStatus": {
        "Status": "ok",
        "Details": [
            {
                "Status": "passed",
                "Name": "reachability"
            }
        ]
    },
    "InstanceState": {
        "Status": "ok",
        "Details": [
            {
                "Status": "passed",
                "Name": "reachability"
            }
        ]
    }
}
]
```

For more information, see [Monitor the status of your instances](#) in the *Amazon EC2 User Guide*.

- For API details, see [DescribeInstanceStatus](#) in *AWS CLI Command Reference*.

describe-instance-topology

The following code example shows how to use `describe-instance-topology`.

AWS CLI

To describe the instance topology of all your instances

The following `describe-instance-topology` example describes the topology of all your instances that match the supported instance types for this command.

```
aws ec2 describe-instance-topology \  
  --region us-west-2
```

Output:

```
{
  "Instances": [
    {
      "InstanceId": "i-1111111111example",
      "InstanceType": "p4d.24xlarge",
      "GroupName": "my-ml-cpg",
      "NetworkNodes": [
        "nn-1111111111example",
        "nn-2222222222example",
        "nn-3333333333example"
      ],
      "ZoneId": "usw2-az2",
      "AvailabilityZone": "us-west-2a"
    },
    {
      "InstanceId": "i-2222222222example",
      "InstanceType": "p4d.24xlarge",
      "NetworkNodes": [
        "nn-1111111111example",
        "nn-2222222222example",
        "nn-3333333333example"
      ],
      "ZoneId": "usw2-az2",
      "AvailabilityZone": "us-west-2a"
    },
    {
      "InstanceId": "i-3333333333example",
      "InstanceType": "trn1.32xlarge",
      "NetworkNodes": [
        "nn-1212121212example",
        "nn-1211122211example",
        "nn-1311133311example"
      ],
      "ZoneId": "usw2-az4",
      "AvailabilityZone": "us-west-2d"
    },
    {
      "InstanceId": "i-4444444444example",
      "InstanceType": "trn1.2xlarge",
      "NetworkNodes": [
        "nn-1111111111example",
        "nn-5434334334example",

```

```
        "nn-1235301234example"  
      ],  
      "ZoneId": "usw2-az2",  
      "AvailabilityZone": "us-west-2a"  
    }  
  ],  
  "NextToken": "SomeEncryptedToken"  
}
```

For more information, including more examples, see [Amazon EC2 instance topology](#) in the *Amazon EC2 User Guide*.

- For API details, see [DescribeInstanceTopology](#) in *AWS CLI Command Reference*.

describe-instance-type-offerings

The following code example shows how to use `describe-instance-type-offerings`.

AWS CLI

Example 1: To list the instance types offered in a Region

The following `describe-instance-type-offerings` example lists the instance types offered in the Region configured as the default Region for the AWS CLI.

```
aws ec2 describe-instance-type-offerings
```

To list the instance types offered in a different Region, specify the Region using the `--region` parameter.

```
aws ec2 describe-instance-type-offerings \  
  --region us-east-2
```

Output:

```
{  
  "InstanceTypeOfferings": [  
    {  
      "InstanceType": "m5.2xlarge",  
      "LocationType": "region",  
      "Location": "us-east-2"    }  
  ]  
}
```

```
    },  
    {  
      "InstanceType": "t3.micro",  
      "LocationType": "region",  
      "Location": "us-east-2"  
    },  
    ...  
  ]  
}
```

Example 2: To list the instance types offered in an Availability Zone

The following `describe-instance-type-offerings` example lists the instance types offered in the specified Availability Zone. The Availability Zone must be in the specified Region.

```
aws ec2 describe-instance-type-offerings \  
  --location-type availability-zone \  
  --filters Name=location,Values=us-east-2a \  
  --region us-east-2
```

Example 3: To check whether an instance type is supported

The following `describe-instance-type-offerings` command indicates whether the `c5.xlarge` instance type is supported in the specified Region.

```
aws ec2 describe-instance-type-offerings \  
  --filters Name=instance-type,Values=c5.xlarge \  
  --region us-east-2
```

The following `describe-instance-type-offerings` example lists all C5 instance types that are supported in the specified Region.

```
aws ec2 describe-instance-type-offerings \  
  --filters Name=instance-type,Values=c5* \  
  --query "InstanceTypeOfferings[].InstanceType" \  
  --region us-east-2
```

Output:

```
[
```

```
"c5d.12xlarge",  
"c5d.9xlarge",  
"c5n.xlarge",  
"c5.xlarge",  
"c5d.metal",  
"c5n.metal",  
"c5.large",  
"c5d.2xlarge",  
"c5n.4xlarge",  
"c5.2xlarge",  
"c5n.large",  
"c5n.9xlarge",  
"c5d.large",  
"c5.18xlarge",  
"c5d.18xlarge",  
"c5.12xlarge",  
"c5n.18xlarge",  
"c5.metal",  
"c5d.4xlarge",  
"c5.24xlarge",  
"c5d.xlarge",  
"c5n.2xlarge",  
"c5d.24xlarge",  
"c5.9xlarge",  
"c5.4xlarge"
```

```
]
```

- For API details, see [DescribeInstanceTypeOfferings](#) in *AWS CLI Command Reference*.

describe-instance-types

The following code example shows how to use `describe-instance-types`.

AWS CLI

Example 1: To describe an instance type

The following `describe-instance-types` example displays details for the specified instance type.

```
aws ec2 describe-instance-types \  
  --instance-types t2.micro
```

Output:

```
{
  "InstanceTypes": [
    {
      "InstanceType": "t2.micro",
      "CurrentGeneration": true,
      "FreeTierEligible": true,
      "SupportedUsageClasses": [
        "on-demand",
        "spot"
      ],
      "SupportedRootDeviceTypes": [
        "ebs"
      ],
      "BareMetal": false,
      "Hypervisor": "xen",
      "ProcessorInfo": {
        "SupportedArchitectures": [
          "i386",
          "x86_64"
        ],
        "SustainedClockSpeedInGhz": 2.5
      },
      "VCpuInfo": {
        "DefaultVCpus": 1,
        "DefaultCores": 1,
        "DefaultThreadsPerCore": 1,
        "ValidCores": [
          1
        ],
        "ValidThreadsPerCore": [
          1
        ]
      },
      "MemoryInfo": {
        "SizeInMiB": 1024
      },
      "InstanceStorageSupported": false,
      "EbsInfo": {
        "EbsOptimizedSupport": "unsupported",
        "EncryptionSupport": "supported"
      },
      "NetworkInfo": {
```



```

        "NetworkPerformance": "Low to Moderate",
        "MaximumNetworkInterfaces": 2,
        "Ipv4AddressesPerInterface": 2,
        "Ipv6AddressesPerInterface": 2,
        "Ipv6Supported": true,
        "EnaSupport": "unsupported"
    },
    "PlacementGroupInfo": {
        "SupportedStrategies": [
            "partition",
            "spread"
        ]
    },
    "HibernationSupported": false,
    "BurstablePerformanceSupported": true,
    "DedicatedHostsSupported": false,
    "AutoRecoverySupported": true
}
]
}

```

For more information, see [Instance Types](#) in *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

Example 2: To filter the available instance types

You can specify a filter to scope the results to instance types that have a specific characteristic. The following `describe-instance-types` example lists the instance types that support hibernation.

```

aws ec2 describe-instance-types \
  --filters Name=hibernation-supported,Values=true --query
  'InstanceTypes[*].InstanceType'

```

Output:

```

[
  "m5.8xlarge",
  "r3.large",
  "c3.8xlarge",
  "r5.large",
  "m4.4xlarge",
  "c4.large",

```

```
"m5.xlarge",  
"m4.xlarge",  
"c3.large",  
"c4.8xlarge",  
"c4.4xlarge",  
"c5.xlarge",  
"c5.12xlarge",  
"r5.4xlarge",  
"c5.4xlarge"  
]
```

For more information, see [Instance Types](#) in *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [DescribeInstanceTypes](#) in *AWS CLI Command Reference*.

describe-instances

The following code example shows how to use `describe-instances`.

AWS CLI

Example 1: To describe an instance

The following `describe-instances` example describes the specified instance.

```
aws ec2 describe-instances \  
--instance-ids i-1234567890abcdef0
```

Output:

```
{  
  "Reservations": [  
    {  
      "Groups": [],  
      "Instances": [  
        {  
          "AmiLaunchIndex": 0,  
          "ImageId": "ami-0abcdef1234567890",  
          "InstanceId": "i-1234567890abcdef0",  
          "InstanceType": "t3.nano",  
          "KeyName": "my-key-pair",  
          "LaunchTime": "2022-11-15T10:48:59+00:00",
```

```
"Monitoring": {
  "State": "disabled"
},
"Placement": {
  "AvailabilityZone": "us-east-2a",
  "GroupName": "",
  "Tenancy": "default"
},
"PrivateDnsName": "ip-10-0-0-157.us-east-2.compute.internal",
"PrivateIpAddress": "10-0-0-157",
"ProductCodes": [],
"PublicDnsName": "ec2-34-253-223-13.us-
east-2.compute.amazonaws.com",
"PublicIpAddress": "34.253.223.13",
"State": {
  "Code": 16,
  "Name": "running"
},
"StateTransitionReason": "",
"SubnetId": "subnet-04a636d18e83cfacb",
"VpcId": "vpc-1234567890abcdef0",
"Architecture": "x86_64",
"BlockDeviceMappings": [
  {
    "DeviceName": "/dev/xvda",
    "Ebs": {
      "AttachTime": "2022-11-15T10:49:00+00:00",
      "DeleteOnTermination": true,
      "Status": "attached",
      "VolumeId": "vol-02e6ccdca7de29cf2"
    }
  }
],
"ClientToken": "1234abcd-1234-abcd-1234-d46a8903e9bc",
"EbsOptimized": true,
"EnaSupport": true,
"Hypervisor": "xen",
"IamInstanceProfile": {
  "Arn": "arn:aws:iam::111111111111:instance-profile/
AmazonSSMRoleForInstancesQuickSetup",
  "Id": "11111111111111111111111111111111"
},
"NetworkInterfaces": [
  {
```

```
    "Association": {
      "IpOwnerId": "amazon",
      "PublicDnsName": "ec2-34-253-223-13.us-
east-2.compute.amazonaws.com",
      "PublicIp": "34.253.223.13"
    },
    "Attachment": {
      "AttachTime": "2022-11-15T10:48:59+00:00",
      "AttachmentId": "eni-attach-1234567890abcdefg",
      "DeleteOnTermination": true,
      "DeviceIndex": 0,
      "Status": "attached",
      "NetworkCardIndex": 0
    },
    "Description": "",
    "Groups": [
      {
        "GroupName": "launch-wizard-146",
        "GroupId": "sg-1234567890abcdefg"
      }
    ],
    "Ipv6Addresses": [],
    "MacAddress": "00:11:22:33:44:55",
    "NetworkInterfaceId": "eni-1234567890abcdefg",
    "OwnerId": "104024344472",
    "PrivateDnsName": "ip-10-0-0-157.us-
east-2.compute.internal",
    "PrivateIpAddress": "10-0-0-157",
    "PrivateIpAddresses": [
      {
        "Association": {
          "IpOwnerId": "amazon",
          "PublicDnsName": "ec2-34-253-223-13.us-
east-2.compute.amazonaws.com",
          "PublicIp": "34.253.223.13"
        },
        "Primary": true,
        "PrivateDnsName": "ip-10-0-0-157.us-
east-2.compute.internal",
        "PrivateIpAddress": "10-0-0-157"
      }
    ],
    "SourceDestCheck": true,
    "Status": "in-use",
```

```
        "SubnetId": "subnet-1234567890abcdefg",
        "VpcId": "vpc-1234567890abcdefg",
        "InterfaceType": "interface"
    }
],
"RootDeviceName": "/dev/xvda",
"RootDeviceType": "ebs",
"SecurityGroups": [
    {
        "GroupName": "launch-wizard-146",
        "GroupId": "sg-1234567890abcdefg"
    }
],
"SourceDestCheck": true,
"Tags": [
    {
        "Key": "Name",
        "Value": "my-instance"
    }
],
"VirtualizationType": "hvm",
"CpuOptions": {
    "CoreCount": 1,
    "ThreadsPerCore": 2
},
"CapacityReservationSpecification": {
    "CapacityReservationPreference": "open"
},
"HibernationOptions": {
    "Configured": false
},
"MetadataOptions": {
    "State": "applied",
    "HttpTokens": "optional",
    "HttpPutResponseHopLimit": 1,
    "HttpEndpoint": "enabled",
    "HttpProtocolIpv6": "disabled",
    "InstanceMetadataTags": "enabled"
},
"EnclaveOptions": {
    "Enabled": false
},
"PlatformDetails": "Linux/UNIX",
"UsageOperation": "RunInstances",
```

```
        "UsageOperationUpdateTime": "2022-11-15T10:48:59+00:00",
        "PrivateDnsNameOptions": {
            "HostnameType": "ip-name",
            "EnableResourceNameDnsARecord": true,
            "EnableResourceNameDnsAAAARecord": false
        },
        "MaintenanceOptions": {
            "AutoRecovery": "default"
        }
    },
    "OwnerId": "111111111111",
    "ReservationId": "r-1234567890abcdefg"
}
]
```

Example 2: To filter for instances with the specified type

The following `describe-instances` example uses filters to scope the results to instances of the specified type.

```
aws ec2 describe-instances \
  --filters Name=instance-type,Values=m5.large
```

For example output, see Example 1.

For more information, see [List and filter using the CLI](#) in the *Amazon EC2 User Guide*.

Example 3: To filter for instances with the specified type and Availability Zone

The following `describe-instances` example uses multiple filters to scope the results to instances with the specified type that are also in the specified Availability Zone.

```
aws ec2 describe-instances \
  --filters Name=instance-type,Values=t2.micro,t3.micro Name=availability-
  zone,Values=us-east-2c
```

For example output, see Example 1.

Example 4: To filter for instances with the specified type and Availability Zone using a JSON file

The following `describe-instances` example uses a JSON input file to perform the same filtering as the previous example. When filters get more complicated, they can be easier to specify in a JSON file.

```
aws ec2 describe-instances \  
  --filters file://filters.json
```

Contents of `filters.json`:

```
[  
  {  
    "Name": "instance-type",  
    "Values": ["t2.micro", "t3.micro"]  
  },  
  {  
    "Name": "availability-zone",  
    "Values": ["us-east-2c"]  
  }  
]
```

For example output, see Example 1.

Example 5: To filter for instances with the specified Owner tag

The following `describe-instances` example uses tag filters to scope the results to instances that have a tag with the specified tag key (Owner), regardless of the tag value.

```
aws ec2 describe-instances \  
  --filters "Name=tag-key,Values=Owner"
```

For example output, see Example 1.

Example 6: To filter for instances with the specified my-team tag value

The following `describe-instances` example uses tag filters to scope the results to instances that have a tag with the specified tag value (my-team), regardless of the tag key.

```
aws ec2 describe-instances \  
  --filters "Name=tag-value,Values=my-team"
```

For example output, see Example 1.

Example 7: To filter for instances with the specified Owner tag and my-team value

The following `describe-instances` example uses tag filters to scope the results to instances that have the specified tag (`Owner=my-team`).

```
aws ec2 describe-instances \  
  --filters "Name=tag:Owner,Values=my-team"
```

For example output, see Example 1.

Example 8: To display only instance and subnet IDs for all instances

The following `describe-instances` examples use the `--query` parameter to display only the instance and subnet IDs for all instances, in JSON format.

Linux and macOS:

```
aws ec2 describe-instances \  
  --query 'Reservations[*].Instances[*].{Instance:InstanceId,Subnet:SubnetId}' \  
  --output json
```

Windows:

```
aws ec2 describe-instances ^  
  --query "Reservations[*].Instances[*].{Instance:InstanceId,Subnet:SubnetId}" ^  
  --output json
```

Output:

```
[  
  {  
    "Instance": "i-057750d42936e468a",  
    "Subnet": "subnet-069beee9b12030077"  
  },  
  {  
    "Instance": "i-001efd250faaa6ffa",  
    "Subnet": "subnet-0b715c6b7db68927a"  
  },  
  {  
    "Instance": "i-027552a73f021f3bd",  
    "Subnet": "subnet-0250c25a1f4e15235"  
  }  
]
```



```
...
]
```

Example 9: To filter instances of the specified type and only display their instance IDs

The following `describe-instances` example uses filters to scope the results to instances of the specified type and the `--query` parameter to display only the instance IDs.

```
aws ec2 describe-instances \
  --filters "Name=instance-type,Values=t2.micro" \
  --query "Reservations[*].Instances[*].[InstanceId]" \
  --output text
```

Output:

```
i-031c0dc19de2fb70c
i-00d8bfff789a736b75
i-0b715c6b7db68927a
i-0626d4edd54f1286d
i-00b8ae04f9f99908e
i-0fc71c25d2374130c
```

Example 10: To filter instances of the specified type and only display their instance IDs, Availability Zone, and the specified tag value

The following `describe-instances` examples display the instance ID, Availability Zone, and the value of the Name tag for instances that have a tag with the name `tag-key`, in table format.

Linux and macOS:

```
aws ec2 describe-instances \
  --filters Name=tag-key,Values=Name \
  --query 'Reservations[*].Instances[*].
{Instance:InstanceId,AZ:Placement.AvailabilityZone,Name:Tags[?Key==`Name`]|
[0].Value}' \
  --output table
```

Windows:

```
aws ec2 describe-instances ^
```

```

--filters Name=tag-key,Values=Name ^
--query "Reservations[*].Instances[*].
{Instance:InstanceId,AZ:Placement.AvailabilityZone,Name:Tags[?Key=='Name']|
[0].Value}" ^
--output table

```

Output:

```

-----
|                               DescribeInstances                               |
+-----+-----+-----+
|      AZ      |      Instance      |      Name      |
+-----+-----+-----+
| us-east-2b  | i-057750d42936e468a | my-prod-server |
| us-east-2a  | i-001efd250faaa6ffa | test-server-1   |
| us-east-2a  | i-027552a73f021f3bd | test-server-2   |
+-----+-----+-----+

```

Example 11: To describe instances in a partition placement group

The following `describe-instances` example describes the specified instance. The output includes the placement information for the instance, which contains the placement group name and the partition number for the instance.

```

aws ec2 describe-instances \
  --instance-ids i-0123a456700123456 \
  --query "Reservations[*].Instances[*].Placement"

```

Output:

```

[
  [
    {
      "AvailabilityZone": "us-east-1c",
      "GroupName": "HDFS-Group-A",
      "PartitionNumber": 3,
      "Tenancy": "default"
    }
  ]
]

```

For more information, see [Describing instances in a placement group](#) in the *Amazon EC2 User Guide*.

Example 12: To filter to instances with the specified placement group and partition number

The following `describe-instances` example filters the results to only those instances with the specified placement group and partition number.

```
aws ec2 describe-instances \  
  --filters "Name=placement-group-name,Values=HDFS-Group-A" "Name=placement-  
partition-number,Values=7"
```

The following shows only the relevant information from the output.

```
"Instances": [  
  {  
    "InstanceId": "i-0123a456700123456",  
    "InstanceType": "r4.large",  
    "Placement": {  
      "AvailabilityZone": "us-east-1c",  
      "GroupName": "HDFS-Group-A",  
      "PartitionNumber": 7,  
      "Tenancy": "default"  
    }  
  },  
  {  
    "InstanceId": "i-9876a543210987654",  
    "InstanceType": "r4.large",  
    "Placement": {  
      "AvailabilityZone": "us-east-1c",  
      "GroupName": "HDFS-Group-A",  
      "PartitionNumber": 7,  
      "Tenancy": "default"  
    }  
  },  
],
```

For more information, see [Describing instances in a placement group](#) in the *Amazon EC2 User Guide*.

Example 13: To filter to instances that are configured to allow access to tags from instance metadata

The following `describe-instances` example filters the results to only those instances that are configured to allow access to instance tags from instance metadata.

```
aws ec2 describe-instances \  
  --filters "Name=metadata-options.instance-metadata-tags,Values=enabled" \  
  --query "Reservations[*].Instances[*].InstanceId" \  
  --output text
```

The following shows the expected output.

```
i-1234567890abcdefg  
i-abcdefg1234567890  
i-1111111111aaaaaaaa  
i-aaaaaaaa1111111111
```

For more information, see [Work with instance tags in instance metadata](#) in the *Amazon EC2 User Guide*.

- For API details, see [DescribeInstances](#) in *AWS CLI Command Reference*.

describe-internet-gateways

The following code example shows how to use `describe-internet-gateways`.

AWS CLI

To describe an internet gateway

The following `describe-internet-gateways` example describes the specified internet gateway.

```
aws ec2 describe-internet-gateways \  
  --internet-gateway-ids igw-0d0fb496b3EXAMPLE
```

Output:

```
{  
  "InternetGateways": [  
    {  
      "Attachments": [  

```

```
        {
            "State": "available",
            "VpcId": "vpc-0a60eb65b4EXAMPLE"
        }
    ],
    "InternetGatewayId": "igw-0d0fb496b3EXAMPLE",
    "OwnerId": "123456789012",
    "Tags": [
        {
            "Key": "Name",
            "Value": "my-igw"
        }
    ]
}
]
```

For more information, see [Internet gateways](#) in the *Amazon VPC User Guide*.

- For API details, see [DescribeInternetGateways](#) in *AWS CLI Command Reference*.

describe-ipam-pools

The following code example shows how to use `describe-ipam-pools`.

AWS CLI

To view the details for an IPAM pool

The following `describe-ipam-pools` example shows the details for pools.

(Linux):

```
aws ec2 describe-ipam-pools \
    --filters Name=owner-id,Values=123456789012 Name=ipam-scope-id,Values=ipam-
scope-02fc38cd4c48e7d38
```

(Windows):

```
aws ec2 describe-ipam-pools ^
    --filters Name=owner-id,Values=123456789012 Name=ipam-scope-id,Values=ipam-
scope-02fc38cd4c48e7d38
```

Output:

```
{
  "IpamPools": [
    {
      "OwnerId": "123456789012",
      "IpamPoolId": "ipam-pool-02ec043a19bbe5d08",
      "IpamPoolArn": "arn:aws:ec2::123456789012:ipam-pool/ipam-
pool-02ec043a19bbe5d08",
      "IpamScopeArn": "arn:aws:ec2::123456789012:ipam-scope/ipam-
scope-02fc38cd4c48e7d38",
      "IpamScopeType": "private",
      "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",
      "IpamRegion": "us-east-1",
      "Locale": "None",
      "PoolDepth": 1,
      "State": "create-complete",
      "AutoImport": true,
      "AddressFamily": "ipv4",
      "AllocationMinNetmaskLength": 16,
      "AllocationMaxNetmaskLength": 26,
      "AllocationDefaultNetmaskLength": 24,
      "AllocationResourceTags": [
        {
          "Key": "Environment",
          "Value": "Preprod"
        }
      ],
      "Tags": [
        {
          "Key": "Name",
          "Value": "Preprod pool"
        }
      ]
    }
  ]
}
```

- For API details, see [DescribeIpamPools](#) in *AWS CLI Command Reference*.

describe-ipam-resource-discoveries

The following code example shows how to use `describe-ipam-resource-discoveries`.

AWS CLI

Example 1: View complete details of resource discoveries

In this example, you're a delegated IPAM admin who wants to create and share a resource discovery with the IPAM admin in another AWS Organization so that the admin can manage and monitor the IP addresses of resources in your organization.

This example may be useful if:

You tried to create a resource discovery, but you got an error that you've reached your limit of 1. You realize that you may have already created a resource discovery and you want to view it in your account. You have resources in a Region that are not being discovered by the IPAM. You want to view the `--operating-regions` defined for the resource and ensure that you've added the right Region as an operating Region so that the resources there can be discovered.

The following `describe-ipam-resource-discoveries` example lists the details of the resource discovery in your AWS account. You can have one resource discovery per AWS Region.

```
aws ec2 describe-ipam-resource-discoveries \
  --region us-east-1
```

Output:

```
{
  "IpamResourceDiscoveries": [
    {
      "OwnerId": "149977607591",
      "IpamResourceDiscoveryId": "ipam-res-disco-0f8bdee9067137c0d",
      "IpamResourceDiscoveryArn": "arn:aws:ec2::149977607591:ipam-resource-
discovery/ipam-res-disco-0f8bdee9067137c0d",
      "IpamResourceDiscoveryRegion": "us-east-1",
      "OperatingRegions": [
        {
          "RegionName": "us-east-1"
        }
      ],
      "IsDefault": false,
      "State": "create-complete",
      "Tags": []
    }
  ]
}
```

```
}
```

For more information, see [Integrate IPAM with accounts outside of your organization](#) in the *Amazon VPC IPAM User Guide*.

Example 2: View only resource discovery IDs

The following `describe-ipam-resource-discoveries` example lists the ID of the resource discovery in your AWS account. You can have one resource discovery per AWS Region.

```
aws ec2 describe-ipam-resource-discoveries \
  --query "IpamResourceDiscoveries[*].IpamResourceDiscoveryId" \
  --output text
```

Output:

```
ipam-res-disco-0481e39b242860333
```

For more information, see [Integrate IPAM with accounts outside of your organization](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [DescribeIpamResourceDiscoveries](#) in *AWS CLI Command Reference*.

describe-ipam-resource-discovery-associations

The following code example shows how to use `describe-ipam-resource-discovery-associations`.

AWS CLI

To view all resource discovery associations with your IPAM

In this example, you're a IPAM delegated admin who has associated resource discoveries with your IPAM to integrate other accounts with your IPAM. You've noticed that your IPAM is not discovering the resources in the operating Regions of the resource discovery as expected. You want to check the status and state of the resource discovery to ensure that the account that created it is still active and the resource discovery is still being shared.

The `--region` must be the home Region of your IPAM.

The following `describe-ipam-resource-discovery-associations` example lists the resource discovery associations in your AWS account.


```
aws ec2 describe-ipam-resource-discovery-associations \  
  --region us-east-1
```

Output:

```
{  
  "IpamResourceDiscoveryAssociations": [  
    {  
      "OwnerId": "320805250157",  
      "IpamResourceDiscoveryAssociationId": "ipam-res-disco-  
assoc-05e6b45eca5bf5cf7",  
      "IpamResourceDiscoveryAssociationArn": "arn:aws:ec2::320805250157:ipam-  
resource-discovery-association/ipam-res-disco-  
assoc-05e6b45eca5bf5cf7",  
      "IpamResourceDiscoveryId": "ipam-res-disco-0f4ef577a9f37a162",  
      "IpamId": "ipam-005f921c17ebd5107",  
      "IpamArn": "arn:aws:ec2::320805250157:ipam/ipam-005f921c17ebd5107",  
      "IpamRegion": "us-east-1",  
      "IsDefault": true,  
      "ResourceDiscoveryStatus": "active",  
      "State": "associate-complete",  
      "Tags": []  
    },  
    {  
      "OwnerId": "149977607591",  
      "IpamResourceDiscoveryAssociationId": "ipam-res-disco-  
assoc-0dfd21ae189ab5f62",  
      "IpamResourceDiscoveryAssociationArn": "arn:aws:ec2::149977607591:ipam-  
resource-discovery-association/ipam-res-disco-  
assoc-0dfd21ae189ab5f62",  
      "IpamResourceDiscoveryId": "ipam-res-disco-0365d2977fc1672fe",  
      "IpamId": "ipam-005f921c17ebd5107",  
      "IpamArn": "arn:aws:ec2::149977607591:ipam/ipam-005f921c17ebd5107",  
      "IpamRegion": "us-east-1",  
      "IsDefault": false,  
      "ResourceDiscoveryStatus": "active",  
      "State": "create-complete",  
      "Tags": []  
    }  
  ]  
}
```

In this example, after running this command, you notice that you have one non-default resource discovery ("IsDefault": false ``) that is ``"ResourceDiscoveryStatus":

"not-found" and "State": "create-complete". The resource discovery owner's account has been closed. If, in another case, you notice that is "ResourceDiscoveryStatus": "not-found" and "State": "associate-complete", this indicates that one of the following has happened:

The resource discovery was deleted by the resource discovery owner. The resource discovery owner unshared the resource discovery.

For more information, see [Integrate IPAM with accounts outside of your organization](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [DescribeIpamResourceDiscoveryAssociations](#) in *AWS CLI Command Reference*.

describe-ipam-scopes

The following code example shows how to use describe-ipam-scopes.

AWS CLI

To view the details for an IPAM scope

The following describe-ipam-scopes example shows the details for scopes.

```
aws ec2 describe-ipam-scopes \
  --filters Name=owner-id,Values=123456789012 Name=ipam-
  id,Values=ipam-08440e7a3acde3908
```

Output:

```
{
  "IpamScopes": [
    {
      "OwnerId": "123456789012",
      "IpamScopeId": "ipam-scope-02fc38cd4c48e7d38",
      "IpamScopeArn": "arn:aws:ec2::123456789012:ipam-scope/ipam-
scope-02fc38cd4c48e7d38",
      "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",
      "IpamRegion": "us-east-1",
      "IpamScopeType": "private",
      "IsDefault": true,
      "PoolCount": 2,
```

```

    "State": "create-complete",
    "Tags": []
  },
  {
    "OwnerId": "123456789012",
    "IpamScopeId": "ipam-scope-0b9eed026396dbc16",
    "IpamScopeArn": "arn:aws:ec2::123456789012:ipam-scope/ipam-
scope-0b9eed026396dbc16",
    "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",
    "IpamRegion": "us-east-1",
    "IpamScopeType": "public",
    "IsDefault": true,
    "PoolCount": 0,
    "State": "create-complete",
    "Tags": []
  },
  {
    "OwnerId": "123456789012",
    "IpamScopeId": "ipam-scope-0f1aff29486355c22",
    "IpamScopeArn": "arn:aws:ec2::123456789012:ipam-scope/ipam-
scope-0f1aff29486355c22",
    "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",
    "IpamRegion": "us-east-1",
    "IpamScopeType": "private",
    "IsDefault": false,
    "Description": "Example description",
    "PoolCount": 0,
    "State": "create-complete",
    "Tags": [
      {
        "Key": "Name",
        "Value": "Example name value"
      }
    ]
  }
]
}

```

- For API details, see [DescribeIpamScopes](#) in *AWS CLI Command Reference*.

describe-ipams

The following code example shows how to use `describe-ipams`.

AWS CLI

To view the details for an IPAM

The following `describe-ipams` example shows the details of an IPAM.

```
aws ec2 describe-ipams \  
  --filters Name=owner-id,Values=123456789012
```

Output:

```
{  
  "Ipams": [  
    {  
      "OwnerId": "123456789012",  
      "IpamId": "ipam-08440e7a3acde3908",  
      "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",  
      "IpamRegion": "us-east-1",  
      "PublicDefaultScopeId": "ipam-scope-0b9eed026396dbc16",  
      "PrivateDefaultScopeId": "ipam-scope-02fc38cd4c48e7d38",  
      "ScopeCount": 3,  
      "OperatingRegions": [  
        {  
          "RegionName": "us-east-1"  
        },  
        {  
          "RegionName": "us-east-2"  
        },  
        {  
          "RegionName": "us-west-1"  
        }  
      ],  
      "State": "create-complete",  
      "Tags": [  
        {  
          "Key": "Name",  
          "Value": "ExampleIPAM"  
        }  
      ]  
    }  
  ]  
}
```

- For API details, see [DescribeIpam](#) in *AWS CLI Command Reference*.

describe-ipv6-pools

The following code example shows how to use `describe-ipv6-pools`.

AWS CLI

To describe your IPv6 address pools

The following `describe-ipv6-pools` example displays details for all of your IPv6 address pools.

```
aws ec2 describe-ipv6-pools
```

Output:

```
{
  "Ipv6Pools": [
    {
      "PoolId": "ipv6pool-ec2-012345abc12345abc",
      "PoolCidrBlocks": [
        {
          "Cidr": "2001:db8:123::/48"
        }
      ],
      "Tags": [
        {
          "Key": "pool-1",
          "Value": "public"
        }
      ]
    }
  ]
}
```

- For API details, see [DescribeIpv6Pools](#) in *AWS CLI Command Reference*.

describe-key-pairs

The following code example shows how to use `describe-key-pairs`.

AWS CLI

To display a key pair

The following `describe-key-pairs` example displays information about the specified key pair.

```
aws ec2 describe-key-pairs \  
  --key-names my-key-pair
```

Output:

```
{  
  "KeyPairs": [  
    {  
      "KeyPairId": "key-0b94643da6EXAMPLE",  
      "KeyFingerprint":  
"1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f",  
      "KeyName": "my-key-pair",  
      "KeyType": "rsa",  
      "Tags": [],  
      "CreateTime": "2022-05-27T21:51:16.000Z"  
    }  
  ]  
}
```

For more information, see [Describe public keys](#) in the *Amazon EC2 User Guide*.

- For API details, see [DescribeKeyPairs](#) in *AWS CLI Command Reference*.

describe-launch-template-versions

The following code example shows how to use `describe-launch-template-versions`.

AWS CLI

To describe launch template versions

This example describes the versions of the specified launch template.

Command:

```
aws ec2 describe-launch-template-versions --launch-template-id lt-068f72b72934aff71
```

Output:

```
{
  "LaunchTemplateVersions": [
    {
      "LaunchTemplateId": "lt-068f72b72934aff71",
      "LaunchTemplateName": "Webservers",
      "VersionNumber": 3,
      "CreatedBy": "arn:aws:iam::123456789102:root",
      "LaunchTemplateData": {
        "KeyName": "kp-us-east",
        "ImageId": "ami-6057e21a",
        "InstanceType": "t2.small",
        "NetworkInterfaces": [
          {
            "SubnetId": "subnet-7b16de0c",
            "DeviceIndex": 0,
            "Groups": [
              "sg-7c227019"
            ]
          }
        ]
      },
      "DefaultVersion": false,
      "CreateTime": "2017-11-20T13:19:54.000Z"
    },
    {
      "LaunchTemplateId": "lt-068f72b72934aff71",
      "LaunchTemplateName": "Webservers",
      "VersionNumber": 2,
      "CreatedBy": "arn:aws:iam::123456789102:root",
      "LaunchTemplateData": {
        "KeyName": "kp-us-east",
        "ImageId": "ami-6057e21a",
        "InstanceType": "t2.medium",
        "NetworkInterfaces": [
          {
            "SubnetId": "subnet-1a2b3c4d",
            "DeviceIndex": 0,
            "Groups": [
              "sg-7c227019"
            ]
          }
        ]
      }
    }
  ]
}
```

```

    },
    "DefaultVersion": false,
    "CreateTime": "2017-11-20T13:12:32.000Z"
  },
  {
    "LaunchTemplateId": "lt-068f72b72934aff71",
    "LaunchTemplateName": "Webservers",
    "VersionNumber": 1,
    "CreatedBy": "arn:aws:iam::123456789102:root",
    "LaunchTemplateData": {
      "UserData": "",
      "KeyName": "kp-us-east",
      "ImageId": "ami-aabbcc11",
      "InstanceType": "t2.medium",
      "NetworkInterfaces": [
        {
          "SubnetId": "subnet-7b16de0c",
          "DeviceIndex": 0,
          "DeleteOnTermination": false,
          "Groups": [
            "sg-7c227019"
          ],
          "AssociatePublicIpAddress": true
        }
      ]
    },
    "DefaultVersion": true,
    "CreateTime": "2017-11-20T12:52:33.000Z"
  }
]
}

```

- For API details, see [DescribeLaunchTemplateVersions](#) in *AWS CLI Command Reference*.

describe-launch-templates

The following code example shows how to use describe-launch-templates.

AWS CLI

To describe launch templates

This example describes your launch templates.

Command:

```
aws ec2 describe-launch-templates
```

Output:

```
{
  "LaunchTemplates": [
    {
      "LatestVersionNumber": 2,
      "LaunchTemplateId": "lt-0e06d290751193123",
      "LaunchTemplateName": "TemplateForWebServer",
      "DefaultVersionNumber": 2,
      "CreatedBy": "arn:aws:iam::123456789012:root",
      "CreateTime": "2017-11-27T09:30:23.000Z"
    },
    {
      "LatestVersionNumber": 6,
      "LaunchTemplateId": "lt-0c45b5e061ec98456",
      "LaunchTemplateName": "DBServersTemplate",
      "DefaultVersionNumber": 1,
      "CreatedBy": "arn:aws:iam::123456789012:root",
      "CreateTime": "2017-11-20T09:25:22.000Z"
    },
    {
      "LatestVersionNumber": 1,
      "LaunchTemplateId": "lt-0d47d774e8e52dabc",
      "LaunchTemplateName": "MyLaunchTemplate2",
      "DefaultVersionNumber": 1,
      "CreatedBy": "arn:aws:iam::123456789012:root",
      "CreateTime": "2017-11-02T12:06:21.000Z"
    },
    {
      "LatestVersionNumber": 3,
      "LaunchTemplateId": "lt-01e5f948eb4f589d6",
      "LaunchTemplateName": "testingtemplate2",
      "DefaultVersionNumber": 1,
      "CreatedBy": "arn:aws:sts::123456789012:assumed-role/AdminRole/i-03ee35176e2e5aabc",
      "CreateTime": "2017-12-01T08:19:48.000Z"
    }
  ]
}
```

- For API details, see [DescribeLaunchTemplates](#) in *AWS CLI Command Reference*.

describe-local-gateway-route-table-virtual-interface-group-associations

The following code example shows how to use `describe-local-gateway-route-table-virtual-interface-group-associations`.

AWS CLI

To describe associations between virtual interface groups and local gateway route tables

The following `describe-local-gateway-route-table-virtual-interface-group-associations` example describes the associations between virtual interface groups and local gateway route tables in your AWS account.

```
aws ec2 describe-local-gateway-route-table-virtual-interface-group-associations
```

Output:

```
{
  "LocalGatewayRouteTableVirtualInterfaceGroupAssociations": [
    {
      "LocalGatewayRouteTableVirtualInterfaceGroupAssociationId": "lgw-vif-grp-assoc-07145b276bEXAMPLE",
      "LocalGatewayVirtualInterfaceGroupId": "lgw-vif-grp-07145b276bEXAMPLE",
      "LocalGatewayId": "lgw-0ab1c23d4eEXAMPLE",
      "LocalGatewayRouteTableId": "lgw-rtb-059615ef7dEXAMPLE",
      "LocalGatewayRouteTableArn": "arn:aws:ec2:us-west-2:123456789012:local-gateway-route-table/lgw-rtb-059615ef7dEXAMPLE",
      "OwnerId": "123456789012",
      "State": "associated",
      "Tags": []
    }
  ]
}
```

For more information, see [Working with local gateways](#) in the *AWS Outposts User Guide*.

- For API details, see [DescribeLocalGatewayRouteTableVirtualInterfaceGroupAssociations](#) in *AWS CLI Command Reference*.

describe-local-gateway-route-table-vpc-associations

The following code example shows how to use `describe-local-gateway-route-table-vpc-associations`.

AWS CLI

To describe the associations between VPCs and local gateway route tables

The following `describe-local-gateway-route-table-vpc-associations` example displays information about the specified association between VPCs and local gateway route tables.

```
aws ec2 describe-local-gateway-route-table-vpc-associations \
  --local-gateway-route-table-vpc-association-ids lgw-vpc-assoc-0e0f27af15EXAMPLE
```

Output:

```
{
  "LocalGatewayRouteTableVpcAssociation": {
    "LocalGatewayRouteTableVpcAssociationId": "lgw-vpc-assoc-0e0f27af1EXAMPLE",
    "LocalGatewayRouteTableId": "lgw-rtb-059615ef7dEXAMPLE",
    "LocalGatewayId": "lgw-09b493aa7cEXAMPLE",
    "VpcId": "vpc-0efe9bde08EXAMPLE",
    "State": "associated"
  }
}
```

For more information, see [Local gateway route tables](#) in the *Outposts User Guide*.

- For API details, see [DescribeLocalGatewayRouteTableVpcAssociations](#) in *AWS CLI Command Reference*.

describe-local-gateway-route-tables

The following code example shows how to use `describe-local-gateway-route-tables`.

AWS CLI

To describe your Local Gateway Route Tables

The following `describe-local-gateway-route-tables` example displays details about the local gateway route tables.

```
aws ec2 describe-local-gateway-route-tables
```

Output:

```
{
  "LocalGatewayRouteTables": [
    {
      "LocalGatewayRouteTableId": "lgw-rtb-059615ef7deEXAMPLE",
      "LocalGatewayId": "lgw-09b493aa7cEXAMPLE",
      "OutpostArn": "arn:aws:outposts:us-west-2:111122223333:outpost/
op-0dc11b66edEXAMPLE",
      "State": "available"
    }
  ]
}
```

- For API details, see [DescribeLocalGatewayRouteTables](#) in *AWS CLI Command Reference*.

`describe-local-gateway-virtual-interface-groups`

The following code example shows how to use `describe-local-gateway-virtual-interface-groups`.

AWS CLI

To describe local gateway virtual interface groups

The following `describe-local-gateway-virtual-interface-groups` example describes the local gateway virtual interface groups in your AWS account.

```
aws ec2 describe-local-gateway-virtual-interface-groups
```

Output:

```
{
  "LocalGatewayVirtualInterfaceGroups": [
    {
```

```

    "LocalGatewayVirtualInterfaceGroupId": "lgw-vif-grp-07145b276bEXAMPLE",
    "LocalGatewayVirtualInterfaceIds": [
      "lgw-vif-01a23bc4d5EXAMPLE",
      "lgw-vif-543ab21012EXAMPLE"
    ],
    "LocalGatewayId": "lgw-0ab1c23d4eEXAMPLE",
    "OwnerId": "123456789012",
    "Tags": []
  }
]
}

```

For more information, see [Working with local gateways](#) in the *AWS Outposts User Guide*.

- For API details, see [DescribeLocalGatewayVirtualInterfaceGroups](#) in *AWS CLI Command Reference*.

describe-local-gateway-virtual-interfaces

The following code example shows how to use `describe-local-gateway-virtual-interfaces`.

AWS CLI

To describe local gateway virtual interfaces

The following `describe-local-gateway-virtual-interfaces` example describes the local gateway virtual interfaces in your AWS account.

```
aws ec2 describe-local-gateway-virtual-interfaces
```

Output:

```

{
  "LocalGatewayVirtualInterfaces": [
    {
      "LocalGatewayVirtualInterfaceId": "lgw-vif-01a23bc4d5EXAMPLE",
      "LocalGatewayId": "lgw-0ab1c23d4eEXAMPLE",
      "Vlan": 2410,
      "LocalAddress": "0.0.0.0/0",
      "PeerAddress": "0.0.0.0/0",
    }
  ]
}

```

```

        "LocalBgpAsn": 65010,
        "PeerBgpAsn": 65000,
        "OwnerId": "123456789012",
        "Tags": []
    },
    {
        "LocalGatewayVirtualInterfaceId": "lgw-vif-543ab21012EXAMPLE",
        "LocalGatewayId": "lgw-0ab1c23d4eEXAMPLE",
        "Vlan": 2410,
        "LocalAddress": "0.0.0.0/0",
        "PeerAddress": "0.0.0.0/0",
        "LocalBgpAsn": 65010,
        "PeerBgpAsn": 65000,
        "OwnerId": "123456789012",
        "Tags": []
    }
]
}

```

For more information, see [Working with local gateways](#) in the *AWS Outposts User Guide*.

- For API details, see [DescribeLocalGatewayVirtualInterfaces](#) in *AWS CLI Command Reference*.

describe-local-gateways

The following code example shows how to use `describe-local-gateways`.

AWS CLI

To describe your Local Gateways

The following `describe-local-gateways` example displays details for the local gateways that are available to you.

```
aws ec2 describe-local-gateways
```

Output:

```

{
  "LocalGateways": [
    {
      "LocalGatewayId": "lgw-09b493aa7cEXAMPLE",

```

```

        "OutpostArn": "arn:aws:outposts:us-west-2:123456789012:outpost/
op-0dc11b66ed59f995a",
        "OwnerId": "123456789012",
        "State": "available"
    }
]
}

```

- For API details, see [DescribeLocalGateways](#) in *AWS CLI Command Reference*.

describe-managed-prefix-lists

The following code example shows how to use `describe-managed-prefix-lists`.

AWS CLI

To describe managed prefix lists

The following `describe-managed-prefix-lists` example describes the prefix lists owned by AWS account 123456789012.

```

aws ec2 describe-managed-prefix-lists \
  --filters Name=owner-id,Values=123456789012

```

Output:

```

{
  "PrefixLists": [
    {
      "PrefixListId": "pl-11223344556677aab",
      "AddressFamily": "IPv6",
      "State": "create-complete",
      "PrefixListArn": "arn:aws:ec2:us-west-2:123456789012:prefix-list/
pl-11223344556677aab",
      "PrefixListName": "vpc-ipv6-cidrs",
      "MaxEntries": 25,
      "Version": 1,
      "Tags": [],
      "OwnerId": "123456789012"
    },
    {
      "PrefixListId": "pl-0123456abcabcabc1",

```

```
    "AddressFamily": "IPv4",
    "State": "active",
    "PrefixListArn": "arn:aws:ec2:us-west-2:123456789012:prefix-list/
pl-0123456abcabcabc1",
    "PrefixListName": "vpc-cidrs",
    "MaxEntries": 10,
    "Version": 1,
    "Tags": [],
    "OwnerId": "123456789012"
  }
]
}
```

For more information, see [Managed prefix lists](#) in the *Amazon VPC User Guide*.

- For API details, see [DescribeManagedPrefixLists](#) in *AWS CLI Command Reference*.

describe-moving-addresses

The following code example shows how to use `describe-moving-addresses`.

AWS CLI

To describe your moving addresses

This example describes all of your moving Elastic IP addresses.

Command:

```
aws ec2 describe-moving-addresses
```

Output:

```
{
  "MovingAddressStatuses": [
    {
      "PublicIp": "198.51.100.0",
      "MoveStatus": "MovingToVpc"
    }
  ]
}
```


This example describes all addresses that are moving to the EC2-VPC platform.

Command:

```
aws ec2 describe-moving-addresses --filters Name=moving-status,Values=MovingToVpc
```

- For API details, see [DescribeMovingAddresses](#) in *AWS CLI Command Reference*.

describe-nat-gateways

The following code example shows how to use describe-nat-gateways.

AWS CLI

Example 1: To describe a public NAT gateway

The following describe-nat-gateways example describes the specified public NAT gateway.

```
aws ec2 describe-nat-gateways \
  --nat-gateway-id nat-01234567890abcdef
```

Output:

```
{
  "NatGateways": [
    {
      "CreateTime": "2023-08-25T01:56:51.000Z",
      "NatGatewayAddresses": [
        {
          "AllocationId": "eipalloc-0790180cd2EXAMPLE",
          "NetworkInterfaceId": "eni-09cc4b2558794f7f9",
          "PrivateIp": "10.0.0.211",
          "PublicIp": "54.85.121.213",
          "AssociationId": "eipassoc-04d295cc9b8815b24",
          "IsPrimary": true,
          "Status": "succeeded"
        },
        {
          "AllocationId": "eipalloc-0be6ecac95EXAMPLE",
          "NetworkInterfaceId": "eni-09cc4b2558794f7f9",
          "PrivateIp": "10.0.0.74",
```

```

        "PublicIp": "3.211.231.218",
        "AssociationId": "eipassoc-0f96bdca17EXAMPLE",
        "IsPrimary": false,
        "Status": "succeeded"
    }
],
"NatGatewayId": "nat-01234567890abcdef",
"State": "available",
"SubnetId": "subnet-655eab5f08EXAMPLE",
"VpcId": "vpc-098eb5ef58EXAMPLE",
"Tags": [
    {
        "Key": "Name",
        "Value": "public-nat"
    }
],
"ConnectivityType": "public"
}
]
}

```

Example 2: To describe a private NAT gateway

The following `describe-nat-gateways` example describes the specified private NAT gateway.

```

aws ec2 describe-nat-gateways \
  --nat-gateway-id nat-1234567890abcdef0

```

Output:

```

{
  "NatGateways": [
    {
      "CreateTime": "2023-08-25T00:50:05.000Z",
      "NatGatewayAddresses": [
        {
          "NetworkInterfaceId": "eni-0065a61b324d1897a",
          "PrivateIp": "10.0.20.240",
          "IsPrimary": true,
          "Status": "succeeded"
        }
      ],
    }
  ],
}

```

```

        {
            "NetworkInterfaceId": "eni-0065a61b324d1897a",
            "PrivateIp": "10.0.20.33",
            "IsPrimary": false,
            "Status": "succeeded"
        },
        {
            "NetworkInterfaceId": "eni-0065a61b324d1897a",
            "PrivateIp": "10.0.20.197",
            "IsPrimary": false,
            "Status": "succeeded"
        }
    ],
    "NatGatewayId": "nat-1234567890abcdef0",
    "State": "available",
    "SubnetId": "subnet-08fc749671EXAMPLE",
    "VpcId": "vpc-098eb5ef58EXAMPLE",
    "Tags": [
        {
            "Key": "Name",
            "Value": "private-nat"
        }
    ],
    "ConnectivityType": "private"
}
]
}

```

For more information, see [NAT gateways](#) in the *Amazon VPC User Guide*.

- For API details, see [DescribeNatGateways](#) in *AWS CLI Command Reference*.

describe-network-acls

The following code example shows how to use `describe-network-acls`.

AWS CLI

To describe your network ACLs

The following `describe-network-acls` example retrieves details about your network ACLs.

```
aws ec2 describe-network-acls
```

Output:

```
{
  "NetworkAcls": [
    {
      "Associations": [
        {
          "NetworkAclAssociationId": "aclassoc-0c1679dc41EXAMPLE",
          "NetworkAclId": "acl-0ea1f54ca7EXAMPLE",
          "SubnetId": "subnet-0931fc2fa5EXAMPLE"
        }
      ],
      "Entries": [
        {
          "CidrBlock": "0.0.0.0/0",
          "Egress": true,
          "Protocol": "-1",
          "RuleAction": "allow",
          "RuleNumber": 100
        },
        {
          "CidrBlock": "0.0.0.0/0",
          "Egress": true,
          "Protocol": "-1",
          "RuleAction": "deny",
          "RuleNumber": 32767
        },
        {
          "CidrBlock": "0.0.0.0/0",
          "Egress": false,
          "Protocol": "-1",
          "RuleAction": "allow",
          "RuleNumber": 100
        },
        {
          "CidrBlock": "0.0.0.0/0",
          "Egress": false,
          "Protocol": "-1",
          "RuleAction": "deny",
          "RuleNumber": 32767
        }
      ],
      "IsDefault": true,
      "NetworkAclId": "acl-0ea1f54ca7EXAMPLE",
    }
  ]
}
```

```
    "Tags": [],
    "VpcId": "vpc-06e4ab6c6cEXAMPLE",
    "OwnerId": "111122223333"
  },
  {
    "Associations": [],
    "Entries": [
      {
        "CidrBlock": "0.0.0.0/0",
        "Egress": true,
        "Protocol": "-1",
        "RuleAction": "allow",
        "RuleNumber": 100
      },
      {
        "Egress": true,
        "Ipv6CidrBlock": ":::/0",
        "Protocol": "-1",
        "RuleAction": "allow",
        "RuleNumber": 101
      },
      {
        "CidrBlock": "0.0.0.0/0",
        "Egress": true,
        "Protocol": "-1",
        "RuleAction": "deny",
        "RuleNumber": 32767
      },
      {
        "Egress": true,
        "Ipv6CidrBlock": ":::/0",
        "Protocol": "-1",
        "RuleAction": "deny",
        "RuleNumber": 32768
      },
      {
        "CidrBlock": "0.0.0.0/0",
        "Egress": false,
        "Protocol": "-1",
        "RuleAction": "allow",
        "RuleNumber": 100
      },
      {
        "Egress": false,
```

```
        "Ipv6CidrBlock": "::/0",
        "Protocol": "-1",
        "RuleAction": "allow",
        "RuleNumber": 101
    },
    {
        "CidrBlock": "0.0.0.0/0",
        "Egress": false,
        "Protocol": "-1",
        "RuleAction": "deny",
        "RuleNumber": 32767
    },
    {
        "Egress": false,
        "Ipv6CidrBlock": "::/0",
        "Protocol": "-1",
        "RuleAction": "deny",
        "RuleNumber": 32768
    }
],
"IsDefault": true,
"NetworkAclId": "acl-0e2a78e4e2EXAMPLE",
"Tags": [],
"VpcId": "vpc-03914afb3eEXAMPLE",
"OwnerId": "111122223333"
}
]
}
```

For more information, see [Network ACLs](#) in the *AWS VPC User Guide*.

- For API details, see [DescribeNetworkAcls](#) in *AWS CLI Command Reference*.

describe-network-insights-access-scope-analyses

The following code example shows how to use `describe-network-insights-access-scope-analyses`.

AWS CLI

To describe Network Insights access scope analyses

The following `describe-network-insights-access-scope-analyses` example describes the access scope analysis in your AWS account.

```
aws ec2 describe-network-insights-access-scope-analyses \
  --region us-east-1
```

Output:

```
{
  "NetworkInsightsAccessScopeAnalyses": [
    {
      "NetworkInsightsAccessScopeAnalysisId": "nisa-123456789111",
      "NetworkInsightsAccessScopeAnalysisArn": "arn:aws:ec2:us-
east-1:123456789012:network-insights-access-scope-analysis/nisa-123456789111",
      "NetworkInsightsAccessScopeId": "nis-123456789222",
      "Status": "succeeded",
      "StartDate": "2022-01-25T19:45:36.842000+00:00",
      "FindingsFound": "true",
      "Tags": []
    }
  ]
}
```

For more information, see [Getting started with Network Access Analyzer using the AWS CLI](#) in the *Network Access Analyzer Guide*.

- For API details, see [DescribeNetworkInsightsAccessScopeAnalyses](#) in *AWS CLI Command Reference*.

describe-network-insights-access-scopes

The following code example shows how to use `describe-network-insights-access-scopes`.

AWS CLI

To describe Network Insights access scopes

The following `describe-network-insights-access-scopes` example describes the access-scope analyses in your AWS account.

```
aws ec2 describe-network-insights-access-scopes \
```

```
--region us-east-1
```

Output:

```
{
  "NetworkInsightsAccessScopes": [
    {
      "NetworkInsightsAccessScopeId": "nis-123456789111",
      "NetworkInsightsAccessScopeArn": "arn:aws:ec2:us-
east-1:123456789012:network-insights-access-scope/nis-123456789111",
      "CreateDate": "2021-11-29T21:12:41.416000+00:00",
      "UpdatedDate": "2021-11-29T21:12:41.416000+00:00",
      "Tags": []
    }
  ]
}
```

For more information, see [Getting started with Network Access Analyzer using the AWS CLI](#) in the *Network Access Analyzer Guide*.

- For API details, see [DescribeNetworkInsightsAccessScopes](#) in *AWS CLI Command Reference*.

describe-network-insights-analyses

The following code example shows how to use `describe-network-insights-analyses`.

AWS CLI

To view the results of a path analysis

The following `describe-network-insights-analyses` example describes the specified analysis. In this example, the source is an internet gateway, the destination is an EC2 instance, and the protocol is TCP. The analysis succeeded (Status is succeeded) and the path is not reachable (NetworkPathFound is false). The explanation code ENI_SG_RULES_MISMATCH indicates that the security group for the instance does not contain a rule that allows traffic on the destination port.

```
aws ec2 describe-network-insights-analyses \
  --network-insights-analysis-ids nia-02207aa13eb480c7a
```

Output:


```

{
  "NetworkInsightsAnalyses": [
    {
      "NetworkInsightsAnalysisId": "nia-02207aa13eb480c7a",
      "NetworkInsightsAnalysisArn": "arn:aws:ec2:us-
east-1:123456789012:network-insights-analysis/nia-02207aa13eb480c7a",
      "NetworkInsightsPathId": "nip-0b26f224f1d131fa8",
      "StartDate": "2021-01-20T22:58:37.495Z",
      "Status": "succeeded",
      "NetworkPathFound": false,
      "Explanations": [
        {
          "Direction": "ingress",
          "ExplanationCode": "ENI_SG_RULES_MISMATCH",
          "NetworkInterface": {
            "Id": "eni-0a25edef15a6cc08c",
            "Arn": "arn:aws:ec2:us-east-1:123456789012:network-
interface/eni-0a25edef15a6cc08c"
          },
          "SecurityGroups": [
            {
              "Id": "sg-02f0d35a850ba727f",
              "Arn": "arn:aws:ec2:us-east-1:123456789012:security-
group/sg-02f0d35a850ba727f"
            }
          ],
          "Subnet": {
            "Id": "subnet-004ff41eccb4d1194",
            "Arn": "arn:aws:ec2:us-east-1:123456789012:subnet/
subnet-004ff41eccb4d1194"
          },
          "Vpc": {
            "Id": "vpc-f1663d98ad28331c7",
            "Arn": "arn:aws:ec2:us-east-1:123456789012:vpc/vpc-
f1663d98ad28331c7"
          }
        }
      ],
      "Tags": []
    }
  ]
}

```

For more information, see [Getting started using the AWS CLI](#) in the *Reachability Analyzer Guide*.

- For API details, see [DescribeNetworkInsightsAnalyses](#) in *AWS CLI Command Reference*.

describe-network-insights-paths

The following code example shows how to use `describe-network-insights-paths`.

AWS CLI

To describe a path

The following `describe-network-insights-paths` example describes the specified path.

```
aws ec2 describe-network-insights-paths \  
  --network-insights-path-ids nip-0b26f224f1d131fa8
```

Output:

```
{  
  "NetworkInsightsPaths": [  
    {  
      "NetworkInsightsPathId": "nip-0b26f224f1d131fa8",  
      "NetworkInsightsPathArn": "arn:aws:ec2:us-east-1:123456789012:network-  
insights-path/nip-0b26f224f1d131fa8",  
      "CreateDate": "2021-01-20T22:43:46.933Z",  
      "Source": "igw-0797cccdc9d73b0e5",  
      "Destination": "i-0495d385ad28331c7",  
      "Protocol": "tcp"  
    }  
  ]  
}
```

For more information, see [Getting started using the AWS CLI](#) in the *Reachability Analyzer Guide*.

- For API details, see [DescribeNetworkInsightsPaths](#) in *AWS CLI Command Reference*.

describe-network-interface-attribute

The following code example shows how to use `describe-network-interface-attribute`.

AWS CLI

To describe the attachment attribute of a network interface

This example command describes the attachment attribute of the specified network interface.

Command:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200 --attribute attachment
```

Output:

```
{
  "NetworkInterfaceId": "eni-686ea200",
  "Attachment": {
    "Status": "attached",
    "DeviceIndex": 0,
    "AttachTime": "2015-05-21T20:02:20.000Z",
    "InstanceId": "i-1234567890abcdef0",
    "DeleteOnTermination": true,
    "AttachmentId": "eni-attach-43348162",
    "InstanceOwnerId": "123456789012"
  }
}
```

To describe the description attribute of a network interface

This example command describes the description attribute of the specified network interface.

Command:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200 --attribute description
```

Output:

```
{
  "NetworkInterfaceId": "eni-686ea200",
  "Description": {
    "Value": "My description"
  }
}
```

```
}  
}
```

To describe the groupSet attribute of a network interface

This example command describes the groupSet attribute of the specified network interface.

Command:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200 --  
attribute groupSet
```

Output:

```
{  
  "NetworkInterfaceId": "eni-686ea200",  
  "Groups": [  
    {  
      "GroupName": "my-security-group",  
      "GroupId": "sg-903004f8"  
    }  
  ]  
}
```

To describe the sourceDestCheck attribute of a network interface

This example command describes the sourceDestCheck attribute of the specified network interface.

Command:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200 --  
attribute sourceDestCheck
```

Output:

```
{  
  "NetworkInterfaceId": "eni-686ea200",  
  "SourceDestCheck": {  
    "Value": true  
  }  
}
```

- For API details, see [DescribeNetworkInterfaceAttribute](#) in *AWS CLI Command Reference*.

describe-network-interface-permissions

The following code example shows how to use describe-network-interface-permissions.

AWS CLI

To describe your network interface permissions

This example describes all of your network interface permissions.

Command:

```
aws ec2 describe-network-interface-permissions
```

Output:

```
{
  "NetworkInterfacePermissions": [
    {
      "PermissionState": {
        "State": "GRANTED"
      },
      "NetworkInterfacePermissionId": "eni-perm-06fd19020ede149ea",
      "NetworkInterfaceId": "eni-b909511a",
      "Permission": "INSTANCE-ATTACH",
      "AwsAccountId": "123456789012"
    }
  ]
}
```

- For API details, see [DescribeNetworkInterfacePermissions](#) in *AWS CLI Command Reference*.

describe-network-interfaces

The following code example shows how to use describe-network-interfaces.

AWS CLI

To describe your network interfaces

This example describes all your network interfaces.

Command:

```
aws ec2 describe-network-interfaces
```

Output:

```
{
  "NetworkInterfaces": [
    {
      "Status": "in-use",
      "MacAddress": "02:2f:8f:b0:cf:75",
      "SourceDestCheck": true,
      "VpcId": "vpc-a01106c2",
      "Description": "my network interface",
      "Association": {
        "PublicIp": "203.0.113.12",
        "AssociationId": "eipassoc-0fbb766a",
        "PublicDnsName": "ec2-203-0-113-12.compute-1.amazonaws.com",
        "IpOwnerId": "123456789012"
      },
      "NetworkInterfaceId": "eni-e5aa89a3",
      "PrivateIpAddresses": [
        {
          "PrivateDnsName": "ip-10-0-1-17.ec2.internal",
          "Association": {
            "PublicIp": "203.0.113.12",
            "AssociationId": "eipassoc-0fbb766a",
            "PublicDnsName": "ec2-203-0-113-12.compute-1.amazonaws.com",
            "IpOwnerId": "123456789012"
          },
          "Primary": true,
          "PrivateIpAddress": "10.0.1.17"
        }
      ],
      "RequesterManaged": false,
      "Ipv6Addresses": [],
      "PrivateDnsName": "ip-10-0-1-17.ec2.internal",
      "AvailabilityZone": "us-east-1d",
      "Attachment": {
        "Status": "attached",
        "DeviceIndex": 1,

```

```
    "AttachTime": "2013-11-30T23:36:42.000Z",
    "InstanceId": "i-1234567890abcdef0",
    "DeleteOnTermination": false,
    "AttachmentId": "eni-attach-66c4350a",
    "InstanceOwnerId": "123456789012"
  },
  "Groups": [
    {
      "GroupName": "default",
      "GroupId": "sg-8637d3e3"
    }
  ],
  "SubnetId": "subnet-b61f49f0",
  "OwnerId": "123456789012",
  "TagSet": [],
  "PrivateIpAddress": "10.0.1.17"
},
{
  "Status": "in-use",
  "MacAddress": "02:58:f5:ef:4b:06",
  "SourceDestCheck": true,
  "VpcId": "vpc-a01106c2",
  "Description": "Primary network interface",
  "Association": {
    "PublicIp": "198.51.100.0",
    "IpOwnerId": "amazon"
  },
  "NetworkInterfaceId": "eni-f9ba99bf",
  "PrivateIpAddresses": [
    {
      "Association": {
        "PublicIp": "198.51.100.0",
        "IpOwnerId": "amazon"
      },
      "Primary": true,
      "PrivateIpAddress": "10.0.1.149"
    }
  ],
  "RequesterManaged": false,
  "Ipv6Addresses": [],
  "AvailabilityZone": "us-east-1d",
  "Attachment": {
    "Status": "attached",
    "DeviceIndex": 0,
```

```

        "AttachTime": "2013-11-30T23:35:33.000Z",
        "InstanceId": "i-0598c7d356eba48d7",
        "DeleteOnTermination": true,
        "AttachmentId": "eni-attach-1b9db777",
        "InstanceOwnerId": "123456789012"
    },
    "Groups": [
        {
            "GroupName": "default",
            "GroupId": "sg-8637d3e3"
        }
    ],
    "SubnetId": "subnet-b61f49f0",
    "OwnerId": "123456789012",
    "TagSet": [],
    "PrivateIpAddress": "10.0.1.149"
}
]
}

```

This example describes network interfaces that have a tag with the key Purpose and the value Prod.

Command:

```
aws ec2 describe-network-interfaces --filters Name=tag:Purpose,Values=Prod
```

Output:

```

{
  "NetworkInterfaces": [
    {
      "Status": "available",
      "MacAddress": "12:2c:bd:f9:bf:17",
      "SourceDestCheck": true,
      "VpcId": "vpc-8941ebec",
      "Description": "ProdENI",
      "NetworkInterfaceId": "eni-b9a5ac93",
      "PrivateIpAddresses": [
        {
          "PrivateDnsName": "ip-10-0-1-55.ec2.internal",
          "Primary": true,
          "PrivateIpAddress": "10.0.1.55"
        }
      ]
    }
  ]
}

```



```
    },
    {
      "PrivateDnsName": "ip-10-0-1-117.ec2.internal",
      "Primary": false,
      "PrivateIpAddress": "10.0.1.117"
    }
  ],
  "RequesterManaged": false,
  "PrivateDnsName": "ip-10-0-1-55.ec2.internal",
  "AvailabilityZone": "us-east-1d",
  "Ipv6Addresses": [],
  "Groups": [
    {
      "GroupName": "MySG",
      "GroupId": "sg-905002f5"
    }
  ],
  "SubnetId": "subnet-31d6c219",
  "OwnerId": "123456789012",
  "TagSet": [
    {
      "Value": "Prod",
      "Key": "Purpose"
    }
  ],
  "PrivateIpAddress": "10.0.1.55"
}
]
```

- For API details, see [DescribeNetworkInterfaces](#) in *AWS CLI Command Reference*.

describe-placement-groups

The following code example shows how to use describe-placement-groups.

AWS CLI

To describe your placement groups

This example command describes all of your placement groups.

Command:

```
aws ec2 describe-placement-groups
```

Output:

```
{
  "PlacementGroups": [
    {
      "GroupName": "my-cluster",
      "State": "available",
      "Strategy": "cluster"
    },
    ...
  ]
}
```

- For API details, see [DescribePlacementGroups](#) in *AWS CLI Command Reference*.

describe-prefix-lists

The following code example shows how to use `describe-prefix-lists`.

AWS CLI**To describe prefix lists**

This example lists all available prefix lists for the region.

Command:

```
aws ec2 describe-prefix-lists
```

Output:

```
{
  "PrefixLists": [
    {
      "PrefixListName": "com.amazonaws.us-east-1.s3",
      "Cidrs": [
        "54.231.0.0/17"
      ],
      "PrefixListId": "pl-63a5400a"
    }
  ]
}
```

```
]
}
```

- For API details, see [DescribePrefixLists](#) in *AWS CLI Command Reference*.

describe-principal-id-format

The following code example shows how to use `describe-principal-id-format`.

AWS CLI

To describe the ID format for IAM users and roles with long ID format enabled

The following `describe-principal-id-format` example describes the ID format for the root user, all IAM roles, and all IAM users with long ID format enabled.

```
aws ec2 describe-principal-id-format \
  --resource instance
```

Output:

```
{
  "Principals": [
    {
      "Arn": "arn:aws:iam::123456789012:root",
      "Statuses": [
        {
          "Deadline": "2016-12-15T00:00:00.000Z",
          "Resource": "reservation",
          "UseLongIds": true
        },
        {
          "Deadline": "2016-12-15T00:00:00.000Z",
          "Resource": "instance",
          "UseLongIds": true
        },
        {
          "Deadline": "2016-12-15T00:00:00.000Z",
          "Resource": "volume",
          "UseLongIds": true
        }
      ]
    }
  ]
}
```

```
    },  
    ...  
  ]  
}
```

- For API details, see [DescribePrincipalIdFormat](#) in *AWS CLI Command Reference*.

describe-public-ipv4-pools

The following code example shows how to use `describe-public-ipv4-pools`.

AWS CLI

To describe your public IPv4 address pools

The following `describe-public-ipv4-pools` example displays details about the address pools that were created when you provisioned public IPv4 address ranges using Bring Your Own IP Addresses (BYOIP).

```
aws ec2 describe-public-ipv4-pools
```

Output:

```
{  
  "PublicIpv4Pools": [  
    {  
      "PoolId": "ipv4pool-ec2-1234567890abcdef0",  
      "PoolAddressRanges": [  
        {  
          "FirstAddress": "203.0.113.0",  
          "LastAddress": "203.0.113.255",  
          "AddressCount": 256,  
          "AvailableAddressCount": 256  
        }  
      ],  
      "TotalAddressCount": 256,  
      "TotalAvailableAddressCount": 256  
    }  
  ]  
}
```

- For API details, see [DescribePublicIpv4Pools](#) in *AWS CLI Command Reference*.

describe-regions

The following code example shows how to use `describe-regions`.

AWS CLI

Example 1: To describe all of your enabled Regions

The following `describe-regions` example describes all of the Regions that are enabled for your account.

```
aws ec2 describe-regions
```

Output:

```
{
  "Regions": [
    {
      "Endpoint": "ec2.eu-north-1.amazonaws.com",
      "RegionName": "eu-north-1",
      "OptInStatus": "opt-in-not-required"
    },
    {
      "Endpoint": "ec2.ap-south-1.amazonaws.com",
      "RegionName": "ap-south-1",
      "OptInStatus": "opt-in-not-required"
    },
    {
      "Endpoint": "ec2.eu-west-3.amazonaws.com",
      "RegionName": "eu-west-3",
      "OptInStatus": "opt-in-not-required"
    },
    {
      "Endpoint": "ec2.eu-west-2.amazonaws.com",
      "RegionName": "eu-west-2",
      "OptInStatus": "opt-in-not-required"
    },
    {
      "Endpoint": "ec2.eu-west-1.amazonaws.com",
      "RegionName": "eu-west-1",
      "OptInStatus": "opt-in-not-required"
    },
    {
```

```
    "Endpoint": "ec2.ap-northeast-3.amazonaws.com",
    "RegionName": "ap-northeast-3",
    "OptInStatus": "opt-in-not-required"
  },
  {
    "Endpoint": "ec2.ap-northeast-2.amazonaws.com",
    "RegionName": "ap-northeast-2",
    "OptInStatus": "opt-in-not-required"
  },
  {
    "Endpoint": "ec2.ap-northeast-1.amazonaws.com",
    "RegionName": "ap-northeast-1",
    "OptInStatus": "opt-in-not-required"
  },
  {
    "Endpoint": "ec2.sa-east-1.amazonaws.com",
    "RegionName": "sa-east-1",
    "OptInStatus": "opt-in-not-required"
  },
  {
    "Endpoint": "ec2.ca-central-1.amazonaws.com",
    "RegionName": "ca-central-1",
    "OptInStatus": "opt-in-not-required"
  },
  {
    "Endpoint": "ec2.ap-southeast-1.amazonaws.com",
    "RegionName": "ap-southeast-1",
    "OptInStatus": "opt-in-not-required"
  },
  {
    "Endpoint": "ec2.ap-southeast-2.amazonaws.com",
    "RegionName": "ap-southeast-2",
    "OptInStatus": "opt-in-not-required"
  },
  {
    "Endpoint": "ec2.eu-central-1.amazonaws.com",
    "RegionName": "eu-central-1",
    "OptInStatus": "opt-in-not-required"
  },
  {
    "Endpoint": "ec2.us-east-1.amazonaws.com",
    "RegionName": "us-east-1",
    "OptInStatus": "opt-in-not-required"
  },
},
```

```
{
  "Endpoint": "ec2.us-east-2.amazonaws.com",
  "RegionName": "us-east-2",
  "OptInStatus": "opt-in-not-required"
},
{
  "Endpoint": "ec2.us-west-1.amazonaws.com",
  "RegionName": "us-west-1",
  "OptInStatus": "opt-in-not-required"
},
{
  "Endpoint": "ec2.us-west-2.amazonaws.com",
  "RegionName": "us-west-2",
  "OptInStatus": "opt-in-not-required"
}
]
```

For more information, see [Regions and Zones](#) in the *Amazon EC2 User Guide*.

Example 2: To describe enabled Regions with an endpoint whose name contains a specific string

The following `describe-regions` example describes all Regions that you have enabled that have the string "us" in the endpoint.

```
aws ec2 describe-regions \
  --filters "Name=endpoint,Values=*us*"
```

Output:

```
{
  "Regions": [
    {
      "Endpoint": "ec2.us-east-1.amazonaws.com",
      "RegionName": "us-east-1"
    },
    {
      "Endpoint": "ec2.us-east-2.amazonaws.com",
      "RegionName": "us-east-2"
    },
    {
      "Endpoint": "ec2.us-west-1.amazonaws.com",
```

```
        "RegionName": "us-west-1"
      },
      {
        "Endpoint": "ec2.us-west-2.amazonaws.com",
        "RegionName": "us-west-2"
      }
    ]
  }
}
```

For more information, see [Regions and Zones](#) in the *Amazon EC2 User Guide*.

Example 3: To describe all Regions

The following `describe-regions` example describes all available Regions, including Regions that are disabled.

```
aws ec2 describe-regions \
  --all-regions
```

Output:

```
{
  "Regions": [
    {
      "Endpoint": "ec2.eu-north-1.amazonaws.com",
      "RegionName": "eu-north-1",
      "OptInStatus": "opt-in-not-required"
    },
    {
      "Endpoint": "ec2.ap-south-1.amazonaws.com",
      "RegionName": "ap-south-1",
      "OptInStatus": "opt-in-not-required"
    },
    {
      "Endpoint": "ec2.eu-west-3.amazonaws.com",
      "RegionName": "eu-west-3",
      "OptInStatus": "opt-in-not-required"
    },
    {
      "Endpoint": "ec2.eu-west-2.amazonaws.com",
      "RegionName": "eu-west-2",
      "OptInStatus": "opt-in-not-required"
    }
  ],
}
```



```
{
  "Endpoint": "ec2.eu-west-1.amazonaws.com",
  "RegionName": "eu-west-1",
  "OptInStatus": "opt-in-not-required"
},
{
  "Endpoint": "ec2.ap-northeast-3.amazonaws.com",
  "RegionName": "ap-northeast-3",
  "OptInStatus": "opt-in-not-required"
},
{
  "Endpoint": "ec2.me-south-1.amazonaws.com",
  "RegionName": "me-south-1",
  "OptInStatus": "not-opted-in"
},
{
  "Endpoint": "ec2.ap-northeast-2.amazonaws.com",
  "RegionName": "ap-northeast-2",
  "OptInStatus": "opt-in-not-required"
},
{
  "Endpoint": "ec2.ap-northeast-1.amazonaws.com",
  "RegionName": "ap-northeast-1",
  "OptInStatus": "opt-in-not-required"
},
{
  "Endpoint": "ec2.sa-east-1.amazonaws.com",
  "RegionName": "sa-east-1",
  "OptInStatus": "opt-in-not-required"
},
{
  "Endpoint": "ec2.ca-central-1.amazonaws.com",
  "RegionName": "ca-central-1",
  "OptInStatus": "opt-in-not-required"
},
{
  "Endpoint": "ec2.ap-east-1.amazonaws.com",
  "RegionName": "ap-east-1",
  "OptInStatus": "not-opted-in"
},
{
  "Endpoint": "ec2.ap-southeast-1.amazonaws.com",
  "RegionName": "ap-southeast-1",
  "OptInStatus": "opt-in-not-required"
}
```

```
    },
    {
      "Endpoint": "ec2.ap-southeast-2.amazonaws.com",
      "RegionName": "ap-southeast-2",
      "OptInStatus": "opt-in-not-required"
    },
    {
      "Endpoint": "ec2.eu-central-1.amazonaws.com",
      "RegionName": "eu-central-1",
      "OptInStatus": "opt-in-not-required"
    },
    {
      "Endpoint": "ec2.us-east-1.amazonaws.com",
      "RegionName": "us-east-1",
      "OptInStatus": "opt-in-not-required"
    },
    {
      "Endpoint": "ec2.us-east-2.amazonaws.com",
      "RegionName": "us-east-2",
      "OptInStatus": "opt-in-not-required"
    },
    {
      "Endpoint": "ec2.us-west-1.amazonaws.com",
      "RegionName": "us-west-1",
      "OptInStatus": "opt-in-not-required"
    },
    {
      "Endpoint": "ec2.us-west-2.amazonaws.com",
      "RegionName": "us-west-2",
      "OptInStatus": "opt-in-not-required"
    }
  ]
}
```

For more information, see [Regions and Zones](#) in the *Amazon EC2 User Guide*.

Example 4: To list the Region names only

The following `describe-regions` example uses the `--query` parameter to filter the output and return only the names of the Regions as text.

```
aws ec2 describe-regions \
  --all-regions \
```

```
--query "Regions[].{Name:RegionName}" \  
--output text
```

Output:

```
eu-north-1  
ap-south-1  
eu-west-3  
eu-west-2  
eu-west-1  
ap-northeast-3  
ap-northeast-2  
me-south-1  
ap-northeast-1  
sa-east-1  
ca-central-1  
ap-east-1  
ap-southeast-1  
ap-southeast-2  
eu-central-1  
us-east-1  
us-east-2  
us-west-1  
us-west-2
```

For more information, see [Regions and Zones](#) in the *Amazon EC2 User Guide*.

- For API details, see [DescribeRegions](#) in *AWS CLI Command Reference*.

describe-replace-root-volume-tasks

The following code example shows how to use `describe-replace-root-volume-tasks`.

AWS CLI

Example 1: To view information about a specific root volume replacement task

The following `describe-replace-root-volume-tasks` example describes root volume replacement task `replacevol-0111122223333abcd`.

```
aws ec2 describe-replace-root-volume-tasks \  
--replace-root-volume-task-ids replacevol-0111122223333abcd
```

Output:

```
{
  "ReplaceRootVolumeTasks": [
    {
      "ReplaceRootVolumeTaskId": "replacevol-0111122223333abcd",
      "Tags": [],
      "InstanceId": "i-0123456789abcdefa",
      "TaskState": "succeeded",
      "StartTime": "2022-03-14T15:16:28Z",
      "CompleteTime": "2022-03-14T15:16:52Z"
    }
  ]
}
```

For more information, see [Replace a root volume](#) in the *Amazon Elastic Compute Cloud User Guide*.

Example 2: To view information about all root volume replacement tasks for a specific instance

The following `describe-replace-root-volume-tasks` example describes all of the root volume replacement tasks for instance `i-0123456789abcdefa`.

```
aws ec2 describe-replace-root-volume-tasks \
  --filters Name=instance-id,Values=i-0123456789abcdefa
```

Output:

```
{
  "ReplaceRootVolumeTasks": [
    {
      "ReplaceRootVolumeTaskId": "replacevol-0111122223333abcd",
      "Tags": [],
      "InstanceId": "i-0123456789abcdefa",
      "TaskState": "succeeded",
      "StartTime": "2022-03-14T15:06:38Z",
      "CompleteTime": "2022-03-14T15:07:03Z"
    },
    {
      "ReplaceRootVolumeTaskId": "replacevol-0444455555555abcd",
      "Tags": [],

```

```
        "InstanceId": "i-0123456789abcdefa",
        "TaskState": "succeeded",
        "StartTime": "2022-03-14T15:16:28Z",
        "CompleteTime": "2022-03-14T15:16:52Z"
    }
]
}
```

For more information, see [Replace a root volume](#) in the *Amazon Elastic Compute Cloud User Guide*.

- For API details, see [DescribeReplaceRootVolumeTasks](#) in *AWS CLI Command Reference*.

describe-reserved-instances-listings

The following code example shows how to use `describe-reserved-instances-listings`.

AWS CLI

To describe a Reserved Instance listing

The following `describe-reserved-instances-listings` example retrieves information about the specified Reserved Instance listing.

```
aws ec2 describe-reserved-instances-listings \
    --reserved-instances-listing-id 5ec28771-05ff-4b9b-aa31-9e57dexample
```

This command produces no output.

- For API details, see [DescribeReservedInstancesListings](#) in *AWS CLI Command Reference*.

describe-reserved-instances-modifications

The following code example shows how to use `describe-reserved-instances-modifications`.

AWS CLI

To describe Reserved Instances modifications

This example command describes all the Reserved Instances modification requests that have been submitted for your account.

Command:

```
aws ec2 describe-reserved-instances-modifications
```

Output:

```
{
  "ReservedInstancesModifications": [
    {
      "Status": "fulfilled",
      "ModificationResults": [
        {
          "ReservedInstancesId": "93bbca2-62f1-4d9d-b225-16bada29e6c7",
          "TargetConfiguration": {
            "AvailabilityZone": "us-east-1b",
            "InstanceType": "m1.large",
            "InstanceCount": 3
          }
        },
        {
          "ReservedInstancesId": "1ba8e2e3-aabb-46c3-bcf5-3fe2fda922e6",
          "TargetConfiguration": {
            "AvailabilityZone": "us-east-1d",
            "InstanceType": "m1.xlarge",
            "InstanceCount": 1
          }
        }
      ],
      "EffectiveDate": "2015-08-12T17:00:00.000Z",
      "CreateDate": "2015-08-12T17:52:52.630Z",
      "UpdateDate": "2015-08-12T18:08:06.698Z",
      "ClientToken": "c9adb218-3222-4889-8216-0cf0e52dc37e",
      "ReservedInstancesModificationId": "rimod-d3ed4335-b1d3-4de6-ab31-0f13aaf46687",
      "ReservedInstancesIds": [
        {
          "ReservedInstancesId": "b847fa93-e282-4f55-b59a-1342f5bd7c02"
        }
      ]
    }
  ]
}
```

- For API details, see [DescribeReservedInstancesModifications](#) in *AWS CLI Command Reference*.

describe-reserved-instances-offerings

The following code example shows how to use `describe-reserved-instances-offerings`.

AWS CLI

To describe Reserved Instances offerings

This example command describes all Reserved Instances available for purchase in the region.

Command:

```
aws ec2 describe-reserved-instances-offerings
```

Output:

```
{
  "ReservedInstancesOfferings": [
    {
      "OfferingType": "Partial Upfront",
      "AvailabilityZone": "us-east-1b",
      "InstanceTenancy": "default",
      "PricingDetails": [],
      "ProductDescription": "Red Hat Enterprise Linux",
      "UsagePrice": 0.0,
      "RecurringCharges": [
        {
          "Amount": 0.088,
          "Frequency": "Hourly"
        }
      ],
      "Marketplace": false,
      "CurrencyCode": "USD",
      "FixedPrice": 631.0,
      "Duration": 94608000,
      "ReservedInstancesOfferingId": "9a06095a-bdc6-47fe-a94a-2a382f016040",
      "InstanceType": "c1.medium"
    },
    {
      "OfferingType": "PartialUpfront",
```

```

    "AvailabilityZone": "us-east-1b",
    "InstanceTenancy": "default",
    "PricingDetails": [],
    "ProductDescription": "Linux/UNIX",
    "UsagePrice": 0.0,
    "RecurringCharges": [
      {
        "Amount": 0.028,
        "Frequency": "Hourly"
      }
    ],
    "Marketplace": false,
    "CurrencyCode": "USD",
    "FixedPrice": 631.0,
    "Duration": 94608000,
    "ReservedInstancesOfferingId": "bfbefc6c-0d10-418d-b144-7258578d329d",
    "InstanceType": "c1.medium"
  },
  ...
}

```

To describe your Reserved Instances offerings using options

This example lists Reserved Instances offered by AWS with the following specifications: t1.micro instance types, Windows (Amazon VPC) product, and Heavy Utilization offerings.

Command:

```
aws ec2 describe-reserved-instances-offerings --no-include-marketplace --instance-type "t1.micro" --product-description "Windows (Amazon VPC)" --offering-type "no upfront"
```

Output:

```

{
  "ReservedInstancesOfferings": [
    {
      "OfferingType": "No Upfront",
      "AvailabilityZone": "us-east-1b",
      "InstanceTenancy": "default",
      "PricingDetails": [],
      "ProductDescription": "Windows",
      "UsagePrice": 0.0,

```



```

    "RecurringCharges": [
      {
        "Amount": 0.015,
        "Frequency": "Hourly"
      }
    ],
    "Marketplace": false,
    "CurrencyCode": "USD",
    "FixedPrice": 0.0,
    "Duration": 31536000,
    "ReservedInstancesOfferingId": "c48ab04c-fe69-4f94-8e39-a23842292823",
    "InstanceType": "t1.micro"
  },
  ...
  {
    "OfferingType": "No Upfront",
    "AvailabilityZone": "us-east-1d",
    "InstanceTenancy": "default",
    "PricingDetails": [],
    "ProductDescription": "Windows (Amazon VPC)",
    "UsagePrice": 0.0,
    "RecurringCharges": [
      {
        "Amount": 0.015,
        "Frequency": "Hourly"
      }
    ],
    "Marketplace": false,
    "CurrencyCode": "USD",
    "FixedPrice": 0.0,
    "Duration": 31536000,
    "ReservedInstancesOfferingId": "3a98bf7d-2123-42d4-b4f5-8dbec4b06dc6",
    "InstanceType": "t1.micro"
  }
]
}

```

- For API details, see [DescribeReservedInstancesOfferings](#) in *AWS CLI Command Reference*.

describe-reserved-instances

The following code example shows how to use `describe-reserved-instances`.

AWS CLI

To describe your Reserved Instances

This example command describes the Reserved Instances that you own.

Command:

```
aws ec2 describe-reserved-instances
```

Output:

```
{
  "ReservedInstances": [
    {
      "ReservedInstancesId": "b847fa93-e282-4f55-b59a-1342fexample",
      "OfferingType": "No Upfront",
      "AvailabilityZone": "us-west-1c",
      "End": "2016-08-14T21:34:34.000Z",
      "ProductDescription": "Linux/UNIX",
      "UsagePrice": 0.00,
      "RecurringCharges": [
        {
          "Amount": 0.104,
          "Frequency": "Hourly"
        }
      ],
      "Start": "2015-08-15T21:34:35.086Z",
      "State": "active",
      "FixedPrice": 0.0,
      "CurrencyCode": "USD",
      "Duration": 31536000,
      "InstanceTenancy": "default",
      "InstanceType": "m3.medium",
      "InstanceCount": 2
    },
    ...
  ]
}
```

To describe your Reserved Instances using filters

This example filters the response to include only three-year, t2.micro Linux/UNIX Reserved Instances in us-west-1c.

Command:

```
aws ec2 describe-reserved-instances --filters Name=duration,Values=94608000
Name=instance-type,Values=t2.micro Name=product-description,Values=Linux/UNIX
Name=availability-zone,Values=us-east-1e
```

Output:

```
{
  "ReservedInstances": [
    {
      "ReservedInstancesId": "f127bd27-edb7-44c9-a0eb-0d7e09259af0",
      "OfferingType": "All Upfront",
      "AvailabilityZone": "us-east-1e",
      "End": "2018-03-26T21:34:34.000Z",
      "ProductDescription": "Linux/UNIX",
      "UsagePrice": 0.00,
      "RecurringCharges": [],
      "Start": "2015-03-27T21:34:35.848Z",
      "State": "active",
      "FixedPrice": 151.0,
      "CurrencyCode": "USD",
      "Duration": 94608000,
      "InstanceTenancy": "default",
      "InstanceType": "t2.micro",
      "InstanceCount": 1
    }
  ]
}
```

For more information, see *Using Amazon EC2 Instances in the AWS Command Line Interface User Guide*.

- For API details, see [DescribeReservedInstances](#) in *AWS CLI Command Reference*.

describe-route-tables

The following code example shows how to use describe-route-tables.

AWS CLI

To describe your route tables

The following `describe-route-tables` example retrieves the details about your route tables

```
aws ec2 describe-route-tables
```

Output:

```
{
  "RouteTables": [
    {
      "Associations": [
        {
          "Main": true,
          "RouteTableAssociationId": "rtbassoc-0df3f54e06EXAMPLE",
          "RouteTableId": "rtb-09ba434c1bEXAMPLE"
        }
      ],
      "PropagatingVgws": [],
      "RouteTableId": "rtb-09ba434c1bEXAMPLE",
      "Routes": [
        {
          "DestinationCidrBlock": "10.0.0.0/16",
          "GatewayId": "local",
          "Origin": "CreateRouteTable",
          "State": "active"
        },
        {
          "DestinationCidrBlock": "0.0.0.0/0",
          "NatGatewayId": "nat-06c018cbd8EXAMPLE",
          "Origin": "CreateRoute",
          "State": "blackhole"
        }
      ],
      "Tags": [],
      "VpcId": "vpc-0065acced4EXAMPLE",
      "OwnerId": "111122223333"
    },
    {
      "Associations": [
```

```
        "Main": true,
        "RouteTableAssociationId": "rtbassoc-9EXAMPLE",
        "RouteTableId": "rtb-a1eec7de"
    }
],
"PropagatingVgws": [],
"RouteTableId": "rtb-a1eec7de",
"Routes": [
    {
        "DestinationCidrBlock": "172.31.0.0/16",
        "GatewayId": "local",
        "Origin": "CreateRouteTable",
        "State": "active"
    },
    {
        "DestinationCidrBlock": "0.0.0.0/0",
        "GatewayId": "igw-fEXAMPLE",
        "Origin": "CreateRoute",
        "State": "active"
    }
],
"Tags": [],
"VpcId": "vpc-3EXAMPLE",
"OwnerId": "111122223333"
},
{
    "Associations": [
        {
            "Main": false,
            "RouteTableAssociationId": "rtbassoc-0b100c28b2EXAMPLE",
            "RouteTableId": "rtb-07a98f76e5EXAMPLE",
            "SubnetId": "subnet-0d3d002af8EXAMPLE"
        }
    ],
    "PropagatingVgws": [],
    "RouteTableId": "rtb-07a98f76e5EXAMPLE",
    "Routes": [
        {
            "DestinationCidrBlock": "10.0.0.0/16",
            "GatewayId": "local",
            "Origin": "CreateRouteTable",
            "State": "active"
        }
    ],
    {
```

```

        "DestinationCidrBlock": "0.0.0.0/0",
        "GatewayId": "igw-06cf664d80EXAMPLE",
        "Origin": "CreateRoute",
        "State": "active"
    }
],
"Tags": [],
"VpcId": "vpc-0065acced4EXAMPLE",
"OwnerId": "111122223333"
}
]
}

```

For more information, see [Working with Route Tables](#) in the *AWS VPC User Guide*.

- For API details, see [DescribeRouteTables](#) in *AWS CLI Command Reference*.

describe-scheduled-instance-availability

The following code example shows how to use `describe-scheduled-instance-availability`.

AWS CLI

To describe an available schedule

This example describes a schedule that occurs every week on Sunday, starting on the specified date.

Command:

```

aws ec2 describe-scheduled-instance-availability --recurrence
Frequency=Weekly,Interval=1,OccurrenceDays=[1] --first-slot-start-time-range
EarliestTime=2016-01-31T00:00:00Z,LatestTime=2016-01-31T04:00:00Z

```

Output:

```

{
  "ScheduledInstanceAvailabilitySet": [
    {
      "AvailabilityZone": "us-west-2b",
      "TotalScheduledInstanceHours": 1219,
      "PurchaseToken": "eyJ2IjoiMSIsInMiOiJEsImMiOi...",
    }
  ]
}

```

```

    "MinTermDurationInDays": 366,
    "AvailableInstanceCount": 20,
    "Recurrence": {
      "OccurrenceDaySet": [
        1
      ],
      "Interval": 1,
      "Frequency": "Weekly",
      "OccurrenceRelativeToEnd": false
    },
    "Platform": "Linux/UNIX",
    "FirstSlotStartTime": "2016-01-31T00:00:00Z",
    "MaxTermDurationInDays": 366,
    "SlotDurationInHours": 23,
    "NetworkPlatform": "EC2-VPC",
    "InstanceType": "c4.large",
    "HourlyPrice": "0.095"
  },
  ...
]
}

```

To narrow the results, you can add filters that specify the operating system, network, and instance type.

Command:

```
--filters Name=platform,Values=Linux/UNIX Name=network-platform,Values=EC2-VPC
Name=instance-type,Values=c4.large
```

- For API details, see [DescribeScheduledInstanceAvailability](#) in *AWS CLI Command Reference*.

describe-scheduled-instances

The following code example shows how to use `describe-scheduled-instances`.

AWS CLI

To describe your Scheduled Instances

This example describes the specified Scheduled Instance.

Command:

```
aws ec2 describe-scheduled-instances --scheduled-instance-ids
sci-1234-1234-1234-1234-123456789012
```

Output:

```
{
  "ScheduledInstanceSet": [
    {
      "AvailabilityZone": "us-west-2b",
      "ScheduledInstanceId": "sci-1234-1234-1234-1234-123456789012",
      "HourlyPrice": "0.095",
      "CreateDate": "2016-01-25T21:43:38.612Z",
      "Recurrence": {
        "OccurrenceDaySet": [
          1
        ],
        "Interval": 1,
        "Frequency": "Weekly",
        "OccurrenceRelativeToEnd": false,
        "OccurrenceUnit": ""
      },
      "Platform": "Linux/UNIX",
      "TermEndDate": "2017-01-31T09:00:00Z",
      "InstanceCount": 1,
      "SlotDurationInHours": 32,
      "TermStartDate": "2016-01-31T09:00:00Z",
      "NetworkPlatform": "EC2-VPC",
      "TotalScheduledInstanceHours": 1696,
      "NextSlotStartTime": "2016-01-31T09:00:00Z",
      "InstanceType": "c4.large"
    }
  ]
}
```

This example describes all your Scheduled Instances.

Command:

```
aws ec2 describe-scheduled-instances
```

- For API details, see [DescribeScheduledInstances](#) in *AWS CLI Command Reference*.

describe-security-group-references

The following code example shows how to use `describe-security-group-references`.

AWS CLI

To describe security group references

This example describes the security group references for `sg-bbbb2222`. The response indicates that security group `sg-bbbb2222` is being referenced by a security group in VPC `vpc-aaaaaaaa`.

Command:

```
aws ec2 describe-security-group-references --group-id sg-bbbbb22222
```

Output:

```
{
  "SecurityGroupsReferenceSet": [
    {
      "ReferencingVpcId": "vpc-aaaaaaaa ",
      "GroupId": "sg-bbbbb22222",
      "VpcPeeringConnectionId": "pcx-b04deed9"
    }
  ]
}
```

- For API details, see [DescribeSecurityGroupReferences](#) in *AWS CLI Command Reference*.

describe-security-group-rules

The following code example shows how to use `describe-security-group-rules`.

AWS CLI

Example 1: To describe the security group rules for a security group

The following `describe-security-group-rules` example describes the security group rules of a specified security group. Use the `filters` option to scope the results to a specific security group.

```
aws ec2 describe-security-group-rules \
  --filters Name="group-id",Values="sg-1234567890abcdef0"
```

Output:

```
{
  "SecurityGroupRules": [
    {
      "SecurityGroupRuleId": "sgr-abcdef01234567890",
      "GroupId": "sg-1234567890abcdef0",
      "GroupOwnerId": "111122223333",
      "IsEgress": false,
      "IpProtocol": "-1",
      "FromPort": -1,
      "ToPort": -1,
      "ReferencedGroupInfo": {
        "GroupId": "sg-1234567890abcdef0",
        "UserId": "111122223333"
      },
      "Tags": []
    },
    {
      "SecurityGroupRuleId": "sgr-bcdef01234567890a",
      "GroupId": "sg-1234567890abcdef0",
      "GroupOwnerId": "111122223333",
      "IsEgress": true,
      "IpProtocol": "-1",
      "FromPort": -1,
      "ToPort": -1,
      "CidrIpv6": "::/0",
      "Tags": []
    },
    {
      "SecurityGroupRuleId": "sgr-cdef01234567890ab",
      "GroupId": "sg-1234567890abcdef0",
      "GroupOwnerId": "111122223333",
      "IsEgress": true,
      "IpProtocol": "-1",
      "FromPort": -1,
      "ToPort": -1,
      "CidrIpv4": "0.0.0.0/0",
      "Tags": []
    }
  ]
}
```

```
]
}
```

Example 2: To describe a security group rule

The following `describe-security-group-rules` example describes the specified security group rule.

```
aws ec2 describe-security-group-rules \
  --security-group-rule-ids sgr-cdef01234567890ab
```

Output:

```
{
  "SecurityGroupRules": [
    {
      "SecurityGroupId": "sgr-cdef01234567890ab",
      "GroupId": "sg-1234567890abcdef0",
      "GroupOwnerId": "111122223333",
      "IsEgress": true,
      "IpProtocol": "-1",
      "FromPort": -1,
      "ToPort": -1,
      "CidrIpv4": "0.0.0.0/0",
      "Tags": []
    }
  ]
}
```

For more information, see [Security group rules](#) in the *Amazon VPC User Guide*.

- For API details, see [DescribeSecurityGroupRules](#) in *AWS CLI Command Reference*.

describe-security-groups

The following code example shows how to use `describe-security-groups`.

AWS CLI

Example 1: To describe a security group

The following `describe-security-groups` example describes the specified security group.

```
aws ec2 describe-security-groups \  
  --group-ids sg-903004f8
```

Output:

```
{  
  "SecurityGroups": [  
    {  
      "IpPermissionsEgress": [  
        {  
          "IpProtocol": "-1",  
          "IpRanges": [  
            {  
              "CidrIp": "0.0.0.0/0"  
            }  
          ],  
          "UserIdGroupPairs": [],  
          "PrefixListIds": []  
        }  
      ],  
      "Description": "My security group",  
      "Tags": [  
        {  
          "Value": "SG1",  
          "Key": "Name"  
        }  
      ],  
      "IpPermissions": [  
        {  
          "IpProtocol": "-1",  
          "IpRanges": [],  
          "UserIdGroupPairs": [  
            {  
              "UserId": "123456789012",  
              "GroupId": "sg-903004f8"  
            }  
          ],  
          "PrefixListIds": []  
        }  
      ],  
      {  
        "PrefixListIds": [],  
        "FromPort": 22,  
        "IpRanges": [  

```

```
        {
            "Description": "Access from NY office",
            "CidrIp": "203.0.113.0/24"
        },
        {
            "ToPort": 22,
            "IpProtocol": "tcp",
            "UserIdGroupPairs": []
        }
    ],
    "GroupName": "MySecurityGroup",
    "VpcId": "vpc-1a2b3c4d",
    "OwnerId": "123456789012",
    "GroupId": "sg-903004f8",
}
]
```

Example 2: To describe security groups that have specific rules

The following `describe-security-groups` example uses filters to scope the results to security groups that have a rule that allows SSH traffic (port 22) and a rule that allows traffic from all addresses (`0.0.0.0/0`). The example uses the `--query` parameter to display only the names of the security groups. Security groups must match all filters to be returned in the results; however, a single rule does not have to match all filters. For example, the output returns a security group with a rule that allows SSH traffic from a specific IP address and another rule that allows HTTP traffic from all addresses.

```
aws ec2 describe-security-groups \
  --filters Name=ip-permission.from-port,Values=22 Name=ip-permission.to-
port,Values=22 Name=ip-permission.cidr,Values='0.0.0.0/0' \
  --query "SecurityGroups[*].[GroupName]" \
  --output text
```

Output:

```
default
my-security-group
web-servers
launch-wizard-1
```

Example 3: To describe security groups based on tags

The following `describe-security-groups` example uses filters to scope the results to security groups that include `test` in the security group name, and that have the tag `Test=To-delete`. The example uses the `--query` parameter to display only the names and IDs of the security groups.

```
aws ec2 describe-security-groups \
  --filters Name=group-name,Values=*test* Name=tag:Test,Values=To-delete \
  --query "SecurityGroups[*].{Name:GroupName, ID:GroupId}"
```

Output:

```
[
  {
    "Name": "testfornewinstance",
    "ID": "sg-33bb22aa"
  },
  {
    "Name": "newgrouptest",
    "ID": "sg-1a2b3c4d"
  }
]
```

For additional examples using tag filters, see [Working with tags](#) in the *Amazon EC2 User Guide*.

- For API details, see [DescribeSecurityGroups](#) in *AWS CLI Command Reference*.

`describe-snapshot-attribute`

The following code example shows how to use `describe-snapshot-attribute`.

AWS CLI

To describe the snapshot attributes for a snapshot

The following `describe-snapshot-attribute` example lists the accounts with which a snapshot is shared.

```
aws ec2 describe-snapshot-attribute \
  --snapshot-id snap-01234567890abcdef \
```

```
--attribute createVolumePermission
```

Output:

```
{
  "SnapshotId": "snap-01234567890abcdef",
  "CreateVolumePermissions": [
    {
      "UserId": "123456789012"
    }
  ]
}
```

For more information, see [Share an Amazon EBS snapshot](#) in the *Amazon Elastic Compute Cloud User Guide*.

- For API details, see [DescribeSnapshotAttribute](#) in *AWS CLI Command Reference*.

describe-snapshot-tier-status

The following code example shows how to use `describe-snapshot-tier-status`.

AWS CLI

To view archival information about an archived snapshot

The following `describe-snapshot-tier-status` example provides archival information about an archived snapshot.

```
aws ec2 describe-snapshot-tier-status \
  --filters "Name=snapshot-id, Values=snap-01234567890abcdef"
```

Output:

```
{
  "SnapshotTierStatuses": [
    {
      "Status": "completed",
      "ArchivalCompleteTime": "2021-09-15T17:33:16.147Z",
      "LastTieringProgress": 100,
      "Tags": [],
      "VolumeId": "vol-01234567890abcdef",
    }
  ]
}
```

```

        "LastTieringOperationState": "archival-completed",
        "StorageTier": "archive",
        "OwnerId": "123456789012",
        "SnapshotId": "snap-01234567890abcdef",
        "LastTieringStartTime": "2021-09-15T16:44:37.574Z"
    }
]
}

```

For more information, see [View archived snapshots](#) in the *Amazon Elastic Compute Cloud User Guide*.

- For API details, see [DescribeSnapshotTierStatus](#) in *AWS CLI Command Reference*.

describe-snapshots

The following code example shows how to use describe-snapshots.

AWS CLI

Example 1: To describe a snapshot

The following describe-snapshots example describes the specified snapshot.

```

aws ec2 describe-snapshots \
  --snapshot-ids snap-1234567890abcdef0

```

Output:

```

{
  "Snapshots": [
    {
      "Description": "This is my snapshot",
      "Encrypted": false,
      "VolumeId": "vol-049df61146c4d7901",
      "State": "completed",
      "VolumeSize": 8,
      "StartTime": "2019-02-28T21:28:32.000Z",
      "Progress": "100%",
      "OwnerId": "012345678910",
      "SnapshotId": "snap-01234567890abcdef",
      "Tags": [
        {

```



```
        "Key": "Stack",
        "Value": "test"
      }
    ]
  }
}
```

For more information, see [Amazon EBS snapshots](#) in the *Amazon EC2 User Guide*.

Example 2: To describe snapshots based on filters

The following `describe-snapshots` example uses filters to scope the results to snapshots owned by your AWS account that are in the pending state. The example uses the `--query` parameter to display only the snapshot IDs and the time the snapshot was started.

```
aws ec2 describe-snapshots \
  --owner-ids self \
  --filters Name=status,Values=pending \
  --query "Snapshots[*].{ID:SnapshotId,Time:StartTime}"
```

Output:

```
[
  {
    "ID": "snap-1234567890abcdef0",
    "Time": "2019-08-04T12:48:18.000Z"
  },
  {
    "ID": "snap-066877671789bd71b",
    "Time": "2019-08-04T02:45:16.000Z"
  },
  ...
]
```

The following `describe-snapshots` example uses filters to scope the results to snapshots created from the specified volume. The example uses the `--query` parameter to display only the snapshot IDs.

```
aws ec2 describe-snapshots \
  --filters Name=volume-id,Values=049df61146c4d7901 \
  --query "Snapshots[*].[SnapshotId]" \
```

```
--output text
```

Output:

```
snap-1234567890abcdef0  
snap-08637175a712c3fb9  
...
```

For additional examples using filters, see [Listing and filtering your resources](#) in the *Amazon EC2 User Guide*.

Example 3: To describe snapshots based on tags

The following `describe-snapshots` example uses tag filters to scope the results to snapshots that have the tag `Stack=Prod`.

```
aws ec2 describe-snapshots \  
  --filters Name=tag:Stack,Values=prod
```

For an example of the output for `describe-snapshots`, see Example 1.

For additional examples using tag filters, see [Working with tags](#) in the *Amazon EC2 User Guide*.

Example 4: To describe snapshots based on age

The following `describe-snapshots` example uses JMESPath expressions to describe all snapshots created by your AWS account before the specified date. It displays only the snapshot IDs.

```
aws ec2 describe-snapshots \  
  --owner-ids 012345678910 \  
  --query "Snapshots[?(StartTime<='2020-03-31')].[SnapshotId]"
```

For additional examples using filters, see [Listing and filtering your resources](#) in the *Amazon EC2 User Guide*.

Example 5: To view only archived snapshots

The following `describe-snapshots` example lists only snapshots that are stored in the archive tier.

```
aws ec2 describe-snapshots \  
  --filters Name=archive-tier,Values=true
```

```
--filters "Name=storage-tier,Values=archive"
```

Output:

```
{
  "Snapshots": [
    {
      "Description": "Snap A",
      "Encrypted": false,
      "VolumeId": "vol-01234567890aaaaaa",
      "State": "completed",
      "VolumeSize": 8,
      "StartTime": "2021-09-07T21:00:00.000Z",
      "Progress": "100%",
      "OwnerId": "123456789012",
      "SnapshotId": "snap-01234567890aaaaaa",
      "StorageTier": "archive",
      "Tags": []
    },
  ]
}
```

For more information, see [View archived snapshots](#) in the *Amazon Elastic Compute Cloud User Guide*.

- For API details, see [DescribeSnapshots](#) in *AWS CLI Command Reference*.

describe-spot-datafeed-subscription

The following code example shows how to use `describe-spot-datafeed-subscription`.

AWS CLI**To describe Spot Instance datafeed subscription for an account**

This example command describes the data feed for the account.

Command:

```
aws ec2 describe-spot-datafeed-subscription
```

Output:

```
{
  "SpotDatafeedSubscription": {
    "OwnerId": "123456789012",
    "Prefix": "spotdata",
    "Bucket": "my-s3-bucket",
    "State": "Active"
  }
}
```

- For API details, see [DescribeSpotDatafeedSubscription](#) in *AWS CLI Command Reference*.

describe-spot-fleet-instances

The following code example shows how to use `describe-spot-fleet-instances`.

AWS CLI

To describe the Spot Instances associated with a Spot fleet

This example command lists the Spot instances associated with the specified Spot fleet.

Command:

```
aws ec2 describe-spot-fleet-instances --spot-fleet-request-id sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

Output:

```
{
  "ActiveInstances": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "InstanceType": "m3.medium",
      "SpotInstanceRequestId": "sir-08b93456"
    },
    ...
  ],
  "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE"
}
```

- For API details, see [DescribeSpotFleetInstances](#) in *AWS CLI Command Reference*.

describe-spot-fleet-request-history

The following code example shows how to use `describe-spot-fleet-request-history`.

AWS CLI

To describe Spot fleet history

This example command returns the history for the specified Spot fleet starting at the specified time.

Command:

```
aws ec2 describe-spot-fleet-request-history --spot-fleet-request-id sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE --start-time 2015-05-26T00:00:00Z
```

The following example output shows the successful launches of two Spot Instances for the Spot fleet.

Output:

```
{
  "HistoryRecords": [
    {
      "Timestamp": "2015-05-26T23:17:20.697Z",
      "EventInformation": {
        "EventSubType": "submitted"
      },
      "EventType": "fleetRequestChange"
    },
    {
      "Timestamp": "2015-05-26T23:17:20.873Z",
      "EventInformation": {
        "EventSubType": "active"
      },
      "EventType": "fleetRequestChange"
    },
    {
      "Timestamp": "2015-05-26T23:21:21.712Z",
      "EventInformation": {
        "InstanceId": "i-1234567890abcdef0",
        "EventSubType": "launched"
      },
    },
  ],
}
```

```

    "EventType": "instanceChange"
  },
  {
    "Timestamp": "2015-05-26T23:21:21.816Z",
    "EventInformation": {
      "InstanceId": "i-1234567890abcdef1",
      "EventSubType": "launched"
    },
    "EventType": "instanceChange"
  }
],
"SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
"NextToken": "CpHNsscimcV5oH7bSsub03CI2Qms5+ypNpNm
+53MNlR0YcXAkp0xFlfKf91yVxSExmbtma3awYxMFzNA663ZskT0AHtJ6TCb2Z8bQC2EnZgyELbymtWPfpZ1ZbauVg
+P+TfG1WxWWB/Vr5dk5d4LfdgA/DRAHUrYgxzrEXAMPLE=",
"StartTime": "2015-05-26T00:00:00Z"
}

```

- For API details, see [DescribeSpotFleetRequestHistory](#) in *AWS CLI Command Reference*.

describe-spot-fleet-requests

The following code example shows how to use `describe-spot-fleet-requests`.

AWS CLI

To describe your Spot fleet requests

This example describes all of your Spot fleet requests.

Command:

```
aws ec2 describe-spot-fleet-requests
```

Output:

```

{
  "SpotFleetRequestConfigs": [
    {
      "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
      "SpotFleetRequestConfig": {
        "TargetCapacity": 20,
        "LaunchSpecifications": [

```

```
    {
      "EbsOptimized": false,
      "NetworkInterfaces": [
        {
          "SubnetId": "subnet-a61dafcf",
          "DeviceIndex": 0,
          "DeleteOnTermination": false,
          "AssociatePublicIpAddress": true,
          "SecondaryPrivateIpAddressCount": 0
        }
      ],
      "InstanceType": "cc2.8xlarge",
      "ImageId": "ami-1a2b3c4d"
    },
    {
      "EbsOptimized": false,
      "NetworkInterfaces": [
        {
          "SubnetId": "subnet-a61dafcf",
          "DeviceIndex": 0,
          "DeleteOnTermination": false,
          "AssociatePublicIpAddress": true,
          "SecondaryPrivateIpAddressCount": 0
        }
      ],
      "InstanceType": "r3.8xlarge",
      "ImageId": "ami-1a2b3c4d"
    }
  ],
  "SpotPrice": "0.05",
  "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role"
},
"SpotFleetRequestState": "active"
},
{
  "SpotFleetRequestId": "sfr-306341ed-9739-402e-881b-ce47bEXAMPLE",
  "SpotFleetRequestConfig": {
    "TargetCapacity": 20,
    "LaunchSpecifications": [
      {
        "EbsOptimized": false,
        "NetworkInterfaces": [
          {
            "SubnetId": "subnet-6e7f829e",
```

```

        "DeviceIndex": 0,
        "DeleteOnTermination": false,
        "AssociatePublicIpAddress": true,
        "SecondaryPrivateIpAddressCount": 0
      }
    ],
    "InstanceType": "m3.medium",
    "ImageId": "ami-1a2b3c4d"
  }
],
"SpotPrice": "0.05",
"IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role"
},
"SpotFleetRequestState": "active"
}
]
}

```

To describe a Spot fleet request

This example describes the specified Spot fleet request.

Command:

```
aws ec2 describe-spot-fleet-requests --spot-fleet-request-ids sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

Output:

```

{
  "SpotFleetRequestConfigs": [
    {
      "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
      "SpotFleetRequestConfig": {
        "TargetCapacity": 20,
        "LaunchSpecifications": [
          {
            "EbsOptimized": false,
            "NetworkInterfaces": [
              {
                "SubnetId": "subnet-a61dafcf",
                "DeviceIndex": 0,

```



```

        "DeleteOnTermination": false,
        "AssociatePublicIpAddress": true,
        "SecondaryPrivateIpAddressCount": 0
      }
    ],
    "InstanceType": "cc2.8xlarge",
    "ImageId": "ami-1a2b3c4d"
  },
  {
    "EbsOptimized": false,
    "NetworkInterfaces": [
      {
        "SubnetId": "subnet-a61dafcf",
        "DeviceIndex": 0,
        "DeleteOnTermination": false,
        "AssociatePublicIpAddress": true,
        "SecondaryPrivateIpAddressCount": 0
      }
    ],
    "InstanceType": "r3.8xlarge",
    "ImageId": "ami-1a2b3c4d"
  }
],
"SpotPrice": "0.05",
"IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role"
},
"SpotFleetRequestState": "active"
}
]
}

```

- For API details, see [DescribeSpotFleetRequests](#) in *AWS CLI Command Reference*.

describe-spot-instance-requests

The following code example shows how to use `describe-spot-instance-requests`.

AWS CLI

Example 1: To describe a Spot Instance request

The following `describe-spot-instance-requests` example describes the specified Spot Instance request.

```
aws ec2 describe-spot-instance-requests \  
  --spot-instance-request-ids sir-08b93456
```

Output:

```
{  
  "SpotInstanceRequests": [  
    {  
      "CreateTime": "2018-04-30T18:14:55.000Z",  
      "InstanceId": "i-1234567890abcdef1",  
      "LaunchSpecification": {  
        "InstanceType": "t2.micro",  
        "ImageId": "ami-003634241a8fcdec0",  
        "KeyName": "my-key-pair",  
        "SecurityGroups": [  
          {  
            "GroupName": "default",  
            "GroupId": "sg-e38f24a7"  
          }  
        ],  
        "BlockDeviceMappings": [  
          {  
            "DeviceName": "/dev/sda1",  
            "Ebs": {  
              "DeleteOnTermination": true,  
              "SnapshotId": "snap-0e54a519c999adbbd",  
              "VolumeSize": 8,  
              "VolumeType": "standard",  
              "Encrypted": false  
            }  
          }  
        ],  
        "NetworkInterfaces": [  
          {  
            "DeleteOnTermination": true,  
            "DeviceIndex": 0,  
            "SubnetId": "subnet-049df61146c4d7901"  
          }  
        ],  
        "Placement": {  
          "AvailabilityZone": "us-east-2b",  
          "Tenancy": "default"  
        }  
      }  
    ]  
  }
```

```

        "Monitoring": {
            "Enabled": false
        }
    },
    "LaunchedAvailabilityZone": "us-east-2b",
    "ProductDescription": "Linux/UNIX",
    "SpotInstanceRequestId": "sir-08b93456",
    "SpotPrice": "0.010000",
    "State": "active",
    "Status": {
        "Code": "fulfilled",
        "Message": "Your Spot request is fulfilled.",
        "UpdateTime": "2018-04-30T18:16:21.000Z"
    },
    "Tags": [],
    "Type": "one-time",
    "InstanceInterruptionBehavior": "terminate"
    }
]
}

```

Example 2: To describe Spot Instance requests based on filters

The following `describe-spot-instance-requests` example uses filters to scope the results to Spot Instance requests with the specified instance type in the specified Availability Zone. The example uses the `--query` parameter to display only the instance IDs.

```

aws ec2 describe-spot-instance-requests \
  --filters Name=launch.instance-type,Values=m3.medium Name=launched-availability-
zone,Values=us-east-2a \
  --query "SpotInstanceRequests[*].[InstanceId]" \
  --output text

```

Output:

```

i-057750d42936e468a
i-001efd250faaa6ffa
i-027552a73f021f3bd
...

```

For additional examples using filters, see [Listing and filtering your resources](#) in the *Amazon Elastic Compute Cloud User Guide*.

Example 3: To describe Spot Instance requests based on tags

The following `describe-spot-instance-requests` example uses tag filters to scope the results to Spot Instance requests that have the tag `cost-center=cc123`.

```
aws ec2 describe-spot-instance-requests \  
  --filters Name=tag:cost-center,Values=cc123
```

For an example of the output for `describe-spot-instance-requests`, see Example 1.

For additional examples using tag filters, see [Working with tags](#) in the *Amazon EC2 User Guide*.

- For API details, see [DescribeSpotInstanceRequests](#) in *AWS CLI Command Reference*.

`describe-spot-price-history`

The following code example shows how to use `describe-spot-price-history`.

AWS CLI

To describe Spot price history

This example command returns the Spot Price history for `m1.xlarge` instances for a particular day in January.

Command:

```
aws ec2 describe-spot-price-history --instance-types m1.xlarge --start-time  
  2014-01-06T07:08:09 --end-time 2014-01-06T08:09:10
```

Output:

```
{  
  "SpotPriceHistory": [  
    {  
      "Timestamp": "2014-01-06T07:10:55.000Z",  
      "ProductDescription": "SUSE Linux",  
      "InstanceType": "m1.xlarge",  
      "SpotPrice": "0.087000",  
      "AvailabilityZone": "us-west-1b"  
    },  
    {
```

```

        "Timestamp": "2014-01-06T07:10:55.000Z",
        "ProductDescription": "SUSE Linux",
        "InstanceType": "m1.xlarge",
        "SpotPrice": "0.087000",
        "AvailabilityZone": "us-west-1c"
    },
    {
        "Timestamp": "2014-01-06T05:42:36.000Z",
        "ProductDescription": "SUSE Linux (Amazon VPC)",
        "InstanceType": "m1.xlarge",
        "SpotPrice": "0.087000",
        "AvailabilityZone": "us-west-1a"
    },
    ...
}

```

To describe Spot price history for Linux/UNIX Amazon VPC

This example command returns the Spot Price history for m1.xlarge, Linux/UNIX Amazon VPC instances for a particular day in January.

Command:

```
aws ec2 describe-spot-price-history --instance-types m1.xlarge --product-
description "Linux/UNIX (Amazon VPC)" --start-time 2014-01-06T07:08:09 --end-time
2014-01-06T08:09:10
```

Output:

```

{
  "SpotPriceHistory": [
    {
      "Timestamp": "2014-01-06T04:32:53.000Z",
      "ProductDescription": "Linux/UNIX (Amazon VPC)",
      "InstanceType": "m1.xlarge",
      "SpotPrice": "0.080000",
      "AvailabilityZone": "us-west-1a"
    },
    {
      "Timestamp": "2014-01-05T11:28:26.000Z",
      "ProductDescription": "Linux/UNIX (Amazon VPC)",
      "InstanceType": "m1.xlarge",
      "SpotPrice": "0.080000",
    }
  ]
}

```

```

        "AvailabilityZone": "us-west-1c"
    }
]
}

```

- For API details, see [DescribeSpotPriceHistory](#) in *AWS CLI Command Reference*.

describe-stale-security-groups

The following code example shows how to use `describe-stale-security-groups`.

AWS CLI

To describe stale security groups

This example describes stale security group rules for `vpc-11223344`. The response shows that `sg-5fa68d3a` in your account has a stale ingress SSH rule that references `sg-279ab042` in the peer VPC, and that `sg-fe6fba9a` in your account has a stale egress SSH rule that references `sg-ef6fba8b` in the peer VPC.

Command:

```
aws ec2 describe-stale-security-groups --vpc-id vpc-11223344
```

Output:

```

{
  "StaleSecurityGroupSet": [
    {
      "VpcId": "vpc-11223344",
      "StaleIpPermissionsEgress": [
        {
          "ToPort": 22,
          "FromPort": 22,
          "UserIdGroupPairs": [
            {
              "VpcId": "vpc-7a20e51f",
              "GroupId": "sg-ef6fba8b",
              "VpcPeeringConnectionId": "pcx-b04deed9",
              "PeeringStatus": "active"
            }
          ]
        }
      ]
    }
  ],
}

```

```

        "IpProtocol": "tcp"
      }
    ],
    "GroupName": "MySG1",
    "StaleIpPermissions": [],
    "GroupId": "sg-fe6fba9a",
    "Description": "MySG1"
  },
  {
    "VpcId": "vpc-11223344",
    "StaleIpPermissionsEgress": [],
    "GroupName": "MySG2",
    "StaleIpPermissions": [
      {
        "ToPort": 22,
        "FromPort": 22,
        "UserIdGroupPairs": [
          {
            "VpcId": "vpc-7a20e51f",
            "GroupId": "sg-279ab042",
            "Description": "Access from pcx-b04deed9",
            "VpcPeeringConnectionId": "pcx-b04deed9",
            "PeeringStatus": "active"
          }
        ]
      },
      {
        "IpProtocol": "tcp"
      }
    ],
    "GroupId": "sg-5fa68d3a",
    "Description": "MySG2"
  }
]
}

```

- For API details, see [DescribeStaleSecurityGroups](#) in *AWS CLI Command Reference*.

describe-store-image-tasks

The following code example shows how to use describe-store-image-tasks.

AWS CLI

To describe the progress of an AMI store task

The following `describe-store-image-tasks` example describes the progress of an AMI store task.

```
aws ec2 describe-store-image-tasks
```

Output:

```
{
  "AmiId": "ami-1234567890abcdef0",
  "Bucket": "my-ami-bucket",
  "ProgressPercentage": 17,
  "S3ObjectKey": "ami-1234567890abcdef0.bin",
  "StoreTaskState": "InProgress",
  "StoreTaskFailureReason": null,
  "TaskStartTime": "2022-01-01T01:01:01.001Z"
}
```

For more information about storing and restoring an AMI using S3, see [Store and restore an AMI using S3](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ami-store-restore.html) <<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ami-store-restore.html>> in the *Amazon EC2 User Guide*.

- For API details, see [DescribeStoreImageTasks](#) in *AWS CLI Command Reference*.

describe-subnets

The following code example shows how to use `describe-subnets`.

AWS CLI

Example 1: To describe all your subnets

The following `describe-subnets` example displays the details of your subnets.

```
aws ec2 describe-subnets
```

Output:

```
{
  "Subnets": [
    {
      "AvailabilityZone": "us-east-1d",
      "AvailabilityZoneId": "use1-az2",

```



```
    "AvailableIpAddressCount": 4089,
    "CidrBlock": "172.31.80.0/20",
    "DefaultForAz": true,
    "MapPublicIpOnLaunch": false,
    "MapCustomerOwnedIpOnLaunch": true,
    "State": "available",
    "SubnetId": "subnet-0bb1c79de3EXAMPLE",
    "VpcId": "vpc-0ee975135dEXAMPLE",
    "OwnerId": "111122223333",
    "AssignIpv6AddressOnCreation": false,
    "Ipv6CidrBlockAssociationSet": [],
    "CustomerOwnedIpv4Pool": "pool-2EXAMPLE",
    "SubnetArn": "arn:aws:ec2:us-east-2:111122223333:subnet/
subnet-0bb1c79de3EXAMPLE",
    "EnableDns64": false,
    "Ipv6Native": false,
    "PrivateDnsNameOptionsOnLaunch": {
      "HostnameType": "ip-name",
      "EnableResourceNameDnsARecord": false,
      "EnableResourceNameDnsAAAARecord": false
    }
  },
  {
    "AvailabilityZone": "us-east-1d",
    "AvailabilityZoneId": "use1-az2",
    "AvailableIpAddressCount": 4089,
    "CidrBlock": "172.31.80.0/20",
    "DefaultForAz": true,
    "MapPublicIpOnLaunch": true,
    "MapCustomerOwnedIpOnLaunch": false,
    "State": "available",
    "SubnetId": "subnet-8EXAMPLE",
    "VpcId": "vpc-3EXAMPLE",
    "OwnerId": "111122223333",
    "AssignIpv6AddressOnCreation": false,
    "Ipv6CidrBlockAssociationSet": [],
    "Tags": [
      {
        "Key": "Name",
        "Value": "MySubnet"
      }
    ],
    "SubnetArn": "arn:aws:ec2:us-east-1:111122223333:subnet/
subnet-8EXAMPLE",
```

```

        "EnableDns64": false,
        "Ipv6Native": false,
        "PrivateDnsNameOptionsOnLaunch": {
            "HostnameType": "ip-name",
            "EnableResourceNameDnsARecord": false,
            "EnableResourceNameDnsAAAARecord": false
        }
    }
]
}

```

For more information, see [Working with VPCs and Subnets](#) in the *AWS VPC User Guide*.

Example 2: To describe the subnets of a specific VPC

The following describe-subnets example uses a filter to retrieve details for the subnets of the specified VPC.

```

aws ec2 describe-subnets \
  --filters "Name=vpc-id,Values=vpc-3EXAMPLE"

```

Output:

```

{
  "Subnets": [
    {
      "AvailabilityZone": "us-east-1d",
      "AvailabilityZoneId": "use1-az2",
      "AvailableIpAddressCount": 4089,
      "CidrBlock": "172.31.80.0/20",
      "DefaultForAz": true,
      "MapPublicIpOnLaunch": true,
      "MapCustomerOwnedIpOnLaunch": false,
      "State": "available",
      "SubnetId": "subnet-8EXAMPLE",
      "VpcId": "vpc-3EXAMPLE",
      "OwnerId": "1111222233333",
      "AssignIpv6AddressOnCreation": false,
      "Ipv6CidrBlockAssociationSet": [],
      "Tags": [
        {
          "Key": "Name",
          "Value": "MySubnet"
        }
      ]
    }
  ]
}

```

```

    }
  ],
  "SubnetArn": "arn:aws:ec2:us-east-1:111122223333:subnet/
subnet-8EXAMPLE",
  "EnableDns64": false,
  "Ipv6Native": false,
  "PrivateDnsNameOptionsOnLaunch": {
    "HostnameType": "ip-name",
    "EnableResourceNameDnsARecord": false,
    "EnableResourceNameDnsAAAARecord": false
  }
}
]
}

```

For more information, see [Working with VPCs and Subnets](#) in the *AWS VPC User Guide*.

Example 3: To describe the subnets with a specific tag

The following `describe-subnets` example uses a filter to retrieve the details of those subnets with the tag `CostCenter=123` and the `--query` parameter to display the subnet IDs of the subnets with this tag.

```

aws ec2 describe-subnets \
  --filters "Name=tag:CostCenter,Values=123" \
  --query "Subnets[*].SubnetId" \
  --output text

```

Output:

```

subnet-0987a87c8b37348ef
subnet-02a95061c45f372ee
subnet-03f720e7de2788d73

```

For more information, see [Working with VPCs and Subnets](#) in the *Amazon VPC User Guide*.

- For API details, see [DescribeSubnets](#) in *AWS CLI Command Reference*.

describe-tags

The following code example shows how to use `describe-tags`.

AWS CLI

Example 1: To describe all tags for a single resource

The following describe-tags example describes the tags for the specified instance.

```
aws ec2 describe-tags \  
  --filters "Name=resource-id,Values=i-1234567890abcdef8"
```

Output:

```
{  
  "Tags": [  
    {  
      "ResourceType": "instance",  
      "ResourceId": "i-1234567890abcdef8",  
      "Value": "Test",  
      "Key": "Stack"  
    },  
    {  
      "ResourceType": "instance",  
      "ResourceId": "i-1234567890abcdef8",  
      "Value": "Beta Server",  
      "Key": "Name"  
    }  
  ]  
}
```

Example 2: To describe all tags for a resource type

The following describe-tags example describes the tags for your volumes.

```
aws ec2 describe-tags \  
  --filters "Name=resource-type,Values=volume"
```

Output:

```
{  
  "Tags": [  
    {  
      "ResourceType": "volume",  
      "ResourceId": "vol-1234567890abcdef0",  
      "Value": "Project1",  
    }  
  ]  
}
```

```

        "Key": "Purpose"
    },
    {
        "ResourceType": "volume",
        "ResourceId": "vol-049df61146c4d7901",
        "Value": "Logs",
        "Key": "Purpose"
    }
]
}

```

Example 3: To describe all your tags

The following describe-tags example describes the tags for all your resources.

```
aws ec2 describe-tags
```

Example 4: To describe the tags for your resources based on a tag key

The following describe-tags example describes the tags for your resources that have a tag with the key Stack.

```
aws ec2 describe-tags \
  --filters Name=key,Values=Stack
```

Output:

```

{
  "Tags": [
    {
      "ResourceType": "volume",
      "ResourceId": "vol-027552a73f021f3b",
      "Value": "Production",
      "Key": "Stack"
    },
    {
      "ResourceType": "instance",
      "ResourceId": "i-1234567890abcdef8",
      "Value": "Test",
      "Key": "Stack"
    }
  ]
}

```

```
}
```

Example 5: To describe the tags for your resources based on a tag key and tag value

The following `describe-tags` example describes the tags for your resources that have the tag `Stack=Test`.

```
aws ec2 describe-tags \  
  --filters Name=key,Values=Stack Name=value,Values=Test
```

Output:

```
{  
  "Tags": [  
    {  
      "ResourceType": "image",  
      "ResourceId": "ami-3ac336533f021f3bd",  
      "Value": "Test",  
      "Key": "Stack"  
    },  
    {  
      "ResourceType": "instance",  
      "ResourceId": "i-1234567890abcdef8",  
      "Value": "Test",  
      "Key": "Stack"  
    }  
  ]  
}
```

The following `describe-tags` example uses alternate syntax to describe resources with the tag `Stack=Test`.

```
aws ec2 describe-tags \  
  --filters "Name=tag:Stack,Values=Test"
```

The following `describe-tags` example describes the tags for all your instances that have a tag with the key `Purpose` and no value.

```
aws ec2 describe-tags \  
  --filters "Name=resource-type,Values=instance" "Name=key,Values=Purpose"  
  "Name=value,Values="
```

Output:

```
{
  "Tags": [
    {
      "ResourceType": "instance",
      "ResourceId": "i-1234567890abcdef5",
      "Value": null,
      "Key": "Purpose"
    }
  ]
}
```

- For API details, see [DescribeTags](#) in *AWS CLI Command Reference*.

describe-traffic-mirror-filters

The following code example shows how to use `describe-traffic-mirror-filters`.

AWS CLI**To view your traffic mirror filters**

The following `describe-traffic-mirror-filters` example displays details for all of your traffic mirror filters.

```
aws ec2 describe-traffic-mirror-filters
```

Output:

```
{
  "TrafficMirrorFilters": [
    {
      "TrafficMirrorFilterId": "tmf-0293f26e86EXAMPLE",
      "IngressFilterRules": [
        {
          "TrafficMirrorFilterRuleId": "tmfr-0ca76e0e08EXAMPLE",
          "TrafficMirrorFilterId": "tmf-0293f26e86EXAMPLE",
          "TrafficDirection": "ingress",
          "RuleNumber": 100,
          "RuleAction": "accept",
          "Protocol": 6,
```

```

        "DestinationCidrBlock": "10.0.0.0/24",
        "SourceCidrBlock": "10.0.0.0/24",
        "Description": "TCP Rule"
    }
],
"EgressFilterRules": [],
"NetworkServices": [],
>Description": "Example filter",
"Tags": []
}
]
}

```

For more information, see [View your traffic mirror filters](#) in the *Traffic Mirroring Guide*.

- For API details, see [DescribeTrafficMirrorFilters](#) in *AWS CLI Command Reference*.

describe-traffic-mirror-sessions

The following code example shows how to use describe-traffic-mirror-sessions.

AWS CLI

To describe a Traffic Mirror Session

The following describe-traffic-mirror-sessions example displays details of the your Traffic Mirror sessions.

```
aws ec2 describe-traffic-mirror-sessions
```

Output:

```

{
  "TrafficMirrorSessions": [
    {
      "Tags": [],
      "VirtualNetworkId": 42,
      "OwnerId": "111122223333",
      "Description": "TCP Session",
      "NetworkInterfaceId": "eni-0a471a5cf3EXAMPLE",
      "TrafficMirrorTargetId": "tmt-0dabe9b0a6EXAMPLE",
      "TrafficMirrorFilterId": "tmf-083e18f985EXAMPLE",
      "PacketLength": 20,
    }
  ]
}

```



```

    "SessionNumber": 1,
    "TrafficMirrorSessionId": "tms-0567a4c684EXAMPLE"
  },
  {
    "Tags": [
      {
        "Key": "Name",
        "Value": "tag test"
      }
    ],
    "VirtualNetworkId": 13314501,
    "OwnerId": "111122223333",
    "Description": "TCP Session",
    "NetworkInterfaceId": "eni-0a471a5cf3EXAMPLE",
    "TrafficMirrorTargetId": "tmt-03665551cbEXAMPLE",
    "TrafficMirrorFilterId": "tmf-06c787846cEXAMPLE",
    "SessionNumber": 2,
    "TrafficMirrorSessionId": "tms-0060101cf8EXAMPLE"
  }
]
}

```

For more information, see [View Traffic Mirror Session Details](#) in the *AWS Traffic Mirroring Guide*.

- For API details, see [DescribeTrafficMirrorSessions](#) in *AWS CLI Command Reference*.

describe-traffic-mirror-targets

The following code example shows how to use describe-traffic-mirror-targets.

AWS CLI

To describe a traffic mirror target

The following describe-traffic-mirror-targets example displays information about the specified traffic mirror target.

```
aws ec2 describe-traffic-mirror-targets \
  --traffic-mirror-target-ids tmt-0dabe9b0a6EXAMPLE
```

Output:

```
{
```

```
"TrafficMirrorTargets": [  
  {  
    "TrafficMirrorTargetId": "tmt-0dabe9b0a6EXAMPLE",  
    "NetworkLoadBalancerArn": "arn:aws:elasticloadbalancing:us-  
east-1:111122223333:loadbalancer/net/NLB/7cdec873fEXAMPLE",  
    "Type": "network-load-balancer",  
    "Description": "Example Network Load Balancer target",  
    "OwnerId": "111122223333",  
    "Tags": []  
  }  
]
```

For more information, see [Traffic mirror targets](#) in the *Amazon VPC Traffic Mirroring Guide*.

- For API details, see [DescribeTrafficMirrorTargets](#) in *AWS CLI Command Reference*.

describe-transit-gateway-attachments

The following code example shows how to use `describe-transit-gateway-attachments`.

AWS CLI

To view your transit gateway attachments

The following `describe-transit-gateway-attachments` example displays details for your transit gateway attachments.

```
aws ec2 describe-transit-gateway-attachments
```

Output:

```
{  
  "TransitGatewayAttachments": [  
    {  
      "TransitGatewayAttachmentId": "tgw-attach-01f8100bc7EXAMPLE",  
      "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",  
      "TransitGatewayOwnerId": "123456789012",  
      "ResourceOwnerId": "123456789012",  
      "ResourceType": "vpc",  
      "ResourceId": "vpc-3EXAMPLE",  
      "State": "available",  
      "Association": {
```

```
        "TransitGatewayRouteTableId": "tgw-rtb-002573ed1eEXAMPLE",
        "State": "associated"
    },
    "CreationTime": "2019-08-26T14:59:25.000Z",
    "Tags": [
        {
            "Key": "Name",
            "Value": "Example"
        }
    ]
},
{
    "TransitGatewayAttachmentId": "tgw-attach-0b5968d3b6EXAMPLE",
    "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",
    "TransitGatewayOwnerId": "123456789012",
    "ResourceOwnerId": "123456789012",
    "ResourceType": "vpc",
    "ResourceId": "vpc-0065acced4EXAMPLE",
    "State": "available",
    "Association": {
        "TransitGatewayRouteTableId": "tgw-rtb-002573ed1eEXAMPLE",
        "State": "associated"
    },
    "CreationTime": "2019-08-07T17:03:07.000Z",
    "Tags": []
},
{
    "TransitGatewayAttachmentId": "tgw-attach-08e0bc912cEXAMPLE",
    "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",
    "TransitGatewayOwnerId": "123456789012",
    "ResourceOwnerId": "123456789012",
    "ResourceType": "direct-connect-gateway",
    "ResourceId": "11460968-4ac1-4fd3-bdb2-00599EXAMPLE",
    "State": "available",
    "Association": {
        "TransitGatewayRouteTableId": "tgw-rtb-002573ed1eEXAMPLE",
        "State": "associated"
    },
    "CreationTime": "2019-08-14T20:27:44.000Z",
    "Tags": []
},
{
    "TransitGatewayAttachmentId": "tgw-attach-0a89069f57EXAMPLE",
    "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",
```

```

    "TransitGatewayOwnerId": "123456789012",
    "ResourceOwnerId": "123456789012",
    "ResourceType": "direct-connect-gateway",
    "ResourceId": "8384da05-13ce-4a91-aada-5a1baEXAMPLE",
    "State": "available",
    "Association": {
      "TransitGatewayRouteTableId": "tgw-rtb-002573ed1eEXAMPLE",
      "State": "associated"
    },
    "CreationTime": "2019-08-14T20:33:02.000Z",
    "Tags": []
  }
]
}

```

For more information, see [Work with transit gateways](#) in the *Transit Gateways Guide*.

- For API details, see [DescribeTransitGatewayAttachments](#) in *AWS CLI Command Reference*.

describe-transit-gateway-connect-peers

The following code example shows how to use `describe-transit-gateway-connect-peers`.

AWS CLI

To describe a Transit Gateway Connect peer

The following `describe-transit-gateway-connect-peers` example describes the specified Connect peer.

```

aws ec2 describe-transit-gateway-connect-peers \
  --transit-gateway-connect-peer-ids tgw-connect-peer-0666adbac4EXAMPLE

```

Output:

```

{
  "TransitGatewayConnectPeers": [
    {
      "TransitGatewayAttachmentId": "tgw-attach-0f0927767cEXAMPLE",
      "TransitGatewayConnectPeerId": "tgw-connect-peer-0666adbac4EXAMPLE",
      "State": "available",
      "CreationTime": "2021-10-13T03:35:17.000Z",
      "ConnectPeerConfiguration": {

```

```

    "TransitGatewayAddress": "10.0.0.234",
    "PeerAddress": "172.31.1.11",
    "InsideCidrBlocks": [
        "169.254.6.0/29"
    ],
    "Protocol": "gre",
    "BgpConfigurations": [
        {
            "TransitGatewayAsn": 64512,
            "PeerAsn": 64512,
            "TransitGatewayAddress": "169.254.6.2",
            "PeerAddress": "169.254.6.1",
            "BgpStatus": "down"
        },
        {
            "TransitGatewayAsn": 64512,
            "PeerAsn": 64512,
            "TransitGatewayAddress": "169.254.6.3",
            "PeerAddress": "169.254.6.1",
            "BgpStatus": "down"
        }
    ],
    "Tags": []
}
]
}

```

For more information, see [Transit gateway Connect attachments and Transit Gateway Connect peers](#) in the *Transit Gateways Guide*.

- For API details, see [DescribeTransitGatewayConnectPeers](#) in *AWS CLI Command Reference*.

describe-transit-gateway-connects

The following code example shows how to use describe-transit-gateway-connects.

AWS CLI

To describe a transit gateway Connect attachment

The following describe-transit-gateway-connects example describes the specified Connect attachment.

```
aws ec2 describe-transit-gateway-connects \  
  --transit-gateway-attachment-ids tgw-attach-037012e5dcEXAMPLE
```

Output:

```
{  
  "TransitGatewayConnects": [  
    {  
      "TransitGatewayAttachmentId": "tgw-attach-037012e5dcEXAMPLE",  
      "TransportTransitGatewayAttachmentId": "tgw-attach-0a89069f57EXAMPLE",  
      "TransitGatewayId": "tgw-02f776b1a7EXAMPLE",  
      "State": "available",  
      "CreationTime": "2021-03-09T19:59:17+00:00",  
      "Options": {  
        "Protocol": "gre"  
      },  
      "Tags": []  
    }  
  ]  
}
```

For more information, see [Transit gateway Connect attachments and Transit Gateway Connect peers](#) in the *Transit Gateways Guide*.

- For API details, see [DescribeTransitGatewayConnects](#) in *AWS CLI Command Reference*.

describe-transit-gateway-multicast-domains

The following code example shows how to use `describe-transit-gateway-multicast-domains`.

AWS CLI**To describe your transit gateway multicast domains**

The following `describe-transit-gateway-multicast-domains` example displays details for all of your transit gateway multicast domains.

```
aws ec2 describe-transit-gateway-multicast-domains
```

Output:

```
{
  "TransitGatewayMulticastDomains": [
    {
      "TransitGatewayMulticastDomainId": "tgw-mcast-domain-000fb24d04EXAMPLE",
      "TransitGatewayId": "tgw-0bf0bfffefaEXAMPLE",
      "TransitGatewayMulticastDomainArn": "arn:aws:ec2:us-
east-1:123456789012:transit-gateway-multicast-domain/tgw-mcast-
domain-000fb24d04EXAMPLE",
      "OwnerId": "123456789012",
      "Options": {
        "Icmpv2Support": "disable",
        "StaticSourcesSupport": "enable",
        "AutoAcceptSharedAssociations": "disable"
      },
      "State": "available",
      "CreationTime": "2019-12-10T18:32:50+00:00",
      "Tags": [
        {
          "Key": "Name",
          "Value": "mc1"
        }
      ]
    }
  ]
}
```

For more information, see [Managing multicast domains](#) in the *Transit Gateways Guide*.

- For API details, see [DescribeTransitGatewayMulticastDomains](#) in *AWS CLI Command Reference*.

describe-transit-gateway-peering-attachments

The following code example shows how to use `describe-transit-gateway-peering-attachments`.

AWS CLI

To describe your transit gateway peering attachments

The following `describe-transit-gateway-peering-attachments` example displays details for all of your transit gateway peering attachments.

```
aws ec2 describe-transit-gateway-peering-attachments
```

Output:

```
{
  "TransitGatewayPeeringAttachments": [
    {
      "TransitGatewayAttachmentId": "tgw-attach-4455667788aabbccd",
      "RequesterTgwInfo": {
        "TransitGatewayId": "tgw-123abc05e04123abc",
        "OwnerId": "123456789012",
        "Region": "us-west-2"
      },
      "AcceptorTgwInfo": {
        "TransitGatewayId": "tgw-11223344aabbcc112",
        "OwnerId": "123456789012",
        "Region": "us-east-2"
      },
      "State": "pendingAcceptance",
      "CreationTime": "2019-12-09T11:38:05.000Z",
      "Tags": []
    }
  ]
}
```

For more information, see [Transit Gateway Peering Attachments](#) in the *Transit Gateways Guide*.

- For API details, see [DescribeTransitGatewayPeeringAttachments](#) in *AWS CLI Command Reference*.

describe-transit-gateway-policy-tables

The following code example shows how to use `describe-transit-gateway-policy-tables`.

AWS CLI

To describe a transit gateway policy table

The following `describe-transit-gateway-policy-tables` example describes the specified transit gateway policy table.

```
aws ec2 describe-transit-gateway-policy-tables \
```



```
--transit-gateway-policy-table-ids tgw-ptb-0a16f134b78668a81
```

Output:

```
{
  "TransitGatewayPolicyTables": [
    {
      "TransitGatewayPolicyTableId": "tgw-ptb-0a16f134b78668a81",
      "TransitGatewayId": "tgw-067f8505c18f0bd6e",
      "State": "available",
      "CreationTime": "2023-11-28T16:36:43+00:00",
      "Tags": []
    }
  ]
}
```

For more information, see [Transit gateway policy tables](#) in the *Transit Gateway User Guide*.

- For API details, see [DescribeTransitGatewayPolicyTables](#) in *AWS CLI Command Reference*.

describe-transit-gateway-route-tables

The following code example shows how to use `describe-transit-gateway-route-tables`.

AWS CLI

To describe your transit gateway route tables

The following `describe-transit-gateway-route-tables` example displays details for your transit gateway route tables.

```
aws ec2 describe-transit-gateway-route-tables
```

Output:

```
{
  "TransitGatewayRouteTables": [
    {
      "TransitGatewayRouteTableId": "tgw-rtb-0ca78a549EXAMPLE",
      "TransitGatewayId": "tgw-0bc994abffEXAMPLE",
      "State": "available",
      "DefaultAssociationRouteTable": true,
    }
  ]
}
```

```

        "DefaultPropagationRouteTable": true,
        "CreationTime": "2018-11-28T14:24:49.000Z",
        "Tags": []
    },
    {
        "TransitGatewayRouteTableId": "tgw-rtb-0e8f48f148EXAMPLE",
        "TransitGatewayId": "tgw-0043d72bb4EXAMPLE",
        "State": "available",
        "DefaultAssociationRouteTable": true,
        "DefaultPropagationRouteTable": true,
        "CreationTime": "2018-11-28T14:24:00.000Z",
        "Tags": []
    }
]
}

```

For more information, see [View transit gateway route tables](#) in the *Transit Gateways Guide*.

- For API details, see [DescribeTransitGatewayRouteTables](#) in *AWS CLI Command Reference*.

describe-transit-gateway-vpc-attachments

The following code example shows how to use `describe-transit-gateway-vpc-attachments`.

AWS CLI

To describe your transit gateway VPC attachments

The following `describe-transit-gateway-vpc-attachments` example displays details for your transit gateway VPC attachments.

```
aws ec2 describe-transit-gateway-vpc-attachments
```

Output:

```

{
  "TransitGatewayVpcAttachments": [
    {
      "TransitGatewayAttachmentId": "tgw-attach-0a08e88308EXAMPLE",
      "TransitGatewayId": "tgw-0043d72bb4EXAMPLE",
      "VpcId": "vpc-0f501f7ee8EXAMPLE",
      "VpcOwnerId": "111122223333",
    }
  ]
}

```

```

    "State": "available",
    "SubnetIds": [
      "subnet-045d586432EXAMPLE",
      "subnet-0a0ad478a6EXAMPLE"
    ],
    "CreationTime": "2019-02-13T11:04:02.000Z",
    "Options": {
      "DnsSupport": "enable",
      "Ipv6Support": "disable"
    },
    "Tags": [
      {
        "Key": "Name",
        "Value": "attachment name"
      }
    ]
  }
]
}

```

For more information, see [View your VPC attachments](#) in the *Transit Gateways Guide*.

- For API details, see [DescribeTransitGatewayVpcAttachments](#) in *AWS CLI Command Reference*.

describe-transit-gateways

The following code example shows how to use describe-transit-gateways.

AWS CLI

To describe your transit gateways

The following describe-transit-gateways example retrieves details about your transit gateways.

```
aws ec2 describe-transit-gateways
```

Output:

```

{
  "TransitGateways": [
    {
      "TransitGatewayId": "tgw-0262a0e521EXAMPLE",

```

```
    "TransitGatewayArn": "arn:aws:ec2:us-east-2:111122223333:transit-
gateway/tgw-0262a0e521EXAMPLE",
    "State": "available",
    "OwnerId": "111122223333",
    "Description": "MyTGW",
    "CreationTime": "2019-07-10T14:02:12.000Z",
    "Options": {
      "AmazonSideAsn": 64516,
      "AutoAcceptSharedAttachments": "enable",
      "DefaultRouteTableAssociation": "enable",
      "AssociationDefaultRouteTableId": "tgw-rtb-018774adf3EXAMPLE",
      "DefaultRouteTablePropagation": "enable",
      "PropagationDefaultRouteTableId": "tgw-rtb-018774adf3EXAMPLE",
      "VpnEcmpSupport": "enable",
      "DnsSupport": "enable"
    },
    "Tags": []
  },
  {
    "TransitGatewayId": "tgw-0fb8421e2dEXAMPLE",
    "TransitGatewayArn": "arn:aws:ec2:us-east-2:111122223333:transit-
gateway/tgw-0fb8421e2da853bf3",
    "State": "available",
    "OwnerId": "111122223333",
    "CreationTime": "2019-03-15T22:57:33.000Z",
    "Options": {
      "AmazonSideAsn": 65412,
      "AutoAcceptSharedAttachments": "disable",
      "DefaultRouteTableAssociation": "enable",
      "AssociationDefaultRouteTableId": "tgw-rtb-06a241a3d8EXAMPLE",
      "DefaultRouteTablePropagation": "enable",
      "PropagationDefaultRouteTableId": "tgw-rtb-06a241a3d8EXAMPLE",
      "VpnEcmpSupport": "enable",
      "DnsSupport": "enable"
    },
    "Tags": [
      {
        "Key": "Name",
        "Value": "TGW1"
      }
    ]
  }
]
```

```
}
```

- For API details, see [DescribeTransitGateways](#) in *AWS CLI Command Reference*.

describe-verified-access-endpoints

The following code example shows how to use `describe-verified-access-endpoints`.

AWS CLI

To describe a Verified Access endpoint

The following `delete-verified-access-endpoints` example describes the specified Verified Access endpoint.

```
aws ec2 describe-verified-access-endpoints \  
  --verified-access-endpoint-ids vae-066fac616d4d546f2
```

Output:

```
{  
  "VerifiedAccessEndpoints": [  
    {  
      "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",  
      "VerifiedAccessGroupId": "vagr-0dbe967baf14b7235",  
      "VerifiedAccessEndpointId": "vae-066fac616d4d546f2",  
      "ApplicationDomain": "example.com",  
      "EndpointType": "network-interface",  
      "AttachmentType": "vpc",  
      "DomainCertificateArn": "arn:aws:acm:us-east-2:123456789012:certificate/  
eb065ea0-26f9-4e75-a6ce-0a1a7EXAMPLE",  
      "EndpointDomain": "my-ava-  
app.edge-00c3372d53b1540bb.vai-0ce000c0b7643abea.prod.verified-access.us-  
east-2.amazonaws.com",  
      "SecurityGroupIds": [  
        "sg-004915970c4c8f13a"  
      ],  
      "NetworkInterfaceOptions": {  
        "NetworkInterfaceId": "eni-0aec70418c8d87a0f",  
        "Protocol": "https",  
        "Port": 443  
      },  
    },  
  ],  
}
```

```
    "Status": {
      "Code": "active"
    },
    "Description": "",
    "CreationTime": "2023-08-25T20:54:43",
    "LastUpdatedTime": "2023-08-25T22:17:26",
    "Tags": [
      {
        "Key": "Name",
        "Value": "my-va-endpoint"
      }
    ]
  }
]
```

For more information, see [Verified Access endpoints](#) in the *AWS Verified Access User Guide*.

- For API details, see [DescribeVerifiedAccessEndpoints](#) in *AWS CLI Command Reference*.

describe-verified-access-groups

The following code example shows how to use `describe-verified-access-groups`.

AWS CLI

To describe a Verified Access group

The following `describe-verified-access-groups` example describes the specified Verified Access group.

```
aws ec2 describe-verified-access-groups \
  --verified-access-group-ids vagr-0dbe967baf14b7235
```

Output:

```
{
  "VerifiedAccessGroups": [
    {
      "VerifiedAccessGroupId": "vagr-0dbe967baf14b7235",
      "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
      "Description": "Testing Verified Access",
```

```

    "Owner": "123456789012",
    "VerifiedAccessGroupArn": "arn:aws:ec2:us-east-2:123456789012:verified-
access-group/vagr-0dbe967baf14b7235",
    "CreationTime": "2023-08-25T19:55:19",
    "LastUpdatedTime": "2023-08-25T22:17:25",
    "Tags": [
      {
        "Key": "Name",
        "Value": "my-va-group"
      }
    ]
  }
]
}

```

For more information, see [Verified Access groups](#) in the *AWS Verified Access User Guide*.

- For API details, see [DescribeVerifiedAccessGroups](#) in *AWS CLI Command Reference*.

describe-verified-access-instance-logging-configurations

The following code example shows how to use `describe-verified-access-instance-logging-configurations`.

AWS CLI

To describe the logging configuration for a Verified Access instance

The following `describe-verified-access-instance-logging-configurations` example describes the logging configuration for the specified Verified Access instance.

```
aws ec2 describe-verified-access-instance-logging-configurations \
  --verified-access-instance-ids vai-0ce000c0b7643abea
```

Output:

```

{
  "LoggingConfigurations": [
    {
      "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
      "AccessLogs": {
        "S3": {

```

```

        "Enabled": false
    },
    "CloudWatchLogs": {
        "Enabled": true,
        "DeliveryStatus": {
            "Code": "success"
        },
        "LogGroup": "my-log-group"
    },
    "KinesisDataFirehose": {
        "Enabled": false
    },
    "LogVersion": "ocsf-1.0.0-rc.2",
    "IncludeTrustContext": false
}
]
}

```

For more information, see [Verified Access logs](#) in the *AWS Verified Access User Guide*.

- For API details, see [DescribeVerifiedAccessInstanceLoggingConfigurations](#) in *AWS CLI Command Reference*.

describe-verified-access-instances

The following code example shows how to use `describe-verified-access-instances`.

AWS CLI

To describe a Verified Access instance

The following `describe-verified-access-instances` example describes the specified Verified Access instance.

```
aws ec2 describe-verified-access-instances \
  --verified-access-instance-ids vai-0ce000c0b7643abea
```

Output:

```
{
  "VerifiedAccessInstances": [
```



```

    {
      "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
      "Description": "Testing Verified Access",
      "VerifiedAccessTrustProviders": [
        {
          "VerifiedAccessTrustProviderId": "vatp-0bb32de759a3e19e7",
          "TrustProviderType": "user",
          "UserTrustProviderType": "iam-identity-center"
        }
      ],
      "CreationTime": "2023-08-25T18:27:56",
      "LastUpdatedTime": "2023-08-25T19:03:32",
      "Tags": [
        {
          "Key": "Name",
          "Value": "my-ava-instance"
        }
      ]
    }
  ]
}

```

For more information, see [Verified Access instances](#) in the *AWS Verified Access User Guide*.

- For API details, see [DescribeVerifiedAccessInstances](#) in *AWS CLI Command Reference*.

describe-verified-access-trust-providers

The following code example shows how to use `describe-verified-access-trust-providers`.

AWS CLI

To describe a Verified Access trust provider

The following `describe-verified-access-trust-providers` example describes the specified Verified Access trust provider.

```

aws ec2 describe-verified-access-trust-providers \
  --verified-access-trust-provider-ids vatp-0bb32de759a3e19e7

```

Output:

```
{
  "VerifiedAccessTrustProviders": [
    {
      "VerifiedAccessTrustProviderId": "vatp-0bb32de759a3e19e7",
      "Description": "Testing Verified Access",
      "TrustProviderType": "user",
      "UserTrustProviderType": "iam-identity-center",
      "PolicyReferenceName": "idc",
      "CreationTime": "2023-08-25T19:00:38",
      "LastUpdatedTime": "2023-08-25T19:03:32",
      "Tags": [
        {
          "Key": "Name",
          "Value": "my-va-trust-provider"
        }
      ]
    }
  ]
}
```

For more information, see [Trust providers for Verified Access](#) in the *AWS Verified Access User Guide*.

- For API details, see [DescribeVerifiedAccessTrustProviders](#) in *AWS CLI Command Reference*.

describe-volume-attribute

The following code example shows how to use `describe-volume-attribute`.

AWS CLI

To describe a volume attribute

This example command describes the `autoEnableIo` attribute of the volume with the ID `vol-049df61146c4d7901`.

Command:

```
aws ec2 describe-volume-attribute --volume-id vol-049df61146c4d7901 --attribute
autoEnableIO
```

Output:

```
{
  "AutoEnableIO": {
    "Value": false
  },
  "VolumeId": "vol-049df61146c4d7901"
}
```

- For API details, see [DescribeVolumeAttribute](#) in *AWS CLI Command Reference*.

describe-volume-status

The following code example shows how to use `describe-volume-status`.

AWS CLI

To describe the status of a single volume

This example command describes the status for the volume `vol-1234567890abcdef0`.

Command:

```
aws ec2 describe-volume-status --volume-ids vol-1234567890abcdef0
```

Output:

```
{
  "VolumeStatuses": [
    {
      "VolumeStatus": {
        "Status": "ok",
        "Details": [
          {
            "Status": "passed",
            "Name": "io-enabled"
          },
          {
            "Status": "not-applicable",
            "Name": "io-performance"
          }
        ]
      },
      "AvailabilityZone": "us-east-1a",
    }
  ]
}
```

```
        "VolumeId": "vol-1234567890abcdef0",
        "Actions": [],
        "Events": []
    }
]
}
```

To describe the status of impaired volumes

This example command describes the status for all volumes that are impaired. In this example output, there are no impaired volumes.

Command:

```
aws ec2 describe-volume-status --filters Name=volume-status.status,Values=impaired
```

Output:

```
{
  "VolumeStatuses": []
}
```

If you have a volume with a failed status check (status is impaired), see [Working with an Impaired Volume](#) in the *Amazon EC2 User Guide*.

- For API details, see [DescribeVolumeStatus](#) in *AWS CLI Command Reference*.

describe-volumes-modifications

The following code example shows how to use `describe-volumes-modifications`.

AWS CLI

To describe the modification status for a volume

The following `describe-volumes-modifications` example describes the volume modification status of the specified volume.

```
aws ec2 describe-volumes-modifications \
  --volume-ids vol-1234567890abcdef0
```

Output:

```
{
  "VolumeModification": {
    "TargetSize": 150,
    "TargetVolumeType": "io1",
    "ModificationState": "optimizing",
    "VolumeId": "vol-1234567890abcdef0",
    "TargetIops": 100,
    "StartTime": "2019-05-17T11:27:19.000Z",
    "Progress": 70,
    "OriginalVolumeType": "io1",
    "OriginalIops": 100,
    "OriginalSize": 100
  }
}
```

- For API details, see [DescribeVolumesModifications](#) in *AWS CLI Command Reference*.

describe-volumes

The following code example shows how to use `describe-volumes`.

AWS CLI**Example 1: To describe a volume**

The following `describe-volumes` example describes the specified volumes in the current Region.

```
aws ec2 describe-volumes \
  --volume-ids vol-049df61146c4d7901 vol-1234567890abcdef0
```

Output:

```
{
  "Volumes": [
    {
      "AvailabilityZone": "us-east-1a",
      "Attachments": [
        {
```

```

        "AttachTime": "2013-12-18T22:35:00.000Z",
        "InstanceId": "i-1234567890abcdef0",
        "VolumeId": "vol-049df61146c4d7901",
        "State": "attached",
        "DeleteOnTermination": true,
        "Device": "/dev/sda1"
    }
],
"Encrypted": true,
"KmsKeyId": "arn:aws:kms:us-east-2a:123456789012:key/8c5b2c63-b9bc-45a3-
a87a-5513eEXAMPLE,
"VolumeType": "gp2",
"VolumeId": "vol-049df61146c4d7901",
"State": "in-use",
"Iops": 100,
"SnapshotId": "snap-1234567890abcdef0",
"CreateTime": "2019-12-18T22:35:00.084Z",
"Size": 8
},
{
    "AvailabilityZone": "us-east-1a",
    "Attachments": [],
    "Encrypted": false,
    "VolumeType": "gp2",
    "VolumeId": "vol-1234567890abcdef0",
    "State": "available",
    "Iops": 300,
    "SnapshotId": "",
    "CreateTime": "2020-02-27T00:02:41.791Z",
    "Size": 100
}
]
}

```

Example 2: To describe volumes that are attached to a specific instance

The following describe-volumes example describes all volumes that are both attached to the specified instance and set to delete when the instance terminates.

```

aws ec2 describe-volumes \
  --region us-east-1 \
  --filters Name=attachment.instance-id,Values=i-1234567890abcdef0
Name=attachment.delete-on-termination,Values=true

```

For an example of the output for `describe-volumes`, see Example 1.

Example 3: To describe available volumes in a specific Availability Zone

The following `describe-volumes` example describes all volumes that have a status of available and are in the specified Availability Zone.

```
aws ec2 describe-volumes \  
  --filters Name=status,Values=available Name=availability-zone,Values=us-east-1a
```

For an example of the output for `describe-volumes`, see Example 1.

Example 4: To describe volumes based on tags

The following `describe-volumes` example describes all volumes that have the tag key Name and a value that begins with Test. The output is then filtered with a query that displays only the tags and IDs of the volumes.

```
aws ec2 describe-volumes \  
  --filters Name=tag:Name,Values=Test* \  
  --query "Volumes[*].{ID:VolumeId,Tag:Tags}"
```

Output:

```
[  
  {  
    "Tag": [  
      {  
        "Value": "Test2",  
        "Key": "Name"  
      }  
    ],  
    "ID": "vol-1234567890abcdef0"  
  },  
  {  
    "Tag": [  
      {  
        "Value": "Test1",  
        "Key": "Name"  
      }  
    ],  
  },  
]
```

```
    "ID": "vol-049df61146c4d7901"  
  }  
]
```

For additional examples using tag filters, see [Working with tags](#) in the *Amazon EC2 User Guide*.

- For API details, see [DescribeVolumes](#) in *AWS CLI Command Reference*.

describe-vpc-attribute

The following code example shows how to use `describe-vpc-attribute`.

AWS CLI

To describe the `enableDnsSupport` attribute

This example describes the `enableDnsSupport` attribute. This attribute indicates whether DNS resolution is enabled for the VPC. If this attribute is `true`, the Amazon DNS server resolves DNS hostnames for your instances to their corresponding IP addresses; otherwise, it does not.

Command:

```
aws ec2 describe-vpc-attribute --vpc-id vpc-a01106c2 --attribute enableDnsSupport
```

Output:

```
{  
  "VpcId": "vpc-a01106c2",  
  "EnableDnsSupport": {  
    "Value": true  
  }  
}
```

To describe the `enableDnsHostnames` attribute

This example describes the `enableDnsHostnames` attribute. This attribute indicates whether the instances launched in the VPC get DNS hostnames. If this attribute is `true`, instances in the VPC get DNS hostnames; otherwise, they do not.

Command:


```
aws ec2 describe-vpc-attribute --vpc-id vpc-a01106c2 --attribute enableDnsHostnames
```

Output:

```
{
  "VpcId": "vpc-a01106c2",
  "EnableDnsHostnames": {
    "Value": true
  }
}
```

- For API details, see [DescribeVpcAttribute](#) in *AWS CLI Command Reference*.

describe-vpc-classic-link-dns-support

The following code example shows how to use `describe-vpc-classic-link-dns-support`.

AWS CLI**To describe ClassicLink DNS support for your VPCs**

This example describes the ClassicLink DNS support status of all of your VPCs.

Command:

```
aws ec2 describe-vpc-classic-link-dns-support
```

Output:

```
{
  "Vpcs": [
    {
      "VpcId": "vpc-88888888",
      "ClassicLinkDnsSupported": true
    },
    {
      "VpcId": "vpc-1a2b3c4d",
      "ClassicLinkDnsSupported": false
    }
  ]
}
```

- For API details, see [DescribeVpcClassicLinkDnsSupport](#) in *AWS CLI Command Reference*.

describe-vpc-classic-link

The following code example shows how to use `describe-vpc-classic-link`.

AWS CLI

To describe the ClassicLink status of your VPCs

This example lists the ClassicLink status of `vpc-88888888`.

Command:

```
aws ec2 describe-vpc-classic-link --vpc-id vpc-88888888
```

Output:

```
{
  "Vpcs": [
    {
      "ClassicLinkEnabled": true,
      "VpcId": "vpc-88888888",
      "Tags": [
        {
          "Value": "classiclinkvpc",
          "Key": "Name"
        }
      ]
    }
  ]
}
```

This example lists only VPCs that are enabled for Classiclink (the filter value of `is-classic-link-enabled` is set to `true`).

Command:

```
aws ec2 describe-vpc-classic-link --filter "Name=is-classic-link-enabled,Values=true"
```

- For API details, see [DescribeVpcClassicLink](#) in *AWS CLI Command Reference*.

describe-vpc-endpoint-connection-notifications

The following code example shows how to use `describe-vpc-endpoint-connection-notifications`.

AWS CLI

To describe endpoint connection notifications

The following `describe-vpc-endpoint-connection-notifications` example describes all of your endpoint connection notifications.

```
aws ec2 describe-vpc-endpoint-connection-notifications
```

Output:

```
{
  "ConnectionNotificationSet": [
    {
      "ConnectionNotificationState": "Enabled",
      "ConnectionNotificationType": "Topic",
      "ConnectionEvents": [
        "Accept",
        "Reject",
        "Delete",
        "Connect"
      ],
      "ConnectionNotificationId": "vpce-nfn-04bcb952bc8af7abc",
      "ConnectionNotificationArn": "arn:aws:sns:us-
east-1:123456789012:VpceNotification",
      "VpcEndpointId": "vpce-0324151a02f327123"
    }
  ]
}
```

- For API details, see [DescribeVpcEndpointConnectionNotifications](#) in *AWS CLI Command Reference*.

describe-vpc-endpoint-connections

The following code example shows how to use `describe-vpc-endpoint-connections`.

AWS CLI

To describe VPC endpoint connections

This example describes the interface endpoint connections to your endpoint service and filters the results to display endpoints that are PendingAcceptance.

Command:

```
aws ec2 describe-vpc-endpoint-connections --filters Name=vpc-endpoint-  
state,Values=pendingAcceptance
```

Output:

```
{  
  "VpcEndpointConnections": [  
    {  
      "VpcEndpointId": "vpce-0abed31004e618123",  
      "ServiceId": "vpce-svc-0abced088d20def56",  
      "CreationTimestamp": "2017-11-30T10:00:24.350Z",  
      "VpcEndpointState": "pendingAcceptance",  
      "VpcEndpointOwner": "123456789012"  
    }  
  ]  
}
```

- For API details, see [DescribeVpcEndpointConnections](#) in *AWS CLI Command Reference*.

describe-vpc-endpoint-service-configurations

The following code example shows how to use describe-vpc-endpoint-service-configurations.

AWS CLI

To describe endpoint service configurations

The following describe-vpc-endpoint-service-configurations example describes your endpoint service configurations.

```
aws ec2 describe-vpc-endpoint-service-configurations
```

Output:

```
{
  "ServiceConfigurations": [
    {
      "ServiceType": [
        {
          "ServiceType": "GatewayLoadBalancer"
        }
      ],
      "ServiceId": "vpce-svc-012d33a1c4321cab",
      "ServiceName": "com.amazonaws.vpce.us-east-1.vpce-
svc-012d33a1c4321cab",
      "ServiceState": "Available",
      "AvailabilityZones": [
        "us-east-1d"
      ],
      "AcceptanceRequired": false,
      "ManagesVpcEndpoints": false,
      "GatewayLoadBalancerArns": [
        "arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/
gwy/GWLBService/123210844e429123"
      ],
      "Tags": []
    },
    {
      "ServiceType": [
        {
          "ServiceType": "Interface"
        }
      ],
      "ServiceId": "vpce-svc-123cab125efa123",
      "ServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-123cab125efa123",
      "ServiceState": "Available",
      "AvailabilityZones": [
        "us-east-1a"
      ],
      "AcceptanceRequired": true,
      "ManagesVpcEndpoints": false,
      "NetworkLoadBalancerArns": [
        "arn:aws:elasticloadbalancing:us-east-1:123456789012:loadbalancer/
net/NLBforService/1238753950b25123"
      ],
      "BaseEndpointDnsNames": [
```

```

        "vpce-svc-123cab125efa123.us-east-1.vpce.amazonaws.com"
    ],
    "PrivateDnsName": "example.com",
    "PrivateDnsNameConfiguration": {
        "State": "failed",
        "Type": "TXT",
        "Value": "vpce:qUAth3FdeABCApUiXabc",
        "Name": "_1d367jvbg34znqvyeFrj"
    },
    "Tags": []
}
]
}

```

For more information, see [VPC endpoint services](#) in the *Amazon VPC User Guide*.

- For API details, see [DescribeVpcEndpointServiceConfigurations](#) in *AWS CLI Command Reference*.

describe-vpc-endpoint-service-permissions

The following code example shows how to use `describe-vpc-endpoint-service-permissions`.

AWS CLI

To describe endpoint service permissions

This example describes the permissions for the specified endpoint service.

Command:

```
aws ec2 describe-vpc-endpoint-service-permissions --service-id vpce-
svc-03d5ebb7d9579a2b3
```

Output:

```
{
  "AllowedPrincipals": [
    {
      "PrincipalType": "Account",
      "Principal": "arn:aws:iam::123456789012:root"
    }
  ]
}
```

```
]
}
```

- For API details, see [DescribeVpcEndpointServicePermissions](#) in *AWS CLI Command Reference*.

describe-vpc-endpoint-services

The following code example shows how to use `describe-vpc-endpoint-services`.

AWS CLI

Example 1: To describe all VPC endpoint services

The following "describe-vpc-endpoint-services" example lists all VPC endpoint services for an AWS Region.

```
aws ec2 describe-vpc-endpoint-services
```

Output:

```
{
  "ServiceDetails": [
    {
      "ServiceType": [
        {
          "ServiceType": "Gateway"
        }
      ],
      "AcceptanceRequired": false,
      "ServiceName": "com.amazonaws.us-east-1.dynamodb",
      "VpcEndpointPolicySupported": true,
      "Owner": "amazon",
      "AvailabilityZones": [
        "us-east-1a",
        "us-east-1b",
        "us-east-1c",
        "us-east-1d",
        "us-east-1e",
        "us-east-1f"
      ],
      "BaseEndpointDnsNames": [
        "dynamodb.us-east-1.amazonaws.com"
      ]
    }
  ]
}
```

```
    ]
  },
  {
    "ServiceType": [
      {
        "ServiceType": "Interface"
      }
    ],
    "PrivateDnsName": "ec2.us-east-1.amazonaws.com",
    "ServiceName": "com.amazonaws.us-east-1.ec2",
    "VpcEndpointPolicySupported": false,
    "Owner": "amazon",
    "AvailabilityZones": [
      "us-east-1a",
      "us-east-1b",
      "us-east-1c",
      "us-east-1d",
      "us-east-1e",
      "us-east-1f"
    ],
    "AcceptanceRequired": false,
    "BaseEndpointDnsNames": [
      "ec2.us-east-1.vpce.amazonaws.com"
    ]
  },
  {
    "ServiceType": [
      {
        "ServiceType": "Interface"
      }
    ],
    "PrivateDnsName": "ssm.us-east-1.amazonaws.com",
    "ServiceName": "com.amazonaws.us-east-1.ssm",
    "VpcEndpointPolicySupported": true,
    "Owner": "amazon",
    "AvailabilityZones": [
      "us-east-1a",
      "us-east-1b",
      "us-east-1c",
      "us-east-1d",
      "us-east-1e"
    ],
    "AcceptanceRequired": false,
    "BaseEndpointDnsNames": [
```



```

        "ssm.us-east-1.vpce.amazonaws.com"
    ]
}
],
"ServiceNames": [
    "com.amazonaws.us-east-1.dynamodb",
    "com.amazonaws.us-east-1.ec2",
    "com.amazonaws.us-east-1.ec2messages",
    "com.amazonaws.us-east-1.elasticloadbalancing",
    "com.amazonaws.us-east-1.kinesis-streams",
    "com.amazonaws.us-east-1.s3",
    "com.amazonaws.us-east-1.ssm"
]
}

```

For more information, see [View available AWS service names](#) in the *User Guide for AWSPrivateLink*.

Example 2: To describe the details about an endpoint service

The following "describe-vpc-endpoint-services" example lists the details of the Amazon S3 interface endpoint service

```

aws ec2 describe-vpc-endpoint-services \
  --filter "Name=service-type,Values=Interface" Name=service-
name,Values=com.amazonaws.us-east-1.s3

```

Output:

```

{
  "ServiceDetails": [
    {
      "ServiceName": "com.amazonaws.us-east-1.s3",
      "ServiceId": "vpce-svc-081d84efcdEXAMPLE",
      "ServiceType": [
        {
          "ServiceType": "Interface"
        }
      ],
      "AvailabilityZones": [
        "us-east-1a",
        "us-east-1b",
        "us-east-1c",

```

```

        "us-east-1d",
        "us-east-1e",
        "us-east-1f"
    ],
    "Owner": "amazon",
    "BaseEndpointDnsNames": [
        "s3.us-east-1.vpce.amazonaws.com"
    ],
    "VpcEndpointPolicySupported": true,
    "AcceptanceRequired": false,
    "ManagesVpcEndpoints": false,
    "Tags": []
  }
],
"ServiceNames": [
  "com.amazonaws.us-east-1.s3"
]
}

```

For more information, see [View available AWS service names](#) in the *User Guide for AWSPrivateLink*.

- For API details, see [DescribeVpcEndpointServices](#) in *AWS CLI Command Reference*.

describe-vpc-endpoints

The following code example shows how to use `describe-vpc-endpoints`.

AWS CLI

To describe your VPC endpoints

The following `describe-vpc-endpoints` example displays details for all of your VPC endpoints.

```
aws ec2 describe-vpc-endpoints
```

Output:

```
{
  "VpcEndpoints": [
    {
```

```

    "PolicyDocument": "{\n\"Version\": \"2008-10-17\", \"Statement\": [{\n\"Effect
\": \"Allow\", \"Principal\": \"*\", \"Action\": \"*\", \"Resource\": \"*\"}]}",
    "VpcId": "vpc-aabb1122",
    "NetworkInterfaceIds": [],
    "SubnetIds": [],
    "PrivateDnsEnabled": true,
    "State": "available",
    "ServiceName": "com.amazonaws.us-east-1.dynamodb",
    "RouteTableIds": [
        "rtb-3d560345"
    ],
    "Groups": [],
    "VpcEndpointId": "vpce-032a826a",
    "VpcEndpointType": "Gateway",
    "CreationTimestamp": "2017-09-05T20:41:28Z",
    "DnsEntries": [],
    "OwnerId": "123456789012"
},
{
    "PolicyDocument": "{\n  \"Statement\": [\n    {\n      \"Action\": \"*
\", \n      \"Effect\": \"Allow\", \n      \"Principal\": \"*\", \n      \"Resource
\": \"*\">\n    }]\n}",
    "VpcId": "vpc-1a2b3c4d",
    "NetworkInterfaceIds": [
        "eni-2ec2b084",
        "eni-1b4a65cf"
    ],
    "SubnetIds": [
        "subnet-d6fcaa8d",
        "subnet-7b16de0c"
    ],
    "PrivateDnsEnabled": false,
    "State": "available",
    "ServiceName": "com.amazonaws.us-east-1.elasticloadbalancing",
    "RouteTableIds": [],
    "Groups": [
        {
            "GroupName": "default",
            "GroupId": "sg-54e8bf31"
        }
    ],
    "VpcEndpointId": "vpce-0f89a33420c1931d7",
    "VpcEndpointType": "Interface",
    "CreationTimestamp": "2017-09-05T17:55:27.583Z",

```

```

    "DnsEntries": [
      {
        "HostedZoneId": "Z7HUB22UULQXV",
        "DnsName": "vpce-0f89a33420c1931d7-
bluzidnv.elasticloadbalancing.us-east-1.vpce.amazonaws.com"
      },
      {
        "HostedZoneId": "Z7HUB22UULQXV",
        "DnsName": "vpce-0f89a33420c1931d7-bluzidnv-us-
east-1b.elasticloadbalancing.us-east-1.vpce.amazonaws.com"
      },
      {
        "HostedZoneId": "Z7HUB22UULQXV",
        "DnsName": "vpce-0f89a33420c1931d7-bluzidnv-us-
east-1a.elasticloadbalancing.us-east-1.vpce.amazonaws.com"
      }
    ],
    "OwnerId": "123456789012"
  },
  {
    "VpcEndpointId": "vpce-aabbaabbaabbaabba",
    "VpcEndpointType": "GatewayLoadBalancer",
    "VpcId": "vpc-111122223333aabbc",
    "ServiceName": "com.amazonaws.vpce.us-east-1.vpce-
svc-123123a1c43abc123",
    "State": "available",
    "SubnetIds": [
      "subnet-0011aabbcc2233445"
    ],
    "RequesterManaged": false,
    "NetworkInterfaceIds": [
      "eni-01010120203030405"
    ],
    "CreationTimestamp": "2020-11-11T08:06:03.522Z",
    "Tags": [],
    "OwnerId": "123456789012"
  }
]
}

```

For more information, see [VPC endpoints](#) in the *Amazon VPC User Guide*.

- For API details, see [DescribeVpcEndpoints](#) in *AWS CLI Command Reference*.

describe-vpc-peering-connections

The following code example shows how to use `describe-vpc-peering-connections`.

AWS CLI

To describe your VPC peering connections

This example describes all of your VPC peering connections.

Command:

```
aws ec2 describe-vpc-peering-connections
```

Output:

```
{
  "VpcPeeringConnections": [
    {
      "Status": {
        "Message": "Active",
        "Code": "active"
      },
      "Tags": [
        {
          "Value": "Peering-1",
          "Key": "Name"
        }
      ],
      "AccepterVpcInfo": {
        "OwnerId": "111122223333",
        "VpcId": "vpc-1a2b3c4d",
        "CidrBlock": "10.0.1.0/28"
      },
      "VpcPeeringConnectionId": "pcx-11122233",
      "RequesterVpcInfo": {
        "PeeringOptions": {
          "AllowEgressFromLocalVpcToRemoteClassicLink": false,
          "AllowEgressFromLocalClassicLinkToRemoteVpc": false
        },
        "OwnerId": "444455556666",
        "VpcId": "vpc-123abc45",
        "CidrBlock": "192.168.0.0/16"
      }
    }
  ]
}
```

```

    }
  },
  {
    "Status": {
      "Message": "Pending Acceptance by 444455556666",
      "Code": "pending-acceptance"
    },
    "Tags": [],
    "RequesterVpcInfo": {
      "PeeringOptions": {
        "AllowEgressFromLocalVpcToRemoteClassicLink": false,
        "AllowEgressFromLocalClassicLinkToRemoteVpc": false
      },
      "OwnerId": "444455556666",
      "VpcId": "vpc-11aa22bb",
      "CidrBlock": "10.0.0.0/28"
    },
    "VpcPeeringConnectionId": "pcx-abababab",
    "ExpirationTime": "2014-04-03T09:12:43.000Z",
    "AcceptorVpcInfo": {
      "OwnerId": "444455556666",
      "VpcId": "vpc-33cc44dd"
    }
  }
]
}

```

To describe specific VPC peering connections

This example describes all of your VPC peering connections that are in the pending-acceptance state.

Command:

```
aws ec2 describe-vpc-peering-connections --filters Name=status-code,Values=pending-acceptance
```

This example describes all of your VPC peering connections that have the tag Owner=Finance.

Command:

```
aws ec2 describe-vpc-peering-connections --filters Name=tag:Owner,Values=Finance
```

This example describes all of the VPC peering connections you requested for the specified VPC, vpc-1a2b3c4d.

Command:

```
aws ec2 describe-vpc-peering-connections --filters Name=requester-vpc-info.vpc-id,Values=vpc-1a2b3c4d
```

- For API details, see [DescribeVpcPeeringConnections](#) in *AWS CLI Command Reference*.

describe-vpcs

The following code example shows how to use describe-vpcs.

AWS CLI

Example 1: To describe all of your VPCs

The following describe-vpcs example retrieves details about your VPCs.

```
aws ec2 describe-vpcs
```

Output:

```
{
  "Vpcs": [
    {
      "CidrBlock": "30.1.0.0/16",
      "DhcpOptionsId": "dopt-19edf471",
      "State": "available",
      "VpcId": "vpc-0e9801d129EXAMPLE",
      "OwnerId": "111122223333",
      "InstanceTenancy": "default",
      "CidrBlockAssociationSet": [
        {
          "AssociationId": "vpc-cidr-assoc-062c64cfafEXAMPLE",
          "CidrBlock": "30.1.0.0/16",
          "CidrBlockState": {
            "State": "associated"
          }
        }
      ]
    }
  ],
}
```

```
    "IsDefault": false,
    "Tags": [
      {
        "Key": "Name",
        "Value": "Not Shared"
      }
    ]
  },
  {
    "CidrBlock": "10.0.0.0/16",
    "DhcpOptionsId": "dopt-19edf471",
    "State": "available",
    "VpcId": "vpc-06e4ab6c6cEXAMPLE",
    "OwnerId": "222222222222",
    "InstanceTenancy": "default",
    "CidrBlockAssociationSet": [
      {
        "AssociationId": "vpc-cidr-assoc-00b17b4eddEXAMPLE",
        "CidrBlock": "10.0.0.0/16",
        "CidrBlockState": {
          "State": "associated"
        }
      }
    ],
    "IsDefault": false,
    "Tags": [
      {
        "Key": "Name",
        "Value": "Shared VPC"
      }
    ]
  }
]
```

Example 2: To describe a specified VPC

The following `describe-vpcs` example retrieves details for the specified VPC.

```
aws ec2 describe-vpcs \
  --vpc-ids vpc-06e4ab6c6cEXAMPLE
```

Output:


```
{
  "Vpcs": [
    {
      "CidrBlock": "10.0.0.0/16",
      "DhcpOptionsId": "dopt-19edf471",
      "State": "available",
      "VpcId": "vpc-06e4ab6c6cEXAMPLE",
      "OwnerId": "111122223333",
      "InstanceTenancy": "default",
      "CidrBlockAssociationSet": [
        {
          "AssociationId": "vpc-cidr-assoc-00b17b4eddEXAMPLE",
          "CidrBlock": "10.0.0.0/16",
          "CidrBlockState": {
            "State": "associated"
          }
        }
      ],
      "IsDefault": false,
      "Tags": [
        {
          "Key": "Name",
          "Value": "Shared VPC"
        }
      ]
    }
  ]
}
```

- For API details, see [DescribeVpcs](#) in *AWS CLI Command Reference*.

describe-vpn-connections

The following code example shows how to use `describe-vpn-connections`.

AWS CLI

Example 1: To describe your VPN connections

The following `describe-vpn-connections` example describes all of your Site-to-Site VPN connections.

```
aws ec2 describe-vpn-connections
```

Output:

```
{
  "VpnConnections": [
    {
      "CustomerGatewayConfiguration": "...configuration information...",
      "CustomerGatewayId": "cgw-01234567abcde1234",
      "Category": "VPN",
      "State": "available",
      "Type": "ipsec.1",
      "VpnConnectionId": "vpn-1122334455aabbccd",
      "TransitGatewayId": "tgw-00112233445566aab",
      "Options": {
        "EnableAcceleration": false,
        "StaticRoutesOnly": true,
        "LocalIpv4NetworkCidr": "0.0.0.0/0",
        "RemoteIpv4NetworkCidr": "0.0.0.0/0",
        "TunnelInsideIpVersion": "ipv4"
      },
      "Routes": [],
      "Tags": [
        {
          "Key": "Name",
          "Value": "CanadaVPN"
        }
      ],
      "VgwTelemetry": [
        {
          "AcceptedRouteCount": 0,
          "LastStatusChange": "2020-07-29T10:35:11.000Z",
          "OutsideIpAddress": "203.0.113.3",
          "Status": "DOWN",
          "StatusMessage": ""
        },
        {
          "AcceptedRouteCount": 0,
          "LastStatusChange": "2020-09-02T09:09:33.000Z",
          "OutsideIpAddress": "203.0.113.5",
          "Status": "UP",
          "StatusMessage": ""
        }
      ]
    }
  ]
}
```

```
    ]
  }
]
}
```

For more information, see [How AWS Site-to-Site VPN works](#) in the *AWS Site-to-Site VPN User Guide*.

Example 2: To describe your available VPN connections

The following `describe-vpn-connections` example describes your Site-to-Site VPN connections with a state of available.

```
aws ec2 describe-vpn-connections \
  --filters "Name=state,Values=available"
```

For more information, see [How AWS Site-to-Site VPN works](#) in the *AWS Site-to-Site VPN User Guide*.

- For API details, see [DescribeVpnConnections](#) in *AWS CLI Command Reference*.

describe-vpn-gateways

The following code example shows how to use `describe-vpn-gateways`.

AWS CLI

To describe your virtual private gateways

This example describes your virtual private gateways.

Command:

```
aws ec2 describe-vpn-gateways
```

Output:

```
{
  "VpnGateways": [
    {
      "State": "available",
      "Type": "ipsec.1",
```

```
    "VpnGatewayId": "vgw-f211f09b",
    "VpcAttachments": [
      {
        "State": "attached",
        "VpcId": "vpc-98eb5ef5"
      }
    ]
  },
  {
    "State": "available",
    "Type": "ipsec.1",
    "VpnGatewayId": "vgw-9a4cacf3",
    "VpcAttachments": [
      {
        "State": "attaching",
        "VpcId": "vpc-a01106c2"
      }
    ]
  }
]
```

- For API details, see [DescribeVpnGateways](#) in *AWS CLI Command Reference*.

detach-classic-link-vpc

The following code example shows how to use `detach-classic-link-vpc`.

AWS CLI

To unlink (detach) an EC2-Classic instance from a VPC

This example unlinks instance `i-0598c7d356eba48d7` from VPC `vpc-88888888`.

Command:

```
aws ec2 detach-classic-link-vpc --instance-id i-0598c7d356eba48d7 --vpc-id
vpc-88888888
```

Output:

```
{
```

```
"Return": true
}
```

- For API details, see [DetachClassicLinkVpc](#) in *AWS CLI Command Reference*.

detach-internet-gateway

The following code example shows how to use `detach-internet-gateway`.

AWS CLI

To detach an internet gateway from your VPC

The following `detach-internet-gateway` example detaches the specified internet gateway from the specific VPC.

```
aws ec2 detach-internet-gateway \
  --internet-gateway-id igw-0d0fb496b3EXAMPLE \
  --vpc-id vpc-0a60eb65b4EXAMPLE
```

This command produces no output.

For more information, see [Internet gateways](#) in the *Amazon VPC User Guide*.

- For API details, see [DetachInternetGateway](#) in *AWS CLI Command Reference*.

detach-network-interface

The following code example shows how to use `detach-network-interface`.

AWS CLI

To detach a network interface from your instance

This example detaches the specified network interface from the specified instance. If the command succeeds, no output is returned.

Command:

```
aws ec2 detach-network-interface --attachment-id eni-attach-66c4350a
```

- For API details, see [DetachNetworkInterface](#) in *AWS CLI Command Reference*.

detach-verified-access-trust-provider

The following code example shows how to use `detach-verified-access-trust-provider`.

AWS CLI

To detach a trust provider from an instance

The following `detach-verified-access-trust-provider` example detaches the specified Verified Access trust provider from the specified Verified Access instance.

```
aws ec2 detach-verified-access-trust-provider \  
  --verified-access-instance-id vai-0ce000c0b7643abea \  
  --verified-access-trust-provider-id vatp-0bb32de759a3e19e7
```

Output:

```
{  
  "VerifiedAccessTrustProvider": {  
    "VerifiedAccessTrustProviderId": "vatp-0bb32de759a3e19e7",  
    "Description": "Testing Verified Access",  
    "TrustProviderType": "user",  
    "UserTrustProviderType": "iam-identity-center",  
    "PolicyReferenceName": "idc",  
    "CreationTime": "2023-08-25T19:00:38",  
    "LastUpdatedTime": "2023-08-25T19:00:38"  
  },  
  "VerifiedAccessInstance": {  
    "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",  
    "Description": "Testing Verified Access",  
    "VerifiedAccessTrustProviders": [],  
    "CreationTime": "2023-08-25T18:27:56",  
    "LastUpdatedTime": "2023-08-25T18:27:56"  
  }  
}
```

For more information, see [Verified Access instances](#) in the *AWS Verified Access User Guide*.

- For API details, see [DetachVerifiedAccessTrustProvider](#) in *AWS CLI Command Reference*.

detach-volume

The following code example shows how to use `detach-volume`.

AWS CLI

To detach a volume from an instance

This example command detaches the volume (`vol-049df61146c4d7901`) from the instance it is attached to.

Command:

```
aws ec2 detach-volume --volume-id vol-1234567890abcdef0
```

Output:

```
{
  "AttachTime": "2014-02-27T19:23:06.000Z",
  "InstanceId": "i-1234567890abcdef0",
  "VolumeId": "vol-049df61146c4d7901",
  "State": "detaching",
  "Device": "/dev/sdb"
}
```

- For API details, see [DetachVolume](#) in *AWS CLI Command Reference*.

detach-vpn-gateway

The following code example shows how to use `detach-vpn-gateway`.

AWS CLI

To detach a virtual private gateway from your VPC

This example detaches the specified virtual private gateway from the specified VPC. If the command succeeds, no output is returned.

Command:

```
aws ec2 detach-vpn-gateway --vpn-gateway-id vgw-9a4cacf3 --vpc-id vpc-a01106c2
```

- For API details, see [DetachVpnGateway](#) in *AWS CLI Command Reference*.

disable-address-transfer

The following code example shows how to use `disable-address-transfer`.

AWS CLI

To disable an Elastic IP address transfer

The following `disable-address-transfer` example disables Elastic IP address transfer for the specified Elastic IP address.

```
aws ec2 disable-address-transfer \  
  --allocation-id eipalloc-09ad461b0d03f6aaf
```

Output:

```
{  
  "AddressTransfer": {  
    "PublicIp": "100.21.184.216",  
    "AllocationId": "eipalloc-09ad461b0d03f6aaf",  
    "AddressTransferStatus": "disabled"  
  }  
}
```

For more information, see [Transfer Elastic IP addresses](#) in the *Amazon VPC User Guide*.

- For API details, see [DisableAddressTransfer](#) in *AWS CLI Command Reference*.

disable-aws-network-performance-metric-subscription

The following code example shows how to use `disable-aws-network-performance-metric-subscription`.

AWS CLI

To disable a metric subscription

The following `disable-aws-network-performance-metric-subscription` example disables the monitoring of aggregate network latency between the specified source and destination Regions.

```
aws ec2 disable-aws-network-performance-metric-subscription \  
  --source-region us-east-1 --destination-region us-west-2
```



```
--source us-east-1 \  
--destination eu-west-1 \  
--metric aggregate-latency \  
--statistic p50
```

Output:

```
{  
  "Output": true  
}
```

For more information, see [Manage subscriptions](#) in the *Infrastructure Performance User Guide*.

- For API details, see [DisableAwsNetworkPerformanceMetricSubscription](#) in *AWS CLI Command Reference*.

disable-ebs-encryption-by-default

The following code example shows how to use `disable-ebs-encryption-by-default`.

AWS CLI

To disable EBS encryption by default

The following `disable-ebs-encryption-by-default` example disables EBS encryption by default for your AWS account in the current Region.

```
aws ec2 disable-ebs-encryption-by-default
```

Output:

```
{  
  "EbsEncryptionByDefault": false  
}
```

- For API details, see [DisableEbsEncryptionByDefault](#) in *AWS CLI Command Reference*.

disable-fast-launch

The following code example shows how to use `disable-fast-launch`.

AWS CLI

To discontinue fast launching for an image

The following `disable-fast-launch` example discontinues fast launching on the specified AMI, and cleans up existing pre-provisioned snapshots.

```
aws ec2 disable-fast-launch \  
  --image-id ami-01234567890abcdef
```

Output:

```
{  
  "ImageId": "ami-01234567890abcdef",  
  "ResourceType": "snapshot",  
  "SnapshotConfiguration": {},  
  "LaunchTemplate": {  
    "LaunchTemplateId": "lt-01234567890abcdef",  
    "LaunchTemplateName": "EC2FastLaunchDefaultResourceCreation-  
a8c6215d-94e6-441b-9272-dbd1f87b07e2",  
    "Version": "1"  
  },  
  "MaxParallelLaunches": 6,  
  "OwnerId": "0123456789123",  
  "State": "disabling",  
  "StateTransitionReason": "Client.UserInitiated",  
  "StateTransitionTime": "2022-01-27T22:47:29.265000+00:00"  
}
```

For more information about configuring a Windows AMI for faster launching, see [Configure your AMI for faster launching](#) in the *Amazon EC2 User Guide*.

- For API details, see [DisableFastLaunch](#) in *AWS CLI Command Reference*.

`disable-fast-snapshot-restores`

The following code example shows how to use `disable-fast-snapshot-restores`.

AWS CLI

To disable fast snapshot restore

The following `disable-fast-snapshot-restores` example disables fast snapshot restore for the specified snapshot in the specified Availability Zone.

```
aws ec2 disable-fast-snapshot-restores \
  --availability-zones us-east-2a \
  --source-snapshot-ids snap-1234567890abcdef0
```

Output:

```
{
  "Successful": [
    {
      "SnapshotId": "snap-1234567890abcdef0"
      "AvailabilityZone": "us-east-2a",
      "State": "disabling",
      "StateTransitionReason": "Client.UserInitiated",
      "OwnerId": "123456789012",
      "EnablingTime": "2020-01-25T23:57:49.602Z"
    }
  ],
  "Unsuccessful": []
}
```

- For API details, see [DisableFastSnapshotRestores](#) in *AWS CLI Command Reference*.

disable-image-block-public-access

The following code example shows how to use `disable-image-block-public-access`.

AWS CLI

To disable block public access for AMIs in the specified Region

The following `disable-image-block-public-access` example disables block public access for AMIs at the account level in the specified Region.

```
aws ec2 disable-image-block-public-access \
  --region us-east-1
```

Output:

```
{
  "ImageBlockPublicAccessState": "unblocked"
}
```

For more information, see [Block public access to your AMIs](#) in the *Amazon EC2 User Guide*.

- For API details, see [DisableImageBlockPublicAccess](#) in *AWS CLI Command Reference*.

disable-image-deprecation

The following code example shows how to use `disable-image-deprecation`.

AWS CLI

To cancel the deprecation of an AMI

The following `disable-image-deprecation` example cancels the deprecation of an AMI, which removes the `DeprecationTime` field from the `describe-images` output. You must be the AMI owner to perform this procedure.

```
aws ec2 disable-image-deprecation \
  --image-id ami-1234567890abcdef0
```

Output:

```
{
  "RequestID": "11aabb229-4eac-35bd-99ed-be587EXAMPLE",
  "Return": "true"
}
```

For more information, see [Deprecate an AMI <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ami-deprecate.html#deprecate-ami>](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ami-deprecate.html#deprecate-ami) in the *Amazon EC2 User Guide*.

- For API details, see [DisableImageDeprecation](#) in *AWS CLI Command Reference*.

disable-image

The following code example shows how to use `disable-image`.

AWS CLI

To disable an AMI

The following `disable-image` example disables the specified AMI.

```
aws ec2 disable-image \  
  --image-id ami-1234567890abcdef0
```

Output:

```
{  
  "Return": "true"  
}
```

For more information, see [Disable an AMI](#) in the *Amazon EC2 User Guide*.

- For API details, see [DisableImage](#) in *AWS CLI Command Reference*.

disable-ipam-organization-admin-account

The following code example shows how to use `disable-ipam-organization-admin-account`.

AWS CLI

To disable the delegated IPAM admin

In certain scenarios, you'll integrate IPAM with AWS Organizations. When you do that, the AWS Organizations management account delegates an AWS Organizations member account as the IPAM admin.

In this example, you are the AWS Organizations management account that delegated the IPAM admin account and you want to disable that account from being the IPAM admin.

You can use any AWS Region for `--region` when making this request. You don't have to use the Region where you originally delegated the admin, where the IPAM was created, or an IPAM operating Region. If you disable the delegated admin account, you can re-enable it at any time or delegate a new account as IPAM admin.

The following `disable-ipam-organization-admin-account` example disables the delegated IPAM admin in your AWS account.

```
aws ec2 disable-ipam-organization-admin-account \  
  --delegated-admin-account-id 320805250157 \  
  --region us-east-1
```

```
--region ap-south-1
```

Output:

```
{
  "Success": true
}
```

For more information, see [Integrate IPAM with accounts in an AWS Organization](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [DisableIpamOrganizationAdminAccount](#) in *AWS CLI Command Reference*.

disable-serial-console-access

The following code example shows how to use `disable-serial-console-access`.

AWS CLI

To disable access to the EC2 serial console for your account

The following `disable-serial-console-access` example disables account access to the serial console.

```
aws ec2 disable-serial-console-access
```

Output:

```
{
  "SerialConsoleAccessEnabled": false
}
```

For more information, see [EC2 Serial Console](#) in the *Amazon EC2 User Guide*.

- For API details, see [DisableSerialConsoleAccess](#) in *AWS CLI Command Reference*.

disable-transit-gateway-route-table-propagation

The following code example shows how to use `disable-transit-gateway-route-table-propagation`.

AWS CLI

To disable a transit gateway attachment to propagate routes to the specified propagation route table

The following `disable-transit-gateway-route-table-propagation` example disables the specified attachment to propagate routes to the specified propagation route table.

```
aws ec2 disable-transit-gateway-route-table-propagation \  
  --transit-gateway-route-table-id tgw-rtb-0a823edbdeEXAMPLE \  
  --transit-gateway-attachment-id tgw-attach-09b52ccdb5EXAMPLE
```

Output:

```
{  
  "Propagation": {  
    "TransitGatewayAttachmentId": "tgw-attach-09b52ccdb5EXAMPLE",  
    "ResourceId": "vpc-4d7de228",  
    "ResourceType": "vpc",  
    "TransitGatewayRouteTableId": "tgw-rtb-0a823edbdeEXAMPLE",  
    "State": "disabled"  
  }  
}
```

For more information, see [Transit gateway route tables](#) in the *Transit Gateways Guide*.

- For API details, see [DisableTransitGatewayRouteTablePropagation](#) in *AWS CLI Command Reference*.

disable-vgw-route-propagation

The following code example shows how to use `disable-vgw-route-propagation`.

AWS CLI

To disable route propagation

This example disables the specified virtual private gateway from propagating static routes to the specified route table. If the command succeeds, no output is returned.

Command:

```
aws ec2 disable-vgw-route-propagation --route-table-id rtb-22574640 --gateway-id
vgw-9a4cacf3
```

- For API details, see [DisableVgwRoutePropagation](#) in *AWS CLI Command Reference*.

disable-vpc-classic-link-dns-support

The following code example shows how to use `disable-vpc-classic-link-dns-support`.

AWS CLI

To disable ClassicLink DNS support for a VPC

This example disables ClassicLink DNS support for `vpc-88888888`.

Command:

```
aws ec2 disable-vpc-classic-link-dns-support --vpc-id vpc-88888888
```

Output:

```
{
  "Return": true
}
```

- For API details, see [DisableVpcClassicLinkDnsSupport](#) in *AWS CLI Command Reference*.

disable-vpc-classic-link

The following code example shows how to use `disable-vpc-classic-link`.

AWS CLI

To disable ClassicLink for a VPC

This example disables ClassicLink for `vpc-88888888`.

Command:

```
aws ec2 disable-vpc-classic-link --vpc-id vpc-88888888
```


Output:

```
{
  "Return": true
}
```

- For API details, see [DisableVpcClassicLink](#) in *AWS CLI Command Reference*.

disassociate-address

The following code example shows how to use `disassociate-address`.

AWS CLI**To disassociate an Elastic IP addresses in EC2-Classic**

This example disassociates an Elastic IP address from an instance in EC2-Classic. If the command succeeds, no output is returned.

Command:

```
aws ec2 disassociate-address --public-ip 198.51.100.0
```

To disassociate an Elastic IP address in EC2-VPC

This example disassociates an Elastic IP address from an instance in a VPC. If the command succeeds, no output is returned.

Command:

```
aws ec2 disassociate-address --association-id eipassoc-2bebb745
```

- For API details, see [DisassociateAddress](#) in *AWS CLI Command Reference*.

disassociate-client-vpn-target-network

The following code example shows how to use `disassociate-client-vpn-target-network`.

AWS CLI**To disassociate a network from a Client VPN endpoint**

The following `disassociate-client-vpn-target-network` example disassociates the target network that's associated with the `cvpn-assoc-12312312312312312` association ID for the specified Client VPN endpoint.

```
aws ec2 disassociate-client-vpn-target-network \  
  --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde \  
  --association-id cvpn-assoc-12312312312312312
```

Output:

```
{  
  "AssociationId": "cvpn-assoc-12312312312312312",  
  "Status": {  
    "Code": "disassociating"  
  }  
}
```

For more information, see [Target Networks](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [DisassociateClientVpnTargetNetwork](#) in *AWS CLI Command Reference*.

disassociate-iam-instance-profile

The following code example shows how to use `disassociate-iam-instance-profile`.

AWS CLI

To disassociate an IAM instance profile

This example disassociates an IAM instance profile with the association ID `iip-assoc-05020b59952902f5f`.

Command:

```
aws ec2 disassociate-iam-instance-profile --association-id iip-  
assoc-05020b59952902f5f
```

Output:

```
{  
  "IamInstanceProfileAssociation": {
```

```

    "InstanceId": "i-123456789abcde123",
    "State": "disassociating",
    "AssociationId": "iip-assoc-05020b59952902f5f",
    "IamInstanceProfile": {
      "Id": "AIPAI5IVIHMFFYY2DKV5Y",
      "Arn": "arn:aws:iam::123456789012:instance-profile/admin-role"
    }
  }
}

```

- For API details, see [DisassociateIamInstanceProfile](#) in *AWS CLI Command Reference*.

disassociate-instance-event-window

The following code example shows how to use `disassociate-instance-event-window`.

AWS CLI

Example 1: To disassociate one or more instances from an event window

The following `disassociate-instance-event-window` example disassociates one or more instances from an event window. Specify the `instance-event-window-id` parameter to specify the event window. To disassociate instances, specify the `association-target` parameter, and for the parameter values, specify one or more instance IDs.

```

aws ec2 disassociate-instance-event-window \
  --region us-east-1 \
  --instance-event-window-id iew-0abcdef1234567890 \
  --association-target "InstanceIds=i-1234567890abcdef0,i-0598c7d356eba48d7"

```

Output:

```

{
  "InstanceEventWindow": {
    "InstanceEventWindowId": "iew-0abcdef1234567890",
    "Name": "myEventWindowName",
    "CronExpression": "* 21-23 * * 2,3",
    "AssociationTarget": {
      "InstanceIds": [],
      "Tags": [],
      "DedicatedHostIds": []
    }
  },
}

```

```

    "State": "creating"
  }
}

```

For event window constraints, see [Considerations](#) in the Scheduled Events section of the *Amazon EC2 User Guide*.

Example 2: To disassociate instance tags from an event window

The following `disassociate-instance-event-window` example disassociates instance tags from an event window. Specify the `instance-event-window-id` parameter to specify the event window. To disassociate instance tags, specify the `association-target` parameter, and for the parameter values, specify one or more tags.

```

aws ec2 disassociate-instance-event-window \
  --region us-east-1 \
  --instance-event-window-id iew-0abcdef1234567890 \
  --association-target "InstanceTags=[{Key=k2,Value=v2},{Key=k1,Value=v1}]"

```

Output:

```

{
  "InstanceEventWindow": {
    "InstanceEventWindowId": "iew-0abcdef1234567890",
    "Name": "myEventWindowName",
    "CronExpression": "* 21-23 * * 2,3",
    "AssociationTarget": {
      "InstanceIds": [],
      "Tags": [],
      "DedicatedHostIds": []
    },
    "State": "creating"
  }
}

```

For event window constraints, see [Considerations](#) in the Scheduled Events section of the *Amazon EC2 User Guide*.

Example 3: To disassociate a Dedicated Host from an event window

The following `disassociate-instance-event-window` example disassociates a Dedicated Host from an event window. Specify the `instance-event-window-id` parameter to specify

the event window. To disassociate a Dedicated Host, specify the `association-target` parameter, and for the parameter values, specify one or more Dedicated Host IDs.

```
aws ec2 disassociate-instance-event-window \
  --region us-east-1 \
  --instance-event-window-id iew-0abcdef1234567890 \
  --association-target DedicatedHostIds=h-029fa35a02b99801d
```

Output:

```
{
  "InstanceEventWindow": {
    "InstanceEventWindowId": "iew-0abcdef1234567890",
    "Name": "myEventWindowName",
    "CronExpression": "* 21-23 * * 2,3",
    "AssociationTarget": {
      "InstanceIds": [],
      "Tags": [],
      "DedicatedHostIds": []
    },
    "State": "creating"
  }
}
```

For event window constraints, see [Considerations](#) in the Scheduled Events section of the *Amazon EC2 User Guide*.

- For API details, see [DisassociateInstanceEventWindow](#) in *AWS CLI Command Reference*.

disassociate-ipam-resource-discovery

The following code example shows how to use `disassociate-ipam-resource-discovery`.

AWS CLI

To disassociate a resource discovery from an IPAM

In this example, you are an IPAM delegated admin account and you want to disassociate an IPAM resource discovery from your IPAM. You ran the `describe` command and noticed that the `"ResourceDiscoveryStatus": "not-found"` and you want to disassociate it from your IPAM to make room for other associations.

The following `disassociate-ipam-resource-discovery` example disassociates an IPAM resource discovery in your AWS account.

```
aws ec2 disassociate-ipam-resource-discovery \  
  --ipam-resource-discovery-association-id ipam-res-disco-assoc-04382a6346357cf82 \  
  \  
  --region us-east-1
```

Output:

```
{  
  "IpamResourceDiscoveryAssociation": {  
    "OwnerId": "320805250157",  
    "IpamResourceDiscoveryAssociationId": "ipam-res-disco-  
assoc-04382a6346357cf82",  
    "IpamResourceDiscoveryAssociationArn":  
    "arn:aws:ec2::320805250157:ipam-resource-discovery-association/ipam-res-disco-  
assoc-04382a6346357cf82",  
    "IpamResourceDiscoveryId": "ipam-res-disco-0365d2977fc1672fe",  
    "IpamId": "ipam-005f921c17ebd5107",  
    "IpamArn": "arn:aws:ec2::320805250157:ipam/ipam-005f921c17ebd5107",  
    "IpamRegion": "us-east-1",  
    "IsDefault": false,  
    "ResourceDiscoveryStatus": "not-found",  
    "State": "disassociate-in-progress"  
  }  
}
```

For more information, see [Integrate IPAM with accounts outside of your organization](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [DisassociateIpamResourceDiscovery](#) in *AWS CLI Command Reference*.

disassociate-nat-gateway-address

The following code example shows how to use `disassociate-nat-gateway-address`.

AWS CLI

To disassociate an Elastic IP address from a public NAT gateway

The following `disassociate-nat-gateway-address` example disassociates the specified Elastic IP address from the specified public NAT gateway.

```
aws ec2 disassociate-nat-gateway-address \
  --nat-gateway-id nat-1234567890abcdef0 \
  --association-ids eipassoc-0f96bdca17EXAMPLE
```

Output:

```
{
  "NatGatewayId": "nat-1234567890abcdef0",
  "NatGatewayAddresses": [
    {
      "AllocationId": "eipalloc-0be6ecac95EXAMPLE",
      "NetworkInterfaceId": "eni-09cc4b2558794f7f9",
      "PrivateIp": "10.0.0.74",
      "PublicIp": "3.211.231.218",
      "AssociationId": "eipassoc-0f96bdca17EXAMPLE",
      "IsPrimary": false,
      "Status": "disassociating"
    }
  ]
}
```

For more information, see [NAT gateways](#) in the *Amazon VPC User Guide*.

- For API details, see [DisassociateNatGatewayAddress](#) in *AWS CLI Command Reference*.

disassociate-route-table

The following code example shows how to use `disassociate-route-table`.

AWS CLI

To disassociate a route table

This example disassociates the specified route table from the specified subnet. If the command succeeds, no output is returned.

Command:

```
aws ec2 disassociate-route-table --association-id rtbassoc-781d0d1a
```

- For API details, see [DisassociateRouteTable](#) in *AWS CLI Command Reference*.

disassociate-subnet-cidr-block

The following code example shows how to use `disassociate-subnet-cidr-block`.

AWS CLI

To disassociate an IPv6 CIDR block from a subnet

This example disassociates an IPv6 CIDR block from a subnet using the association ID for the CIDR block.

Command:

```
aws ec2 disassociate-subnet-cidr-block --association-id subnet-cidr-assoc-3aa54053
```

Output:

```
{
  "SubnetId": "subnet-5f46ec3b",
  "Ipv6CidrBlockAssociation": {
    "Ipv6CidrBlock": "2001:db8:1234:1a00::/64",
    "AssociationId": "subnet-cidr-assoc-3aa54053",
    "Ipv6CidrBlockState": {
      "State": "disassociating"
    }
  }
}
```

- For API details, see [DisassociateSubnetCidrBlock](#) in *AWS CLI Command Reference*.

disassociate-transit-gateway-multicast-domain

The following code example shows how to use `disassociate-transit-gateway-multicast-domain`.

AWS CLI

To disassociate subnets from a multicast domain

The following `disassociate-transit-gateway-multicast-domain` example disassociates a subnet from the specified multicast domain.

```
aws ec2 disassociate-transit-gateway-multicast-domain \  
  --transit-gateway-attachment-id tgw-attach-070e571cd1EXAMPLE \  
  --subnet-id subnet-000de86e3bEXAMPLE \  
  --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef7EXAMPLE
```

Output:

```
{  
  "Associations": [  
    {  
      "TransitGatewayMulticastDomainId": "tgw-mcast-domain-0c4905cef7EXAMPLE",  
      "TransitGatewayAttachmentId": "tgw-attach-070e571cd1EXAMPLE",  
      "ResourceId": "vpc-7EXAMPLE",  
      "ResourceType": "vpc",  
      "Subnets": [  
        {  
          "SubnetId": "subnet-000de86e3bEXAMPLE",  
          "State": "disassociating"  
        }  
      ]  
    }  
  ]  
}
```

For more information, see [Working with multicast](#) in the *Transit Gateways Guide*'.

- For API details, see [DisassociateTransitGatewayMulticastDomain](#) in *AWS CLI Command Reference*.

disassociate-transit-gateway-route-table

The following code example shows how to use `disassociate-transit-gateway-route-table`.

AWS CLI

To disassociate a transit gateway route table from a resource attachment

The following `disassociate-transit-gateway-route-table` example disassociates the specified attachment from the transit gateway route table.

```
aws ec2 disassociate-transit-gateway-route-table \  
  --transit-gateway-route-table-id tgw-rtb-002573ed1eEXAMPLE \  
  --transit-gateway-attachment-id tgw-attach-08e0bc912cEXAMPLE
```

Output:

```
{  
  "Association": {  
    "TransitGatewayRouteTableId": "tgw-rtb-002573ed1eEXAMPLE",  
    "TransitGatewayAttachmentId": "tgw-attach-08e0bc912cEXAMPLE",  
    "ResourceId": "11460968-4ac1-4fd3-bdb2-00599EXAMPLE",  
    "ResourceType": "direct-connect-gateway",  
    "State": "disassociating"  
  }  
}
```

For more information, see [Transit gateway route tables](#) in the *Transit Gateways Guide*.

- For API details, see [DisassociateTransitGatewayRouteTable](#) in *AWS CLI Command Reference*.

disassociate-vpc-cidr-block

The following code example shows how to use `disassociate-vpc-cidr-block`.

AWS CLI

To disassociate an IPv6 CIDR block from a VPC

This example disassociates an IPv6 CIDR block from a VPC using the association ID for the CIDR block.

Command:

```
aws ec2 disassociate-vpc-cidr-block --association-id vpc-cidr-assoc-eca54085
```

Output:

```
{  
  "Ipv6CidrBlockAssociation": {  
    "Ipv6CidrBlock": "2001:db8:1234:1a00::/56",  
    "AssociationId": "vpc-cidr-assoc-eca54085",  
  }  
}
```

```
    "Ipv6CidrBlockState": {
      "State": "disassociating"
    }
  },
  "VpcId": "vpc-a034d6c4"
}
```

To disassociate an IPv4 CIDR block from a VPC

This example disassociates an IPv4 CIDR block from a VPC.

Command:

```
aws ec2 disassociate-vpc-cidr-block --association-id vpc-cidr-assoc-0287ac6b
```

Output:

```
{
  "CidrBlockAssociation": {
    "AssociationId": "vpc-cidr-assoc-0287ac6b",
    "CidrBlock": "172.18.0.0/16",
    "CidrBlockState": {
      "State": "disassociating"
    }
  },
  "VpcId": "vpc-27621243"
}
```

- For API details, see [DisassociateVpcCidrBlock](#) in *AWS CLI Command Reference*.

enable-address-transfer

The following code example shows how to use `enable-address-transfer`.

AWS CLI

To enable an Elastic IP address transfer

The following `enable-address-transfer` example enables Elastic IP address transfer for the specified Elastic IP address to the specified account.

```
aws ec2 enable-address-transfer \
```

```
--allocation-id eipalloc-09ad461b0d03f6aaf \  
--transfer-account-id 123456789012
```

Output:

```
{  
  "AddressTransfer": {  
    "PublicIp": "100.21.184.216",  
    "AllocationId": "eipalloc-09ad461b0d03f6aaf",  
    "TransferAccountId": "123456789012",  
    "TransferOfferExpirationTimestamp": "2023-02-22T20:51:01.000Z",  
    "AddressTransferStatus": "pending"  
  }  
}
```

For more information, see [Transfer Elastic IP addresses](#) in the *Amazon VPC User Guide*.

- For API details, see [EnableAddressTransfer](#) in *AWS CLI Command Reference*.

enable-aws-network-performance-metric-subscription

The following code example shows how to use `enable-aws-network-performance-metric-subscription`.

AWS CLI**To enable a metric subscription**

The following `enable-aws-network-performance-metric-subscription` example enables the monitoring of aggregate network latency between the specified source and destination Regions.

```
aws ec2 enable-aws-network-performance-metric-subscription \  
  --source us-east-1 \  
  --destination eu-west-1 \  
  --metric aggregate-latency \  
  --statistic p50
```

Output:

```
{
```

```
"Output": true
}
```

For more information, see [Manage subscriptions](#) in the *Infrastructure Performance User Guide*.

- For API details, see [EnableAwsNetworkPerformanceMetricSubscription](#) in *AWS CLI Command Reference*.

enable-ebs-encryption-by-default

The following code example shows how to use `enable-ebs-encryption-by-default`.

AWS CLI

To enable EBS encryption by default

The following `enable-ebs-encryption-by-default` example enables EBS encryption by default for your AWS account in the current Region.

```
aws ec2 enable-ebs-encryption-by-default
```

Output:

```
{
  "EbsEncryptionByDefault": true
}
```

- For API details, see [EnableEbsEncryptionByDefault](#) in *AWS CLI Command Reference*.

enable-fast-launch

The following code example shows how to use `enable-fast-launch`.

AWS CLI

To start fast launching for an image

The following `enable-fast-launch` example starts fast launching on the specified AMI and sets the maximum number of parallel instances to launch to 6. The type of resource to use to pre-provision the AMI is set to `snapshot`, which is also the default value.

```
aws ec2 enable-fast-launch \
  --image-id ami-01234567890abcdef \
  --max-parallel-launches 6 \
  --resource-type snapshot
```

Output:

```
{
  "ImageId": "ami-01234567890abcdef",
  "ResourceType": "snapshot",
  "SnapshotConfiguration": {
    "TargetResourceCount": 10
  },
  "LaunchTemplate": {},
  "MaxParallelLaunches": 6,
  "OwnerId": "0123456789123",
  "State": "enabling",
  "StateTransitionReason": "Client.UserInitiated",
  "StateTransitionTime": "2022-01-27T22:16:03.199000+00:00"
}
```

For more information about configuring a Windows AMI for faster launching, see [Configure your AMI for faster launching](#) in the *Amazon EC2 User Guide*.

- For API details, see [EnableFastLaunch](#) in *AWS CLI Command Reference*.

enable-fast-snapshot-restores

The following code example shows how to use `enable-fast-snapshot-restores`.

AWS CLI

To enable fast snapshot restore

The following `enable-fast-snapshot-restores` example enables fast snapshot restore for the specified snapshot in the specified Availability Zones.

```
aws ec2 enable-fast-snapshot-restores \
  --availability-zones us-east-2a us-east-2b \
  --source-snapshot-ids snap-1234567890abcdef0
```

Output:

```
{
  "Successful": [
    {
      "SnapshotId": "snap-1234567890abcdef0"
      "AvailabilityZone": "us-east-2a",
      "State": "enabling",
      "StateTransitionReason": "Client.UserInitiated",
      "OwnerId": "123456789012",
      "EnablingTime": "2020-01-25T23:57:49.602Z"
    },
    {
      "SnapshotId": "snap-1234567890abcdef0"
      "AvailabilityZone": "us-east-2b",
      "State": "enabling",
      "StateTransitionReason": "Client.UserInitiated",
      "OwnerId": "123456789012",
      "EnablingTime": "2020-01-25T23:57:49.596Z"
    }
  ],
  "Unsuccessful": []
}
```

- For API details, see [EnableFastSnapshotRestores](#) in *AWS CLI Command Reference*.

enable-image-block-public-access

The following code example shows how to use `enable-image-block-public-access`.

AWS CLI**To enable block public access for AMIs in the specified Region**

The following `enable-image-block-public-access` example enables block public access for AMIs at the account level in the specified Region.

```
aws ec2 enable-image-block-public-access \
  --region us-east-1 \
  --image-block-public-access-state block-new-sharing
```

Output:

```
{
  "ImageBlockPublicAccessState": "block-new-sharing"
}
```

For more information, see [Block public access to your AMIs](#) in the *Amazon EC2 User Guide*.

- For API details, see [EnableImageBlockPublicAccess](#) in *AWS CLI Command Reference*.

enable-image-deprecation

The following code example shows how to use `enable-image-deprecation`.

AWS CLI

Example 1: To deprecate an AMI

The following `enable-image-deprecation` example deprecates an AMI on a specific date and time. If you specify a value for seconds, Amazon EC2 rounds the seconds to the nearest minute. You must be the AMI owner to perform this procedure.

```
aws ec2 enable-image-deprecation \
  --image-id ami-1234567890abcdef0 \
  --deprecate-at "2022-10-15T13:17:12.000Z"
```

Output:

```
{
  "RequestID": "59dbff89-35bd-4eac-99ed-be587EXAMPLE",
  "Return": "true"
}
```

For more information, see [Deprecate an AMI <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ami-deprecate.html#deprecate-ami>](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ami-deprecate.html#deprecate-ami) in the *Amazon EC2 User Guide*.

- For API details, see [EnableImageDeprecation](#) in *AWS CLI Command Reference*.

enable-image

The following code example shows how to use `enable-image`.

AWS CLI

To enable an AMI

The following `enable-image` example enables the specified AMI.

```
aws ec2 enable-image \  
  --image-id ami-1234567890abcdef0
```

Output:

```
{  
  "Return": "true"  
}
```

For more information, see [Disable an AMI](#) in the *Amazon EC2 User Guide*.

- For API details, see [EnableImage](#) in *AWS CLI Command Reference*.

`enable-ipam-organization-admin-account`

The following code example shows how to use `enable-ipam-organization-admin-account`.

AWS CLI

To integrate with AWS Organizations and delegate a member account as the IPAM account

The following `enable-ipam-organization-admin-account` example integrates IPAM with AWS Organizations and delegates a member account as the IPAM account.

```
aws ec2 enable-ipam-organization-admin-account \  
  --delegated-admin-account-id 320805250157
```

Output:

```
{  
  "Success": true  
}
```

For more information, see [Integrate IPAM with AWS Organizations](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [EnablepamOrganizationAdminAccount](#) in *AWS CLI Command Reference*.

enable-reachability-analyzer-organization-sharing

The following code example shows how to use `enable-reachability-analyzer-organization-sharing`.

AWS CLI

To enable trusted access for Reachability Analyzer

The following `enable-reachability-analyzer-organization-sharing` example enables trusted access for Reachability Analyzer.

```
aws ec2 enable-reachability-analyzer-organization-sharing
```

This command produces no output.

For more information, see [Cross-account analyses](#) in the *Reachability Analyzer User Guide*.

- For API details, see [EnableReachabilityAnalyzerOrganizationSharing](#) in *AWS CLI Command Reference*.

enable-serial-console-access

The following code example shows how to use `enable-serial-console-access`.

AWS CLI

To enable access to the serial console for your account

The following `enable-serial-console-access` example enables account access to the serial console.

```
aws ec2 enable-serial-console-access
```

Output:

```
{
  "SerialConsoleAccessEnabled": true
}
```

For more information, see [EC2 Serial Console](#) in the *Amazon EC2 User Guide*.

- For API details, see [EnableSerialConsoleAccess](#) in *AWS CLI Command Reference*.

enable-transit-gateway-route-table-propagation

The following code example shows how to use `enable-transit-gateway-route-table-propagation`.

AWS CLI

To enable a transit gateway attachment to propagate routes to the specified propagation route table

The following `enable-transit-gateway-route-table-propagation` example enables the specified attachment to propagate routes to the specified propagation route table.

```
aws ec2 enable-transit-gateway-route-table-propagation \  
  --transit-gateway-route-table-id tgw-rtb-0a823eddbdeEXAMPLE \  
  --transit-gateway-attachment-id tgw-attach-09b52ccdb5EXAMPLE
```

Output:

```
{  
  "Propagation": {  
    "TransitGatewayAttachmentId": "tgw-attach-09b52ccdb5EXAMPLE",  
    "ResourceId": "vpc-4d7de228",  
    "ResourceType": "vpc",  
    "TransitGatewayRouteTableId": "tgw-rtb-0a823eddbdeEXAMPLE",  
    "State": "disabled"  
  }  
}
```

For more information, see [Transit gateway route tables](#) in the *Transit Gateways Guide*.

- For API details, see [EnableTransitGatewayRouteTablePropagation](#) in *AWS CLI Command Reference*.

enable-vgw-route-propagation

The following code example shows how to use `enable-vgw-route-propagation`.

AWS CLI

To enable route propagation

This example enables the specified virtual private gateway to propagate static routes to the specified route table. If the command succeeds, no output is returned.

Command:

```
aws ec2 enable-vgw-route-propagation --route-table-id rtb-22574640 --gateway-id
vgw-9a4cacf3
```

- For API details, see [EnableVgwRoutePropagation](#) in *AWS CLI Command Reference*.

enable-volume-io

The following code example shows how to use `enable-volume-io`.

AWS CLI

To enable I/O for a volume

This example enables I/O on volume `vol-1234567890abcdef0`.

Command:

```
aws ec2 enable-volume-io --volume-id vol-1234567890abcdef0
```

Output:

```
{
  "Return": true
}
```

- For API details, see [EnableVolumelo](#) in *AWS CLI Command Reference*.

enable-vpc-classic-link-dns-support

The following code example shows how to use `enable-vpc-classic-link-dns-support`.

AWS CLI

To enable ClassicLink DNS support for a VPC

This example enables ClassicLink DNS support for vpc-88888888.

Command:

```
aws ec2 enable-vpc-classic-link-dns-support --vpc-id vpc-88888888
```

Output:

```
{
  "Return": true
}
```

- For API details, see [EnableVpcClassicLinkDnsSupport](#) in *AWS CLI Command Reference*.

enable-vpc-classic-link

The following code example shows how to use enable-vpc-classic-link.

AWS CLI

To enable a VPC for ClassicLink

This example enables vpc-88888888 for ClassicLink.

Command:

```
aws ec2 enable-vpc-classic-link --vpc-id vpc-88888888
```

Output:

```
{
  "Return": true
}
```

- For API details, see [EnableVpcClassicLink](#) in *AWS CLI Command Reference*.

export-client-vpn-client-certificate-revocation-list

The following code example shows how to use `export-client-vpn-client-certificate-revocation-list`.

AWS CLI

To export a client certificate revocation list

The following `export-client-vpn-client-certificate-revocation-list` example exports the client certificate revocation list for the specified Client VPN endpoint. In this example, the output is returned in text format to make it easier to read.

```
aws ec2 export-client-vpn-client-certificate-revocation-list \
  --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde \
  --output text
```

Output:

```
-----BEGIN X509 CRL-----
MIICiTCcAFICCCQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAKGA1UEBhMC
VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAStC01BTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWx1eHAd
BgkqhkiG9w0BCQEWEG5vb251QGFTYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI1MjA0NTIxWjCBiDELMAKGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYD
VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAStC01BTSBDb25z
b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWx1eHAdBgkqhkiG9w0BCQEWEG5vb251QGFT
YXpvbi5jb20wZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLyGVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb30hjZnzcVQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVvxYUntneD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJ10ZxBHjJnyp3780D8uTs7fLvJx79LjStb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END X509 CRL-----
STATUS      pending
```

For more information, see [Client Certificate Revocation Lists](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [ExportClientVpnClientCertificateRevocationList](#) in *AWS CLI Command Reference*.

export-client-vpn-client-configuration

The following code example shows how to use `export-client-vpn-client-configuration`.

AWS CLI

To export the client configuration

The following `export-client-vpn-client-configuration` example exports the client configuration for the specified Client VPN endpoint. In this example, the output is returned in text format to make it easier to read.

```
aws ec2 export-client-vpn-client-configuration \  
  --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde \  
  --output text
```

Output:

```
client  
dev tun  
proto udp  
remote cvpn-endpoint-123456789123abcde.prod.clientvpn.ap-south-1.amazonaws.com 443  
remote-random-hostname  
resolv-retry infinite  
nobind  
persist-key  
persist-tun  
remote-cert-tls server  
cipher AES-256-GCM  
verb 3  
<ca>  
-----BEGIN CERTIFICATE-----  
MIICiTCCAfICCCQD6m7oRw0uX0jANBgqhkiG9w0BAQUFADCBiDELMAKGA1UEBhMC  
VVMxCzAJBgNVBAgTAldBMRawDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6  
b24xFDASBgNVBAstC01BTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWx1eHAd  
BgkqhkiG9w0BCQEWEG5vb251QGftYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN  
MTIwNDI1MjA0NTIxWjCBiDELMAKGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRawDgYD  
VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAstC01BTSBDb25z  
b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWx1eHAdBgkqhkiG9w0BCQEWEG5vb251QGft  
YXpvbi5jb20wZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ  
21uUSfwfEvySwTc2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T  
rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
```

```
Ibb30hjZnzcvcQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----
</ca>
reneg-sec 0
```

For more information, see [Client VPN Endpoints](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [ExportClientVpnClientConfiguration](#) in *AWS CLI Command Reference*.

export-image

The following code example shows how to use `export-image`.

AWS CLI

To export a VM from an AMI

The following `export-image` example exports the specified AMI to the specified bucket in the specified format.

```
aws ec2 export-image \
  --image-id ami-1234567890abcdef0 \
  --disk-image-format VMDK \
  --s3-export-location S3Bucket=my-export-bucket,S3Prefix=exports/
```

Output:

```
{
  "DiskImageFormat": "vmdk",
  "ExportImageTaskId": "export-ami-1234567890abcdef0"
  "ImageId": "ami-1234567890abcdef0",
  "RoleName": "vmimport",
  "Progress": "0",
  "S3ExportLocation": {
    "S3Bucket": "my-export-bucket",
    "S3Prefix": "exports/"
  },
  "Status": "active",
  "StatusMessage": "validating"
```



```
}
```

- For API details, see [ExportImage](#) in *AWS CLI Command Reference*.

get-associated-ipv6-pool-cidrs

The following code example shows how to use `get-associated-ipv6-pool-cidrs`.

AWS CLI

To get the associations for an IPv6 address pool

The following `get-associated-ipv6-pool-cidrs` example gets the associations for the specified IPv6 address pool.

```
aws ec2 get-associated-ipv6-pool-cidrs \
  --pool-id ipv6pool-ec2-012345abc12345abc
```

Output:

```
{
  "Ipv6CidrAssociations": [
    {
      "Ipv6Cidr": "2001:db8:1234:1a00::/56",
      "AssociatedResource": "vpc-111111222222333ab"
    }
  ]
}
```

- For API details, see [GetAssociatedIpv6PoolCidrs](#) in *AWS CLI Command Reference*.

get-aws-network-performance-data

The following code example shows how to use `get-aws-network-performance-data`.

AWS CLI

To get network performance data

The following `get-aws-network-performance-data` example retrieves data about the network performance between the specified Regions in the specified time period.

```
aws ec2 get-aws-network-performance-data \  
  --start-time 2022-10-26T12:00:00.000Z \  
  --end-time 2022-10-26T12:30:00.000Z \  
  --data-queries Id=my-query,Source=us-east-1,Destination=eu-  
west-1,Metric=aggregate-latency,Statistic=p50,Period=five-minutes
```

Output:

```
{  
  "DataResponses": [  
    {  
      "Id": "my-query",  
      "Source": "us-east-1",  
      "Destination": "eu-west-1",  
      "Metric": "aggregate-latency",  
      "Statistic": "p50",  
      "Period": "five-minutes",  
      "MetricPoints": [  
        {  
          "StartDate": "2022-10-26T12:00:00+00:00",  
          "EndDate": "2022-10-26T12:05:00+00:00",  
          "Value": 62.44349,  
          "Status": "OK"  
        },  
        {  
          "StartDate": "2022-10-26T12:05:00+00:00",  
          "EndDate": "2022-10-26T12:10:00+00:00",  
          "Value": 62.483498,  
          "Status": "OK"  
        },  
        {  
          "StartDate": "2022-10-26T12:10:00+00:00",  
          "EndDate": "2022-10-26T12:15:00+00:00",  
          "Value": 62.51248,  
          "Status": "OK"  
        },  
        {  
          "StartDate": "2022-10-26T12:15:00+00:00",  
          "EndDate": "2022-10-26T12:20:00+00:00",  
          "Value": 62.635475,  
          "Status": "OK"  
        },  
        {
```

```
        "StartDate": "2022-10-26T12:20:00+00:00",
        "EndDate": "2022-10-26T12:25:00+00:00",
        "Value": 62.733974,
        "Status": "OK"
    },
    {
        "StartDate": "2022-10-26T12:25:00+00:00",
        "EndDate": "2022-10-26T12:30:00+00:00",
        "Value": 62.773975,
        "Status": "OK"
    },
    {
        "StartDate": "2022-10-26T12:30:00+00:00",
        "EndDate": "2022-10-26T12:35:00+00:00",
        "Value": 62.75349,
        "Status": "OK"
    }
]
}
```

For more information, see [Monitor network performance](#) in the *Infrastructure Performance User Guide*.

- For API details, see [GetAwsNetworkPerformanceData](#) in *AWS CLI Command Reference*.

get-capacity-reservation-usage

The following code example shows how to use `get-capacity-reservation-usage`.

AWS CLI

To view capacity reservation usage across AWS accounts

The following `get-capacity-reservation-usage` example displays usage information for the specified capacity reservation.

```
aws ec2 get-capacity-reservation-usage \
    --capacity-reservation-id cr-1234abcd56EXAMPLE
```

Output:

```
{
  "CapacityReservationId": "cr-1234abcd56EXAMPLE ",
  "InstanceUsages": [
    {
      "UsedInstanceCount": 1,
      "AccountId": "123456789012"
    }
  ],
  "AvailableInstanceCount": 4,
  "TotalInstanceCount": 5,
  "State": "active",
  "InstanceType": "t2.medium"
}
```

For more information, see [Viewing Shared Capacity Reservation Usage](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [GetCapacityReservationUsage](#) in *AWS CLI Command Reference*.

get-coip-pool-usage

The following code example shows how to use `get-coip-pool-usage`.

AWS CLI

To get customer-owned IP address pool usage

The following `get-coip-pool-usage` example gets the usage details for the specified customer-owned IP address pool.

```
aws ec2 get-coip-pool-usage \
  --pool-id ipv4pool-coip-123a45678bEXAMPLE
```

Output:

```
{
  "CoipPoolId": "ipv4pool-coip-123a45678bEXAMPLE",
  "CoipAddressUsages": [
    {
      "CoIp": "0.0.0.0"
    }
  ],
}
```

```
{
  "AllocationId": "eipalloc-123ab45c6dEXAMPLE",
  "AwsAccountId": "123456789012",
  "CoIp": "0.0.0.0"
},
{
  "AllocationId": "eipalloc-123ab45c6dEXAMPLE",
  "AwsAccountId": "123456789111",
  "CoIp": "0.0.0.0"
}
],
"LocalGatewayRouteTableId": "lgw-rtb-059615ef7dEXAMPLE"
}
```

For more information, see [Customer-owned IP addresses](#) in the *AWS Outposts User Guide*.

- For API details, see [GetCoipPoolUsage](#) in *AWS CLI Command Reference*.

get-console-output

The following code example shows how to use `get-console-output`.

AWS CLI

Example 1: To get the console output

The following `get-console-output` example gets the console output for the specified Linux instance.

```
aws ec2 get-console-output \
  --instance-id i-1234567890abcdef0
```

Output:

```
{
  "InstanceId": "i-1234567890abcdef0",
  "Timestamp": "2013-07-25T21:23:53.000Z",
  "Output": "..."
}
```

For more information, see [Instance console output](#) in the *Amazon EC2 User Guide*.

Example 2: To get the latest console output

The following `get-console-output` example gets the latest console output for the specified Linux instance.

```
aws ec2 get-console-output \  
  --instance-id i-1234567890abcdef0 \  
  --latest \  
  --output text
```

Output:

```
i-1234567890abcdef0 [ 0.000000] Command line: root=LABEL=/ console=tty1  
console=ttyS0 selinux=0 nvme_core.io_timeout=4294967295  
[ 0.000000] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point  
registers'  
[ 0.000000] x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'  
[ 0.000000] x86/fpu: Supporting XSAVE feature 0x004: 'AVX registers'  
...  
Cloud-init v. 0.7.6 finished at Wed, 09 May 2018 19:01:13 +0000. Datasource  
DataSourceEc2. Up 21.50 seconds  
Amazon Linux AMI release 2018.03  
Kernel 4.14.26-46.32.amzn1.x
```

For more information, see [Instance console output](#) in the *Amazon EC2 User Guide*.

- For API details, see [GetConsoleOutput](#) in *AWS CLI Command Reference*.

get-console-screenshot

The following code example shows how to use `get-console-screenshot`.

AWS CLI

To retrieve a screenshot of a running instance

The following `get-console-screenshot` example retrieves a screenshot of the specified instance in `.jpg` format. The screenshot is returned as a Base64-encoded string.

```
aws ec2 get-console-screenshot \  
  --instance-id i-1234567890abcdef0
```

Output:

```
{
  "ImageData": "997987/8kgj49ikjhewkwe0008084EXAMPLE",
  "InstanceId": "i-1234567890abcdef0"
}
```

- For API details, see [GetConsoleScreenshot](#) in *AWS CLI Command Reference*.

get-default-credit-specification

The following code example shows how to use `get-default-credit-specification`.

AWS CLI**To describe the default credit option**

The following `get-default-credit-specification` example describes the default credit option for T2 instances.

```
aws ec2 get-default-credit-specification \
  --instance-family t2
```

Output:

```
{
  "InstanceFamilyCreditSpecification": {
    "InstanceFamily": "t2",
    "CpuCredits": "standard"
  }
}
```

- For API details, see [GetDefaultCreditSpecification](#) in *AWS CLI Command Reference*.

get-ebs-default-kms-key-id

The following code example shows how to use `get-ebs-default-kms-key-id`.

AWS CLI**To describe your default CMK for EBS encryption**

The following `get-ebs-default-kms-key-id` example describes the default CMK for EBS encryption for your AWS account.

```
aws ec2 get-ebs-default-kms-key-id
```

The output shows the default CMK for EBS encryption, which is an AWS managed CMK with the alias `alias/aws/ebs`.

```
{
  "KmsKeyId": "alias/aws/ebs"
}
```

The following output shows a custom CMK for EBS encryption.

```
{
  "KmsKeyId": "arn:aws:kms:us-
west-2:123456789012:key/0ea3fef3-80a7-4778-9d8c-1c0c6EXAMPLE"
}
```

- For API details, see [GetEbsDefaultKmsKeyId](#) in *AWS CLI Command Reference*.

get-ebs-encryption-by-default

The following code example shows how to use `get-ebs-encryption-by-default`.

AWS CLI

To describe whether EBS encryption by default is enabled

The following `get-ebs-encryption-by-default` example indicates whether EBS encryption by default is enabled for your AWS account in the current Region.

```
aws ec2 get-ebs-encryption-by-default
```

The following output indicates that EBS encryption by default is disabled.

```
{
  "EbsEncryptionByDefault": false
}
```


The following output indicates that EBS encryption by default is enabled.

```
{
  "EbsEncryptionByDefault": true
}
```

- For API details, see [GetEbsEncryptionByDefault](#) in *AWS CLI Command Reference*.

get-flow-logs-integration-template

The following code example shows how to use `get-flow-logs-integration-template`.

AWS CLI

To create a CloudFormation template to automate the integration of VPC flow logs with Amazon Athena

The following `get-flow-logs-integration-template` examples create a CloudFormation template to automate the integration of VPC flow logs with Amazon Athena.

Linux:

```
aws ec2 get-flow-logs-integration-template \
  --flow-log-id fl-1234567890abcdef0 \
  --config-delivery-s3-destination-arn arn:aws:s3:::DOC-EXAMPLE-BUCKET \
  --integrate-services
  AthenaIntegrations='[{IntegrationResultS3DestinationArn=arn:aws:s3:::DOC-EXAMPLE-
BUCKET,PartitionLoadFrequency=none,PartitionStartDate=2021-07-21T00:40:00,PartitionEndDate=2
}{IntegrationResultS3DestinationArn=arn:aws:s3:::DOC-EXAMPLE-
BUCKET,PartitionLoadFrequency=none,PartitionStartDate=2021-07-21T00:40:00,PartitionEndDate=2
```

Windows:

```
aws ec2 get-flow-logs-integration-template ^
  --flow-log-id fl-1234567890abcdef0 ^
  --config-delivery-s3-destination-arn arn:aws:s3:::DOC-EXAMPLE-BUCKET ^
  --integrate-services
  AthenaIntegrations=[{IntegrationResultS3DestinationArn=arn:aws:s3:::DOC-EXAMPLE-
BUCKET,PartitionLoadFrequency=none,PartitionStartDate=2021-07-21T00:40:00,PartitionEndDate=2
}{IntegrationResultS3DestinationArn=arn:aws:s3:::DOC-EXAMPLE-
BUCKET,PartitionLoadFrequency=none,PartitionStartDate=2021-07-21T00:40:00,PartitionEndDate=2
```

Output:

```
{
  "Result": "https://DOC-EXAMPLE-BUCKET.s3.us-east-2.amazonaws.com/
VPCFlowLogsIntegrationTemplate_f1-1234567890abcdef0_Wed%20Jul
%2021%2000%3A57%3A56%20UTC%202021.yml"
}
```

For information on using CloudFormation templates, see [Working with AWS CloudFormation templates](#) in the *AWS CloudFormation User Guide*.

For information on using Amazon Athena and flow logs, see [Query flow logs using Amazon Athena](#) in the *Amazon Virtual Private Cloud User Guide*.

- For API details, see [GetFlowLogsIntegrationTemplate](#) in *AWS CLI Command Reference*.

get-groups-for-capacity-reservation

The following code example shows how to use `get-groups-for-capacity-reservation`.

AWS CLI**To list the resource groups with a Capacity Reservation**

The following `get-groups-for-capacity-reservation` example lists the resource groups to which the specified Capacity Reservation was added.

```
aws ec2 get-groups-for-capacity-reservation \
  --capacity-reservation-id cr-1234abcd56EXAMPLE
```

Output:

```
{
  "CapacityReservationsGroup": [
    {
      "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/my-
resource-group",
      "OwnerId": "123456789012"
    }
  ]
}
```

For more information, see [Working with Capacity Reservations](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [GetGroupsForCapacityReservation](#) in *AWS CLI Command Reference*.

get-host-reservation-purchase-preview

The following code example shows how to use `get-host-reservation-purchase-preview`.

AWS CLI

To get a purchase preview for a Dedicated Host Reservation

This example provides a preview of the costs for a specified Dedicated Host Reservation for the specified Dedicated Host in your account.

Command:

```
aws ec2 get-host-reservation-purchase-preview --offering-id hro-03f707bf363b6b324 --host-id-set h-013abcd2a00cbd123
```

Output:

```
{
  "TotalHourlyPrice": "1.499",
  "Purchase": [
    {
      "HourlyPrice": "1.499",
      "InstanceFamily": "m4",
      "PaymentOption": "NoUpfront",
      "HostIdSet": [
        "h-013abcd2a00cbd123"
      ],
      "UpfrontPrice": "0.000",
      "Duration": 31536000
    }
  ],
  "TotalUpfrontPrice": "0.000"
}
```

- For API details, see [GetHostReservationPurchasePreview](#) in *AWS CLI Command Reference*.

get-image-block-public-access-state

The following code example shows how to use `get-image-block-public-access-state`.

AWS CLI

To get the block public access state for AMIs in the specified Region

The following `get-image-block-public-access-state` example gets the block public access state for AMIs at the account level in the specified Region.

```
aws ec2 get-image-block-public-access-state \  
  --region us-east-1
```

Output:

```
{  
  "ImageBlockPublicAccessState": "block-new-sharing"  
}
```

For more information, see [Block public access to your AMIs](#) in the *Amazon EC2 User Guide*.

- For API details, see [GetImageBlockPublicAccessState](#) in *AWS CLI Command Reference*.

get-instance-types-from-instance-requirements

The following code example shows how to use `get-instance-types-from-instance-requirements`.

AWS CLI

To preview the instance types that match specified attributes

The following `get-instance-types-from-instance-requirements` example first generates a list of all of the possible attributes that can be specified using the `--generate-cli-skeleton` parameter, and saves the list to a JSON file. Then, the JSON file is used to customize the attributes for which to preview matched instance types.

To generate all possible attributes and save the output directly to a JSON file, use the following command.

```
aws ec2 get-instance-types-from-instance-requirements \  
--region us-east-1 \  
--generate-cli-skeleton input > attributes.json
```

Output:

```
{  
  "DryRun": true,  
  "ArchitectureTypes": [  
    "x86_64_mac"  
  ],  
  "VirtualizationTypes": [  
    "paravirtual"  
  ],  
  "InstanceRequirements": {  
    "VCpuCount": {  
      "Min": 0,  
      "Max": 0  
    },  
    "MemoryMiB": {  
      "Min": 0,  
      "Max": 0  
    },  
    "CpuManufacturers": [  
      "intel"  
    ],  
    "MemoryGiBPerVCpu": {  
      "Min": 0.0,  
      "Max": 0.0  
    },  
    "ExcludedInstanceTypes": [  
      ""  
    ],  
    "InstanceGenerations": [  
      "current"  
    ],  
    "SpotMaxPricePercentageOverLowestPrice": 0,  
    "OnDemandMaxPricePercentageOverLowestPrice": 0,  
    "BareMetal": "included",  
    "BurstablePerformance": "excluded",  
    "RequireHibernateSupport": true,  
    "NetworkInterfaceCount": {  
      "Min": 0,  
      "Max": 0  
    }  
  }  
}
```

```
        "Max": 0
    },
    "LocalStorage": "required",
    "LocalStorageTypes": [
        "hdd"
    ],
    "TotalLocalStorageGB": {
        "Min": 0.0,
        "Max": 0.0
    },
    "BaselineEbsBandwidthMbps": {
        "Min": 0,
        "Max": 0
    },
    "AcceleratorTypes": [
        "inference"
    ],
    "AcceleratorCount": {
        "Min": 0,
        "Max": 0
    },
    "AcceleratorManufacturers": [
        "xilinx"
    ],
    "AcceleratorNames": [
        "t4"
    ],
    "AcceleratorTotalMemoryMiB": {
        "Min": 0,
        "Max": 0
    }
}
},
"MaxResults": 0,
"NextToken": ""
}
```

Configure the JSON file. You must provide values for `ArchitectureTypes`, `VirtualizationTypes`, `VcpuCount`, and `MemoryMiB`. You can omit the other attributes. When omitted, default values are used. For a description of each attribute and their default values, see `get-instance-types-from-instance-requirements` <<https://docs.aws.amazon.com/cli/latest/reference/ec2/get-instance-types-from-instance-requirements.html>>.

Preview the instance types that have the attributes specified in `attributes.json`. Specify the name and path to your JSON file by using the `--cli-input-json` parameter. In the following request, the output is formatted as a table.

```
aws ec2 get-instance-types-from-instance-requirements \
  --cli-input-json file://attributes.json \
  --output table
```

Contents of `attributes.json` file:

```
{
  "ArchitectureTypes": [
    "x86_64"
  ],
  "VirtualizationTypes": [
    "hvm"
  ],
  "InstanceRequirements": {
    "VCpuCount": {
      "Min": 4,
      "Max": 6
    },
    "MemoryMiB": {
      "Min": 2048
    },
    "InstanceGenerations": [
      "current"
    ]
  }
}
```

Output:

```
-----
|GetInstanceTypesFromInstanceRequirements|
+-----+
||           InstanceTypes           ||
|+-----+|
||           InstanceType           ||
|+-----+|
||  c4.xlarge                        ||
```

```

|| c5.xlarge           ||
|| c5a.xlarge         ||
|| c5ad.xlarge        ||
|| c5d.xlarge         ||
|| c5n.xlarge         ||
|| d2.xlarge          ||
...

```

For more information about attribute-based instance type selection, see [How attribute-based instance type selection works](#) in the *Amazon EC2 User Guide*.

- For API details, see [GetInstanceTypesFromInstanceRequirements](#) in *AWS CLI Command Reference*.

get-instance-uefi-data

The following code example shows how to use `get-instance-uefi-data`.

AWS CLI

To retrieve UEFI data from an instance

The following `get-instance-uefi-data` example retrieves UEFI data from an instance. If the output is empty, the instance does not contain UEFI data.

```
aws ec2 get-instance-uefi-data \
  --instance-id i-0123456789example
```

Output:

```
{
  "InstanceId": "i-0123456789example",
  "UefiData": "QU1aTlVFRkkf+uLXAAAAAHj5a7fZ9+3dBzxXb/."
  <snipped>
  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAD4L/J/A0Dshho="
}
```

For more information, see [UEFI Secure Boot](#) in the *Amazon EC2 User Guide*.

- For API details, see [GetInstanceUefiData](#) in *AWS CLI Command Reference*.

get-ipam-address-history

The following code example shows how to use `get-ipam-address-history`.

AWS CLI

To get the history of a CIDR

The following `get-ipam-address-history` example gets the history of a CIDR.

(Linux):

```
aws ec2 get-ipam-address-history \  
  --cidr 10.0.0.0/16 \  
  --ipam-scope-id ipam-scope-02fc38cd4c48e7d38 \  
  --start-time 2021-12-08T01:00:00.000Z \  
  --end-time 2021-12-10T01:00:00.000Z
```

(Windows):

```
aws ec2 get-ipam-address-history ^  
  --cidr 10.0.0.0/16 ^  
  --ipam-scope-id ipam-scope-02fc38cd4c48e7d38 ^  
  --start-time 2021-12-08T01:00:00.000Z ^  
  --end-time 2021-12-10T01:00:00.000Z
```

Output:

```
{  
  "HistoryRecords": [  
    {  
      "ResourceOwnerId": "123456789012",  
      "ResourceRegion": "us-west-1",  
      "ResourceType": "vpc",  
      "ResourceId": "vpc-06cbefa9ee907e1c0",  
      "ResourceCidr": "10.0.0.0/16",  
      "ResourceName": "Demo",  
      "ResourceComplianceStatus": "unmanaged",  
      "ResourceOverlapStatus": "overlapping",  
      "VpcId": "vpc-06cbefa9ee907e1c0",  
      "SampledStartTime": "2021-12-08T19:54:57.675000+00:00"  
    },  
    {
```

```

    "ResourceOwnerId": "123456789012",
    "ResourceRegion": "us-east-2",
    "ResourceType": "vpc",
    "ResourceId": "vpc-042702f474812c9ad",
    "ResourceCidr": "10.0.0.0/16",
    "ResourceName": "test",
    "ResourceComplianceStatus": "unmanaged",
    "ResourceOverlapStatus": "overlapping",
    "VpcId": "vpc-042702f474812c9ad",
    "SampledStartTime": "2021-12-08T19:54:59.019000+00:00"
  },
  {
    "ResourceOwnerId": "123456789012",
    "ResourceRegion": "us-east-2",
    "ResourceType": "vpc",
    "ResourceId": "vpc-042b8a44f64267d67",
    "ResourceCidr": "10.0.0.0/16",
    "ResourceName": "tester",
    "ResourceComplianceStatus": "unmanaged",
    "ResourceOverlapStatus": "overlapping",
    "VpcId": "vpc-042b8a44f64267d67",
    "SampledStartTime": "2021-12-08T19:54:59.019000+00:00"
  }
]
}

```

For more information, see [View the history of IP addresses](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [GetIpamAddressHistory](#) in *AWS CLI Command Reference*.

get-ipam-discovered-accounts

The following code example shows how to use `get-ipam-discovered-accounts`.

AWS CLI

To view the accounts discovered by an IPAM

In this scenario, you're a IPAM delegated admin who wants to view the AWS accounts that own resources that the IPAM is discovering.

The `--discovery-region` is the IPAM operating Region you want to view the monitored account statuses in. For example, if you have three IPAM operating Regions, you may want to

make this request three times to view the timestamps specific to discovery in each of those particular Regions.

The following `get-ipam-discovered-accounts` example lists the AWS accounts that own resources that the IPAM is discovering.

```
aws ec2 get-ipam-discovered-accounts \
  --ipam-resource-discovery-id ipam-res-disco-0365d2977fc1672fe \
  --discovery-region us-east-1
```

Output:

```
{
  "IpamDiscoveredAccounts": [
    {
      "AccountId": "149977607591",
      "DiscoveryRegion": "us-east-1",
      "LastAttemptedDiscoveryTime": "2024-02-09T19:04:31.379000+00:00",
      "LastSuccessfulDiscoveryTime": "2024-02-09T19:04:31.379000+00:00"
    }
  ]
}
```

For more information, see [Integrate IPAM with accounts outside of your organization](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [GetIpamDiscoveredAccounts](#) in *AWS CLI Command Reference*.

get-ipam-discovered-public-addresses

The following code example shows how to use `get-ipam-discovered-public-addresses`.

AWS CLI

To view discovered public IP addresses

In this example, you are an IPAM delegated admin and you want to view the IP addresses of resources discovered by IPAM. You can get the resource discovery ID with [describe-ipam-resource-discoveries](#).

The following `get-ipam-discovered-public-addresses` example shows the discovered public IP addresses for a resource discovery.

```
aws ec2 get-ipam-discovered-public-addresses \  
  --ipam-resource-discovery-id ipam-res-disco-0f4ef577a9f37a162 \  
  --address-region us-east-1 \  
  --region us-east-1
```

Output:

```
{  
  "IpamDiscoveredPublicAddresses": [  
    {  
      "IpamResourceDiscoveryId": "ipam-res-disco-0f4ef577a9f37a162",  
      "AddressRegion": "us-east-1",  
      "Address": "54.208.155.7",  
      "AddressOwnerId": "320805250157",  
      "AssociationStatus": "associated",  
      "AddressType": "ec2-public-ip",  
      "VpcId": "vpc-073b294916198ce49",  
      "SubnetId": "subnet-0b6c8a8839e9a4f15",  
      "NetworkInterfaceId": "eni-081c446b5284a5e06",  
      "NetworkInterfaceDescription": "",  
      "InstanceId": "i-07459a6fca5b35823",  
      "Tags": {},  
      "NetworkBorderGroup": "us-east-1c",  
      "SecurityGroups": [  
        {  
          "GroupName": "launch-wizard-2",  
          "GroupId": "sg-0a489dd6a65c244ce"  
        }  
      ],  
      "SampleTime": "2024-04-05T15:13:59.228000+00:00"  
    },  
    {  
      "IpamResourceDiscoveryId": "ipam-res-disco-0f4ef577a9f37a162",  
      "AddressRegion": "us-east-1",  
      "Address": "44.201.251.218",  
      "AddressOwnerId": "470889052923",  
      "AssociationStatus": "associated",  
      "AddressType": "ec2-public-ip",  
      "VpcId": "vpc-6c31a611",  
      "SubnetId": "subnet-062f47608b99834b1",  
      "NetworkInterfaceId": "eni-024845359c2c3ae9b",  
      "NetworkInterfaceDescription": "",  
      "InstanceId": "i-04ef786d9c4e03f41",
```

```
    "Tags": {},
    "NetworkBorderGroup": "us-east-1a",
    "SecurityGroups": [
      {
        "GroupName": "launch-wizard-32",
        "GroupId": "sg-0ed1a426e96a68374"
      }
    ],
    "SampleTime": "2024-04-05T15:13:59.145000+00:00"
  }
}
```

For more information, see [View public IP insights](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [GetIpamDiscoveredPublicAddresses](#) in *AWS CLI Command Reference*.

get-ipam-discovered-resource-cidrs

The following code example shows how to use `get-ipam-discovered-resource-cidrs`.

AWS CLI

To view the IP address CIDRs discovered by an IPAM

In this example, you're a IPAM delegated admin who wants to view details related to the IP address CIDRs for resources that the IPAM is discovering.

To complete this request:

The resource discovery you choose must be associated with the IPAM. The `--resource-region` is the AWS Region where resource was created.

The following `get-ipam-discovered-resource-cidrs` example lists the IP addresses for resources that the IPAM is discovering.

```
aws ec2 get-ipam-discovered-resource-cidrs \
  --ipam-resource-discovery-id ipam-res-disco-0365d2977fc1672fe \
  --resource-region us-east-1
```

Output:

```
{
```

```

{
  "IpamDiscoveredResourceCidrs": [
    {
      "IpamResourceDiscoveryId": "ipam-res-disco-0365d2977fc1672fe",
      "ResourceRegion": "us-east-1",
      "ResourceId": "vpc-0c974c95ca7ceef4a",
      "ResourceOwnerId": "149977607591",
      "ResourceCidr": "172.31.0.0/16",
      "ResourceType": "vpc",
      "ResourceTags": [],
      "IpUsage": 0.375,
      "VpcId": "vpc-0c974c95ca7ceef4a",
      "SampleTime": "2024-02-09T19:15:16.529000+00:00"
    },
    {
      "IpamResourceDiscoveryId": "ipam-res-disco-0365d2977fc1672fe",
      "ResourceRegion": "us-east-1",
      "ResourceId": "subnet-07fe028119082a8c1",
      "ResourceOwnerId": "149977607591",
      "ResourceCidr": "172.31.0.0/20",
      "ResourceType": "subnet",
      "ResourceTags": [],
      "IpUsage": 0.0012,
      "VpcId": "vpc-0c974c95ca7ceef4a",
      "SampleTime": "2024-02-09T19:15:16.529000+00:00"
    },
    {
      "IpamResourceDiscoveryId": "ipam-res-disco-0365d2977fc1672fe",
      "ResourceRegion": "us-east-1",
      "ResourceId": "subnet-0a96893763984cc4e",
      "ResourceOwnerId": "149977607591",
      "ResourceCidr": "172.31.64.0/20",
      "ResourceType": "subnet",
      "ResourceTags": [],
      "IpUsage": 0.0012,
      "VpcId": "vpc-0c974c95ca7ceef4a",
      "SampleTime": "2024-02-09T19:15:16.529000+00:00"
    }
  ]
}

```

For more information, see [Monitor CIDR usage by resource](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [GetIpamDiscoveredResourceCidrs](#) in *AWS CLI Command Reference*.

get-ipam-pool-allocations

The following code example shows how to use `get-ipam-pool-allocations`.

AWS CLI

To get the CIDRs allocated from an IPAM pool

The following `get-ipam-pool-allocations` example gets the CIDRs allocated from an IPAM pool.

(Linux):

```
aws ec2 get-ipam-pool-allocations \  
  --ipam-pool-id ipam-pool-0533048da7d823723 \  
  --filters Name=ipam-pool-allocation-id,Values=ipam-pool-  
alloc-0e6186d73999e47389266a5d6991e6220
```

(Windows):

```
aws ec2 get-ipam-pool-allocations ^  
  --ipam-pool-id ipam-pool-0533048da7d823723 ^  
  --filters Name=ipam-pool-allocation-id,Values=ipam-pool-  
alloc-0e6186d73999e47389266a5d6991e6220
```

Output:

```
{  
  "IpamPoolAllocations": [  
    {  
      "Cidr": "10.0.0.0/16",  
      "IpamPoolAllocationId": "ipam-pool-  
alloc-0e6186d73999e47389266a5d6991e6220",  
      "ResourceType": "custom",  
      "ResourceOwner": "123456789012"  
    }  
  ]  
}
```

- For API details, see [GetIpamPoolAllocations](#) in *AWS CLI Command Reference*.

get-ipam-pool-cidrs

The following code example shows how to use `get-ipam-pool-cidrs`.

AWS CLI

To get the CIDRs provisioned to an IPAM pool

The following `get-ipam-pool-cidrs` example gets the CIDRs provisioned to an IPAM pool.

(Linux):

```
aws ec2 get-ipam-pool-cidrs \  
  --ipam-pool-id ipam-pool-0533048da7d823723 \  
  --filters 'Name=cidr,Values=10.*'
```

(Windows):

```
aws ec2 get-ipam-pool-cidrs ^  
  --ipam-pool-id ipam-pool-0533048da7d823723 ^  
  --filters Name=cidr,Values=10.*
```

Output:

```
{  
  "IpamPoolCidr": {  
    "Cidr": "10.0.0.0/24",  
    "State": "provisioned"  
  }  
}
```

- For API details, see [GetIpamPoolCidrs](#) in *AWS CLI Command Reference*.

get-ipam-resource-cidrs

The following code example shows how to use `get-ipam-resource-cidrs`.

AWS CLI

To get the CIDRs allocated to a resource

The following `get-ipam-resource-cidrs` example gets the CIDRs allocated to a resource.

(Linux):

```
aws ec2 get-ipam-resource-cidrs \  
  --ipam-scope-id ipam-scope-02fc38cd4c48e7d38 \  
  --filters Name=management-state,Values=unmanaged
```

(Windows):

```
aws ec2 get-ipam-resource-cidrs ^  
  --ipam-scope-id ipam-scope-02fc38cd4c48e7d38 ^  
  --filters Name=management-state,Values=unmanaged
```

Output:

```
{  
  "IpamResourceCidrs": [  
    {  
      "IpamId": "ipam-08440e7a3acde3908",  
      "IpamScopeId": "ipam-scope-02fc38cd4c48e7d38",  
      "ResourceRegion": "us-east-2",  
      "ResourceOwnerId": "123456789012",  
      "ResourceId": "vpc-621b8709",  
      "ResourceName": "Default AWS VPC",  
      "ResourceCidr": "172.33.0.0/16",  
      "ResourceType": "vpc",  
      "ResourceTags": [  
        {  
          "Key": "Environment",  
          "Value": "Test"  
        },  
        {  
          "Key": "Name",  
          "Value": "Default AWS VPC"  
        }  
      ],  
      "IpUsage": 0.0039,  
      "ComplianceStatus": "unmanaged",  
      "ManagementState": "unmanaged",  
      "OverlapStatus": "nonoverlapping",  
      "VpcId": "vpc-621b8709"  
    }  
  ]  
}
```

```
}
```

For more information, see [Monitor CIDR usage by resource](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [GetIpamResourceCidrs](#) in *AWS CLI Command Reference*.

get-launch-template-data

The following code example shows how to use `get-launch-template-data`.

AWS CLI

To get instance data for a launch template

This example gets data about the specified instance and uses the `--query` option to return the contents in `LaunchTemplateData`. You can use the output as a base to create a new launch template or launch template version.

Command:

```
aws ec2 get-launch-template-data --instance-id i-0123d646e8048babc --query  
'LaunchTemplateData'
```

Output:

```
{  
  "Monitoring": {},  
  "ImageId": "ami-8c1be5f6",  
  "BlockDeviceMappings": [  
    {  
      "DeviceName": "/dev/xvda",  
      "Ebs": {  
        "DeleteOnTermination": true  
      }  
    }  
  ],  
  "EbsOptimized": false,  
  "Placement": {  
    "Tenancy": "default",  
    "GroupName": "",  
    "AvailabilityZone": "us-east-1a"  
  },  
}
```

```

    "InstanceType": "t2.micro",
    "NetworkInterfaces": [
      {
        "Description": "",
        "NetworkInterfaceId": "eni-35306abc",
        "PrivateIpAddresses": [
          {
            "Primary": true,
            "PrivateIpAddress": "10.0.0.72"
          }
        ],
        "SubnetId": "subnet-7b16de0c",
        "Groups": [
          "sg-7c227019"
        ],
        "Ipv6Addresses": [
          {
            "Ipv6Address": "2001:db8:1234:1a00::123"
          }
        ],
        "PrivateIpAddress": "10.0.0.72"
      }
    ]
  }
}

```

- For API details, see [GetLaunchTemplateData](#) in *AWS CLI Command Reference*.

get-managed-prefix-list-associations

The following code example shows how to use `get-managed-prefix-list-associations`.

AWS CLI

To get prefix list associations

The following `get-managed-prefix-list-associations` example gets the resources that are associated with the specified prefix list.

```

aws ec2 get-managed-prefix-list-associations \
  --prefix-list-id pl-0123456abcabc1

```

Output:

```
{
  "PrefixListAssociations": [
    {
      "ResourceId": "sg-0abc123456abc12345",
      "ResourceOwner": "123456789012"
    }
  ]
}
```

For more information, see [Managed prefix lists](#) in the *Amazon VPC User Guide*.

- For API details, see [GetManagedPrefixListAssociations](#) in *AWS CLI Command Reference*.

get-managed-prefix-list-entries

The following code example shows how to use `get-managed-prefix-list-entries`.

AWS CLI

To get the entries for a prefix list

The following `get-managed-prefix-list-entries` gets the entries for the specified prefix list.

```
aws ec2 get-managed-prefix-list-entries \
  --prefix-list-id pl-0123456abcabc1
```

Output:

```
{
  "Entries": [
    {
      "Cidr": "10.0.0.0/16",
      "Description": "vpc-a"
    },
    {
      "Cidr": "10.2.0.0/16",
      "Description": "vpc-b"
    }
  ]
}
```

For more information, see [Managed prefix lists](#) in the *Amazon VPC User Guide*.

- For API details, see [GetManagedPrefixListEntries](#) in *AWS CLI Command Reference*.

get-network-insights-access-scope-analysis-findings

The following code example shows how to use `get-network-insights-access-scope-analysis-findings`.

AWS CLI

To get the findings of Network Insights access scope analysis

The following `get-network-insights-access-scope-analysis-findings` example gets the selected scope analysis findings in your AWS account.

```
aws ec2 get-network-insights-access-scope-analysis-findings \
  --region us-east-1 \
  --network-insights-access-scope-analysis-id nis \
  --nis-123456789111
```

Output:

```
{
  "NetworkInsightsAccessScopeAnalysisId": "nisa-123456789222",
  "AnalysisFindings": [
    {
      "NetworkInsightsAccessScopeAnalysisId": "nisa-123456789222",
      "NetworkInsightsAccessScopeId": "nis-123456789111",
      "FindingComponents": [
        {
          "SequenceNumber": 1,
          "Component": {
            "Id": "eni-02e3d42d5cceca67d",
            "Arn": "arn:aws:ec2:us-east-1:936459623503:network-
interface/eni-02e3d32d9cceca17d"
          },
          "OutboundHeader": {
            "DestinationAddresses": [
              "0.0.0.0/5",
              "11.0.0.0/8",
              "12.0.0.0/6",
              "128.0.0.0/3",
```

```

        "16.0.0.0/4",
        "160.0.0.0/5",
        "168.0.0.0/6",
        "172.0.0.0/12"
        "8.0.0.0/7"
    ],
    "DestinationPortRanges": [
        {
            "From": 0,
            "To": 65535
        }
    ],
    "Protocol": "6",
    "SourceAddresses": [
        "10.0.2.253/32"
    ],
    "SourcePortRanges": [
        {
            "From": 0,
            "To": 65535
        }
    ]
}, [etc]
]
}
]
}
}

```

For more information, see [Getting started with Network Access Analyzer using the AWS CLI](#) in the *Network Access Analyzer Guide*.

- For API details, see [GetNetworkInsightsAccessScopeAnalysisFindings](#) in *AWS CLI Command Reference*.

get-network-insights-access-scope-content

The following code example shows how to use `get-network-insights-access-scope-content`.

AWS CLI

To get Network Insights access scope content

The following `get-network-insights-access-scope-content` example gets the content of the selected scope analysis ID in your AWS account.

```
aws ec2 get-network-insights-access-scope-content \
  --region us-east-1 \
  --network-insights-access-scope-id nis-123456789222
```

Output:

```
{
  "NetworkInsightsAccessScopeContent": {
    "NetworkInsightsAccessScopeId": "nis-123456789222",
    "MatchPaths": [
      {
        "Source": {
          "ResourceStatement": {
            "ResourceTypes": [
              "AWS::EC2::NetworkInterface"
            ]
          }
        },
        "Destination": {
          "ResourceStatement": {
            "ResourceTypes": [
              "AWS::EC2::InternetGateway"
            ]
          }
        }
      }
    ]
  }
}
```

For more information, see [Getting started with Network Access Analyzer using the AWS CLI](#) in the *Network Access Analyzer Guide*.

- For API details, see [GetNetworkInsightsAccessScopeContent](#) in *AWS CLI Command Reference*.

get-password-data

The following code example shows how to use `get-password-data`.

AWS CLI

To get the encrypted password

This example gets the encrypted password.

Command:

```
aws ec2 get-password-data --instance-id i-1234567890abcdef0
```

Output:

```
{
  "InstanceId": "i-1234567890abcdef0",
  "Timestamp": "2013-08-07T22:18:38.000Z",
  "PasswordData": "gSlJFq+VpcZXqy+iktXMF6NyxQ4qCrT4+ga0uN0enX1MmgXPTj7XEXAMPLE
UQ+YeFfb+L1U4C4AKv652Ux1iRB3CPTY7WmU3TUnhsuBd+p6LVk7T2lKUm160Xbk6WPW1VYYm/TRPB1
e1DQ7PY4an/DgZT4mwcprFfigzhniQgDDe01InvSDcwoUTwNs0Y1S8ouri2W4n5GNlriM3Q0AnNVe1Vz/
53TkDtxbNoU606M1gK9zUWSxqEgwbvV2j8c5rP0WCuaMWSF14ziDu4bd7q+4RSyi8NUsVWnKZ4aEZffu
DPGzKrF5yL1f3etP2L4ZR6CvG7K1hx7VK0QVN32Dajw=="
}
```

To get the decrypted password

This example gets the decrypted password.

Command:

```
aws ec2 get-password-data --instance-id i-1234567890abcdef0 --priv-launch-key C:
\Keys\MyKeyPair.pem
```

Output:

```
{
  "InstanceId": "i-1234567890abcdef0",
  "Timestamp": "2013-08-30T23:18:05.000Z",
  "PasswordData": "&ViJ652e*u"
}
```

- For API details, see [GetPasswordData](#) in *AWS CLI Command Reference*.

get-reserved-instances-exchange-quote

The following code example shows how to use `get-reserved-instances-exchange-quote`.

AWS CLI

To get a quote for exchanging a Convertible Reserved Instance

This example gets the exchange information for the specified Convertible Reserved Instances.

Command:

```
aws ec2 get-reserved-instances-exchange-quote --reserved-instance-ids
7b8750c3-397e-4da4-bbcb-a45ebexample --target-configurations OfferingId=6fea5434-
b379-434c-b07b-a7abexample
```

Output:

```
{
  "CurrencyCode": "USD",
  "ReservedInstanceValueSet": [
    {
      "ReservedInstanceId": "7b8750c3-397e-4da4-bbcb-a45ebexample",
      "ReservationValue": {
        "RemainingUpfrontValue": "0.000000",
        "HourlyPrice": "0.027800",
        "RemainingTotalValue": "730.556200"
      }
    }
  ],
  "PaymentDue": "424.983828",
  "TargetConfigurationValueSet": [
    {
      "TargetConfiguration": {
        "InstanceCount": 5,
        "OfferingId": "6fea5434-b379-434c-b07b-a7abexample"
      },
      "ReservationValue": {
        "RemainingUpfrontValue": "424.983828",
        "HourlyPrice": "0.016000",
        "RemainingTotalValue": "845.447828"
      }
    }
  ]
}
```

```

],
  "IsValidExchange": true,
  "OutputReservedInstancesWillExpireAt": "2020-10-01T13:03:39Z",
  "ReservedInstanceValueRollup": {
    "RemainingUpfrontValue": "0.000000",
    "HourlyPrice": "0.027800",
    "RemainingTotalValue": "730.556200"
  },
  "TargetConfigurationValueRollup": {
    "RemainingUpfrontValue": "424.983828",
    "HourlyPrice": "0.016000",
    "RemainingTotalValue": "845.447828"
  }
}
}

```

- For API details, see [GetReservedInstancesExchangeQuote](#) in *AWS CLI Command Reference*.

get-security-groups-for-vpc

The following code example shows how to use `get-security-groups-for-vpc`.

AWS CLI

To view security groups that can be associated with network interfaces in a specified VPC.

The following `get-security-groups-for-vpc` example shows the security groups that can be associated with network interfaces in the VPC.

```

aws ec2 get-security-groups-for-vpc \
  --vpc-id vpc-6c31a611 \
  --region us-east-1

```

Output:

```

{
  "SecurityGroupForVpcs": [
    {
      "Description": "launch-wizard-36 created 2022-08-29T15:59:35.338Z",
      "GroupName": "launch-wizard-36",
      "OwnerId": "470889052923",
      "GroupId": "sg-007e0c3027ee885f5",
      "Tags": [],

```

```
        "PrimaryVpcId": "vpc-6c31a611"
    },
    {
        "Description": "launch-wizard-18 created 2024-01-19T20:22:27.527Z",
        "GroupName": "launch-wizard-18",
        "OwnerId": "470889052923",
        "GroupId": "sg-0147193bef51c9eef",
        "Tags": [],
        "PrimaryVpcId": "vpc-6c31a611"
    }
}
```

- For API details, see [GetSecurityGroupsForVpc](#) in *AWS CLI Command Reference*.

get-serial-console-access-status

The following code example shows how to use `get-serial-console-access-status`.

AWS CLI

To view the status of account access to the serial console

The following `get-serial-console-access-status` example determines whether serial console access is enabled for your account.

```
aws ec2 get-serial-console-access-status
```

Output:

```
{
  "SerialConsoleAccessEnabled": true
}
```

For more information, see [EC2 Serial Console](#) in the *Amazon EC2 User Guide*.

- For API details, see [GetSerialConsoleAccessStatus](#) in *AWS CLI Command Reference*.

get-spot-placement-scores

The following code example shows how to use `get-spot-placement-scores`.

AWS CLI

To calculate the Spot placement score for specified requirements

The following `get-spot-placement-scores` example first generates a list of all of the possible parameters that can be specified for the Spot placement score configuration using the `--generate-cli-skeleton` parameter, and saves the list to a JSON file. Then, the JSON file is used to configure the requirements to use to calculate the Spot placement score.

To generate all possible parameters that can be specified for the Spot placement score configuration, and save the output directly to a JSON file.

```
aws ec2 get-spot-placement-scores \  
  --region us-east-1 \  
  --generate-cli-skeleton input > attributes.json
```

Output:

```
{  
  "InstanceTypes": [  
    ""  
  ],  
  "TargetCapacity": 0,  
  "TargetCapacityUnitType": "vcpu",  
  "SingleAvailabilityZone": true,  
  "RegionNames": [  
    ""  
  ],  
  "InstanceRequirementsWithMetadata": {  
    "ArchitectureTypes": [  
      "x86_64_mac"  
    ],  
    "VirtualizationTypes": [  
      "hvm"  
    ],  
    "InstanceRequirements": {  
      "VCpuCount": {  
        "Min": 0,  
        "Max": 0  
      },  
      "MemoryMiB": {  
        "Min": 0,  
        "Max": 0  
      }  
    }  
  }  
}
```

```
  },
  "CpuManufacturers": [
    "amd"
  ],
  "MemoryGiBPerVCpu": {
    "Min": 0.0,
    "Max": 0.0
  },
  "ExcludedInstanceTypes": [
    ""
  ],
  "InstanceGenerations": [
    "previous"
  ],
  "SpotMaxPricePercentageOverLowestPrice": 0,
  "OnDemandMaxPricePercentageOverLowestPrice": 0,
  "BareMetal": "excluded",
  "BurstablePerformance": "excluded",
  "RequireHibernateSupport": true,
  "NetworkInterfaceCount": {
    "Min": 0,
    "Max": 0
  },
  "LocalStorage": "included",
  "LocalStorageTypes": [
    "hdd"
  ],
  "TotalLocalStorageGB": {
    "Min": 0.0,
    "Max": 0.0
  },
  "BaselineEbsBandwidthMbps": {
    "Min": 0,
    "Max": 0
  },
  "AcceleratorTypes": [
    "fpga"
  ],
  "AcceleratorCount": {
    "Min": 0,
    "Max": 0
  },
  "AcceleratorManufacturers": [
    "amd"
```

```

    ],
    "AcceleratorNames": [
        "vu9p"
    ],
    "AcceleratorTotalMemoryMiB": {
        "Min": 0,
        "Max": 0
    }
}
},
"DryRun": true,
"MaxResults": 0,
"NextToken": ""
}

```

Configure the JSON file. You must provide a value for `TargetCapacity`. For a description of each parameter and their default values, see [Calculate the Spot placement score \(AWS CLI\)](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/spot-placement-score.html#calculate-sps-cli) <<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/spot-placement-score.html#calculate-sps-cli>>.

Calculate the Spot placement score for the requirements specified in `attributes.json`. Specify the name and path to your JSON file by using the `--cli-input-json` parameter.

```

aws ec2 get-spot-placement-scores \
  --region us-east-1 \
  --cli-input-json file://attributes.json

```

Output if `SingleAvailabilityZone` is set to `false` or omitted (if omitted, it defaults to `false`). A scored list of Regions is returned.

```

"Recommendation": [
  {
    "Region": "us-east-1",
    "Score": 7
  },
  {
    "Region": "us-west-1",
    "Score": 5
  },
  ...

```

Output if `SingleAvailabilityZone` is set to `true`. A scored list of `SingleAvailability Zones` is returned.

```
"Recommendation": [  
  {  
    "Region": "us-east-1",  
    "AvailabilityZoneId": "use1-az1"  
    "Score": 8  
  },  
  {  
    "Region": "us-east-1",  
    "AvailabilityZoneId": "usw2-az3"  
    "Score": 6  
  },  
  ...  
]
```

For more information about calculating a Spot placement score, and for example configurations, see [Calculate a Spot placement score](#) in the *Amazon EC2 User Guide*.

- For API details, see [GetSpotPlacementScores](#) in *AWS CLI Command Reference*.

get-subnet-cidr-reservations

The following code example shows how to use `get-subnet-cidr-reservations`.

AWS CLI

To get information about a subnet CIDR reservation

The following `get-subnet-cidr-reservations` example displays information about the specified subnet CIDR reservation.

```
aws ec2 get-subnet-cidr-reservations \  
  --subnet-id subnet-03c51e2e6cEXAMPLE
```

Output:

```
{  
  "SubnetIpv4CidrReservations": [  
    {  
      "SubnetCidrReservationId": "scr-044f977c4eEXAMPLE",  
      "SubnetId": "subnet-03c51e2e6cEXAMPLE",  
    }  
  ]  
}
```

```

        "Cidr": "10.1.0.16/28",
        "ReservationType": "prefix",
        "OwnerId": "123456789012"
    }
],
"SubnetIpv6CidrReservations": []
}

```

For more information, see [Subnet CIDR reservations](#) in the *Amazon VPC User Guide*.

- For API details, see [GetSubnetCidrReservations](#) in *AWS CLI Command Reference*.

get-transit-gateway-attachment-propagations

The following code example shows how to use `get-transit-gateway-attachment-propagations`.

AWS CLI

To list the route tables to which the specified resource attachment propagates routes

The following `get-transit-gateway-attachment-propagations` example lists the route table to which the specified resource attachment propagates routes.

```

aws ec2 get-transit-gateway-attachment-propagations \
  --transit-gateway-attachment-id tgw-attach-09fbd47ddfEXAMPLE

```

Output:

```

{
  "TransitGatewayAttachmentPropagations": [
    {
      "TransitGatewayRouteTableId": "tgw-rtb-0882c61b97EXAMPLE",
      "State": "enabled"
    }
  ]
}

```

For more information, see [Transit gateway route tables](#) in the *Transit Gateways Guide*.

- For API details, see [GetTransitGatewayAttachmentPropagations](#) in *AWS CLI Command Reference*.

get-transit-gateway-multicast-domain-associations

The following code example shows how to use `get-transit-gateway-multicast-domain-associations`.

AWS CLI

To view the information about the transit gateway multicast domain associations

The following `get-transit-gateway-multicast-domain-associations` example returns the associations for the specified multicast domain.

```
aws ec2 get-transit-gateway-multicast-domain-associations \  
  --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef7EXAMPLE
```

Output:

```
{  
  "MulticastDomainAssociations": [  
    {  
      "TransitGatewayAttachmentId": "tgw-attach-028c1dd0f8EXAMPLE",  
      "ResourceId": "vpc-01128d2c24EXAMPLE",  
      "ResourceType": "vpc",  
      "Subnet": {  
        "SubnetId": "subnet-000de86e3bEXAMPLE",  
        "State": "associated"  
      }  
    },  
    {  
      "TransitGatewayAttachmentId": "tgw-attach-070e571cd1EXAMPLE",  
      "ResourceId": "vpc-7EXAMPLE",  
      "ResourceType": "vpc",  
      "Subnet": {  
        "SubnetId": "subnet-4EXAMPLE",  
        "State": "associated"  
      }  
    },  
    {  
      "TransitGatewayAttachmentId": "tgw-attach-070e571cd1EXAMPLE",  
      "ResourceId": "vpc-7EXAMPLE",  
      "ResourceType": "vpc",  
      "Subnet": {  
        "SubnetId": "subnet-5EXAMPLE",  
        "State": "associated"  
      }  
    }  
  ]  
}
```

```

        "State": "associated"
      }
    },
    {
      "TransitGatewayAttachmentId": "tgw-attach-070e571cd1EXAMPLE",
      "ResourceId": "vpc-7EXAMPLE",
      "ResourceType": "vpc",
      "Subnet": {
        "SubnetId": "subnet-aEXAMPLE",
        "State": "associated"
      }
    },
    {
      "TransitGatewayAttachmentId": "tgw-attach-070e571cd1EXAMPLE",
      "ResourceId": "vpc-7EXAMPLE",
      "ResourceType": "vpc",
      "Subnet": {
        "SubnetId": "subnet-fEXAMPLE",
        "State": "associated"
      }
    }
  ]
}

```

For more information, see [Managing multicast domains](#) in the *Transit Gateways Guide*.

- For API details, see [GetTransitGatewayMulticastDomainAssociations](#) in *AWS CLI Command Reference*.

get-transit-gateway-prefix-list-references

The following code example shows how to use `get-transit-gateway-prefix-list-references`.

AWS CLI

To get prefix list references in a transit gateway route table

The following `get-transit-gateway-prefix-list-references` example gets the prefix list references for the specified transit gateway route table, and filters by the ID of a specific prefix list.

```
aws ec2 get-transit-gateway-prefix-list-references \
```

```
--transit-gateway-route-table-id tgw-rtb-0123456789abcd123 \  
--filters Name=prefix-list-id,Values=pl-1111112222222333
```

Output:

```
{  
  "TransitGatewayPrefixListReferences": [  
    {  
      "TransitGatewayRouteTableId": "tgw-rtb-0123456789abcd123",  
      "PrefixListId": "pl-1111112222222333",  
      "PrefixListOwnerId": "123456789012",  
      "State": "available",  
      "Blackhole": false,  
      "TransitGatewayAttachment": {  
        "TransitGatewayAttachmentId": "tgw-attach-aabbccddaabbccaab",  
        "ResourceType": "vpc",  
        "ResourceId": "vpc-112233445566aabbcc"  
      }  
    }  
  ]  
}
```

For more information, see [Prefix list references](#) in the *Transit Gateways Guide*.

- For API details, see [GetTransitGatewayPrefixListReferences](#) in *AWS CLI Command Reference*.

get-transit-gateway-route-table-associations

The following code example shows how to use `get-transit-gateway-route-table-associations`.

AWS CLI

To get information about the associations for the specified transit gateway route table

The following `get-transit-gateway-route-table-associations` example displays information about the associations for the specified transit gateway route table.

```
aws ec2 get-transit-gateway-route-table-associations \  
  --transit-gateway-route-table-id tgw-rtb-0a823edbdeEXAMPLE
```

Output:

```
{
  "Associations": [
    {
      "TransitGatewayAttachmentId": "tgw-attach-09b52ccdb5EXAMPLE",
      "ResourceId": "vpc-4d7de228",
      "ResourceType": "vpc",
      "State": "associating"
    }
  ]
}
```

For more information, see [Transit gateway route tables](#) in the *Transit Gateways Guide*.

- For API details, see [GetTransitGatewayRouteTableAssociations](#) in *AWS CLI Command Reference*.

get-transit-gateway-route-table-propagations

The following code example shows how to use `get-transit-gateway-route-table-propagations`.

AWS CLI

To display information about the route table propagations for the specified transit gateway route table

The following `get-transit-gateway-route-table-propagations` example returns the route table propagations for the specified route table.

```
aws ec2 get-transit-gateway-route-table-propagations \
  --transit-gateway-route-table-id tgw-rtb-002573ed1eEXAMPLE
```

Output:

```
{
  "TransitGatewayRouteTablePropagations": [
    {
      "TransitGatewayAttachmentId": "tgw-attach-01f8100bc7EXAMPLE",
      "ResourceId": "vpc-3EXAMPLE",
      "ResourceType": "vpc",
      "State": "enabled"
    }
  ]
}
```

```

    },
    {
      "TransitGatewayAttachmentId": "tgw-attach-08e0bc912cEXAMPLE",
      "ResourceId": "11460968-4ac1-4fd3-bdb2-00599EXAMPLE",
      "ResourceType": "direct-connect-gateway",
      "State": "enabled"
    },
    {
      "TransitGatewayAttachmentId": "tgw-attach-0a89069f57EXAMPLE",
      "ResourceId": "8384da05-13ce-4a91-aada-5a1baEXAMPLE",
      "ResourceType": "direct-connect-gateway",
      "State": "enabled"
    }
  ]
}

```

For more information, see [Transit gateway route tables](#) in the *Transit Gateways Guide*.

- For API details, see [GetTransitGatewayRouteTablePropagations](#) in *AWS CLI Command Reference*.

get-verified-access-endpoint-policy

The following code example shows how to use `get-verified-access-endpoint-policy`.

AWS CLI

To get the Verified Access policy of an endpoint

The following `get-verified-access-endpoint-policy` example gets the Verified Access policy of the specified endpoint.

```
aws ec2 get-verified-access-endpoint-policy \
  --verified-access-endpoint-id vae-066fac616d4d546f2
```

Output:

```
{
  "PolicyEnabled": true,
  "PolicyDocument": "permit(principal,action,resource)\nwhen
{\n  context.identity.groups.contains(\"finance\") &&\n
context.identity.email_verified == true\n};"
```

```
}
```

For more information, see [Verified Access policies](#) in the *AWS Verified Access User Guide*.

- For API details, see [GetVerifiedAccessEndpointPolicy](#) in *AWS CLI Command Reference*.

get-verified-access-group-policy

The following code example shows how to use `get-verified-access-group-policy`.

AWS CLI

To get the Verified Access policy of a group

The following `get-verified-access-group-policy` example gets the Verified Access policy of the specified group.

```
aws ec2 get-verified-access-group-policy \  
  --verified-access-group-id vagr-0dbe967baf14b7235
```

Output:

```
{  
  "PolicyEnabled": true,  
  "PolicyDocument": "permit(principal,action,resource)\nwhen  
{\n  context.identity.groups.contains(\"finance\") &&\n  context.identity.email_verified == true\n};"  
}
```

For more information, see [Verified Access groups](#) in the *AWS Verified Access User Guide*.

- For API details, see [GetVerifiedAccessGroupPolicy](#) in *AWS CLI Command Reference*.

get-vpn-connection-device-sample-configuration

The following code example shows how to use `get-vpn-connection-device-sample-configuration`.

AWS CLI

To download a sample configuration file

The following `get-vpn-connection-device-sample-configuration` example downloads the specified sample configuration file. To list the gateway devices with a sample configuration file, call the `get-vpn-connection-device-types` command.

```
aws ec2 get-vpn-connection-device-sample-configuration \  
  --vpn-connection-id vpn-123456789abc01234 \  
  --vpn-connection-device-type-id 5fb390ba
```

Output:

```
{  
  "VpnConnectionDeviceSampleConfiguration": "contents-of-the-sample-configuration-  
file"  
}
```

For more information, see [Download the configuration file](#) in the *AWS Site-to-Site VPN User Guide*.

- For API details, see [GetVpnConnectionDeviceSampleConfiguration](#) in *AWS CLI Command Reference*.

get-vpn-connection-device-types

The following code example shows how to use `get-vpn-connection-device-types`.

AWS CLI

To list gateway devices with a sample configuration file

The following `get-vpn-connection-device-types` example lists the gateway devices from Palo Alto Networks that have sample configuration files.

```
aws ec2 get-vpn-connection-device-types \  
  --query "VpnConnectionDeviceTypes[?Vendor==`Palo Alto Networks`]"
```

Output:

```
[  
  {  
    "VpnConnectionDeviceTypeId": "754a6372",  
    "Vendor": "Palo Alto Networks",
```

```

    "Platform": "PA Series",
    "Software": "PANOS 4.1.2+"
  },
  {
    "VpnConnectionDeviceTypeId": "9612cbed",
    "Vendor": "Palo Alto Networks",
    "Platform": "PA Series",
    "Software": "PANOS 4.1.2+ (GUI)"
  },
  {
    "VpnConnectionDeviceTypeId": "5fb390ba",
    "Vendor": "Palo Alto Networks",
    "Platform": "PA Series",
    "Software": "PANOS 7.0+"
  }
]

```

For more information, see [Download the configuration file](#) in the *AWS Site-to-Site VPN user Guide*.

- For API details, see [GetVpnConnectionDeviceTypes](#) in *AWS CLI Command Reference*.

import-client-vpn-client-certificate-revocation-list

The following code example shows how to use `import-client-vpn-client-certificate-revocation-list`.

AWS CLI

To import a client certificate revocation list

The following `import-client-vpn-client-certificate-revocation-list` example imports a client certificate revocation list to the Client VPN endpoint by specifying the location of the file on the local computer.

```

aws ec2 import-client-vpn-client-certificate-revocation-list \
  --certificate-revocation-list file:///path/to/crl.pem \
  --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde

```

Output:

```
{
```



```
"Return": true
}
```

For more information, see [Client Certificate Revocation Lists](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [ImportClientVpnClientCertificateRevocationList](#) in *AWS CLI Command Reference*.

import-image

The following code example shows how to use `import-image`.

AWS CLI

To import a VM image file as an AMI

The following `import-image` example imports the specified OVA.

```
aws ec2 import-image \  
  --disk-containers Format=ova,UserBucket="{S3Bucket=my-import-bucket,S3Key=vms/my-  
server-vm.ova}"
```

Output:

```
{  
  "ImportTaskId": "import-ami-1234567890abcdef0",  
  "Progress": "2",  
  "SnapshotDetails": [  
    {  
      "DiskImageSize": 0.0,  
      "Format": "ova",  
      "UserBucket": {  
        "S3Bucket": "my-import-bucket",  
        "S3Key": "vms/my-server-vm.ova"  
      }  
    }  
  ],  
  "Status": "active",  
  "StatusMessage": "pending"  
}
```

- For API details, see [ImportImage](#) in *AWS CLI Command Reference*.

import-key-pair

The following code example shows how to use `import-key-pair`.

AWS CLI

To import a public key

First, generate a key pair with the tool of your choice. For example, use this `ssh-keygen` command:

Command:

```
ssh-keygen -t rsa -C "my-key" -f ~/.ssh/my-key
```

Output:

```
Generating public/private rsa key pair.  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /home/ec2-user/.ssh/my-key.  
Your public key has been saved in /home/ec2-user/.ssh/my-key.pub.  
...
```

This example command imports the specified public key.

Command:

```
aws ec2 import-key-pair --key-name "my-key" --public-key-material fileb://~/.ssh/my-key.pub
```

Output:

```
{  
  "KeyName": "my-key",  
  "KeyFingerprint": "1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca"  
}
```

- For API details, see [ImportKeyPair](#) in *AWS CLI Command Reference*.

import-snapshot

The following code example shows how to use `import-snapshot`.

AWS CLI

To import a snapshot

The following `import-snapshot` example imports the specified disk as a snapshot.

```
aws ec2 import-snapshot \  
  --description "My server VMDK" \  
  --disk-container Format=VMDK,UserBucket={S3Bucket=my-import-bucket,S3Key=vms/my-server-vm.vmdk}
```

Output:

```
{  
  "Description": "My server VMDK",  
  "ImportTaskId": "import-snap-1234567890abcdef0",  
  "SnapshotTaskDetail": {  
    "Description": "My server VMDK",  
    "DiskImageSize": "0.0",  
    "Format": "VMDK",  
    "Progress": "3",  
    "Status": "active",  
    "StatusMessage": "pending"  
    "UserBucket": {  
      "S3Bucket": "my-import-bucket",  
      "S3Key": "vms/my-server-vm.vmdk"  
    }  
  }  
}
```

- For API details, see [ImportSnapshot](#) in *AWS CLI Command Reference*.

list-images-in-recycle-bin

The following code example shows how to use `list-images-in-recycle-bin`.

AWS CLI

To list the images in the Recycle Bin

The following `list-images-in-recycle-bin` example lists all of the images that are currently retained in the Recycle Bin.

```
aws ec2 list-images-in-recycle-bin
```

Output:

```
{
  "Images": [
    {
      "RecycleBinEnterTime": "2022-03-14T15:35:08.000Z",
      "Description": "Monthly AMI One",
      "RecycleBinExitTime": "2022-03-15T15:35:08.000Z",
      "Name": "AMI_01",
      "ImageId": "ami-0111222333444abcd"
    }
  ]
}
```

For more information, see [Recover AMIs from the Recycle Bin](#) in the *Amazon Elastic Compute Cloud User Guide*.

- For API details, see [ListImagesInRecycleBin](#) in *AWS CLI Command Reference*.

`list-snapshots-in-recycle-bin`

The following code example shows how to use `list-snapshots-in-recycle-bin`.

AWS CLI

To view snapshots in the Recycle Bin

The following `list-snapshots-in-recycle-bin` example lists information about snapshots in the Recycle Bin, including the snapshot ID, a description of the snapshot, The ID of the volume from which the snapshot was created, the date and time when the snapshot was deleted and it entered the Recycle Bin, and the date and time when the retention period expires.

```
aws ec2 list-snapshots-in-recycle-bin \
  --snapshot-id snap-01234567890abcdef
```

Output:

```
{
  "SnapshotRecycleBinInfo": [
    {
      "Description": "Monthly data backup snapshot",
      "RecycleBinEnterTime": "2022-12-01T13:00:00.000Z",
      "RecycleBinExitTime": "2022-12-15T13:00:00.000Z",
      "VolumeId": "vol-abcdef09876543210",
      "SnapshotId": "snap-01234567890abcdef"
    }
  ]
}
```

For more information about Recycle Bin for Amazon EBS, see [Recover snapshots from the Recycle Bin](#) in the *Amazon EC2 User Guide*.

- For API details, see [ListSnapshotsInRecycleBin](#) in *AWS CLI Command Reference*.

modify-address-attribute

The following code example shows how to use `modify-address-attribute`.

AWS CLI**To modify the domain name attribute associated with an elastic IP address**

The following `modify-address-attribute` examples modify the domain name attribute of an elastic IP address.

Linux:

```
aws ec2 modify-address-attribute \
  --allocation-id eipalloc-abcdef01234567890 \
  --domain-name example.com
```

Windows:

```
aws ec2 modify-address-attribute ^
  --allocation-id eipalloc-abcdef01234567890 ^
  --domain-name example.com
```

Output:

```
{
  "Addresses": [
    {
      "PublicIp": "192.0.2.0",
      "AllocationId": "eipalloc-abcdef01234567890",
      "PtrRecord": "example.net."
      "PtrRecordUpdate": {
        "Value": "example.com.",
        "Status": "PENDING"
      }
    }
  ]
}
```

To monitor the pending change and to view the modified attributes of an elastic IP address, see [describe-addresses-attribute](#) in the *AWS CLI Command Reference*.

- For API details, see [ModifyAddressAttribute](#) in *AWS CLI Command Reference*.

modify-availability-zone-group

The following code example shows how to use `modify-availability-zone-group`.

AWS CLI**To enable a zone group**

The following `modify-availability-zone-group` example enables the specified zone group.

```
aws ec2 modify-availability-zone-group \
  --group-name us-west-2-lax-1 \
  --opt-in-status opted-in
```

Output:

```
{
  "Return": true
}
```

For more information, see [Regions and Zones](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [ModifyAvailabilityZoneGroup](#) in *AWS CLI Command Reference*.

modify-capacity-reservation-fleet

The following code example shows how to use `modify-capacity-reservation-fleet`.

AWS CLI

Example 1: To modify the total target capacity of a Capacity Reservation Fleet

The following `modify-capacity-reservation-fleet` example modifies the total target capacity of the specified Capacity Reservation Fleet. When you modify the total target capacity of a Capacity Reservation Fleet, the Fleet automatically creates new Capacity Reservations, or modifies or cancels existing Capacity Reservations in the Fleet to meet the new total target capacity. You can't attempt additional modifications to a Fleet while it is in the modifying state.

```
aws ec2 modify-capacity-reservation-fleet \  
  --capacity-reservation-fleet-id crf-01234567890abcdef \  
  --total-target-capacity 160
```

Output:

```
{  
  "Return": true  
}
```

Example 2: To modify the end date of a Capacity Reservation Fleet

The following `modify-capacity-reservation-fleet` example modifies the end date of the specified Capacity Reservation Fleet. When you modify the end date for the Fleet, the end dates for all of the individual Capacity Reservations are updated accordingly. You can't attempt additional modifications to a Fleet while it is in the modifying state.

```
aws ec2 modify-capacity-reservation-fleet \  
  --capacity-reservation-fleet-id crf-01234567890abcdef \  
  --end-date 2022-07-04T23:59:59.000Z
```

Output:

```
{
  "Return": true
}
```

For more information about Capacity Reservation Fleets, see [Capacity Reservation Fleets](#) in the *Amazon EC2 User Guide*.

- For API details, see [ModifyCapacityReservationFleet](#) in *AWS CLI Command Reference*.

modify-capacity-reservation

The following code example shows how to use `modify-capacity-reservation`.

AWS CLI**Example 1: To change the number of instances reserved by an existing capacity reservation**

The following `modify-capacity-reservation` example changes the number of instances for which the capacity reservation reserves capacity.

```
aws ec2 modify-capacity-reservation \
  --capacity-reservation-id cr-1234abcd56EXAMPLE \
  --instance-count 5
```

Output:

```
{
  "Return": true
}
```

Example 2: To change the end date and time for an existing capacity reservation

The following `modify-capacity-reservation` example modifies an existing capacity reservation to end at the specified date and time.

```
aws ec2 modify-capacity-reservation \
  --capacity-reservation-id cr-1234abcd56EXAMPLE \
  --end-date-type limited \
  --end-date 2019-08-31T23:59:59Z
```


For more information, see [Modifying a Capacity Reservation](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [ModifyCapacityReservation](#) in *AWS CLI Command Reference*.

modify-client-vpn-endpoint

The following code example shows how to use `modify-client-vpn-endpoint`.

AWS CLI

To modify a Client VPN endpoint

The following `modify-client-vpn-endpoint` example enables client connection logging for the specified Client VPN endpoint.

```
aws ec2 modify-client-vpn-endpoint \  
  --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde \  
  --connection-log-options Enabled=true,CloudwatchLogGroup=ClientVPNLogs
```

Output:

```
{  
  "Return": true  
}
```

For more information, see [Client VPN Endpoints](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [ModifyClientVpnEndpoint](#) in *AWS CLI Command Reference*.

modify-default-credit-specification

The following code example shows how to use `modify-default-credit-specification`.

AWS CLI

To modify the default credit option

The following `modify-default-credit-specification` example modifies the default credit option for T2 instances.

```
aws ec2 modify-default-credit-specification \  
  --credit-specification T2Unlimited
```

```
--instance-family t2 \  
--cpu-credits unlimited
```

Output:

```
{  
  "InstanceFamilyCreditSpecification": {  
    "InstanceFamily": "t2",  
    "CpuCredits": "unlimited"  
  }  
}
```

- For API details, see [ModifyDefaultCreditSpecification](#) in *AWS CLI Command Reference*.

modify-ebs-default-kms-key-id

The following code example shows how to use `modify-ebs-default-kms-key-id`.

AWS CLI**To set your default CMK for EBS encryption**

The following `modify-ebs-default-kms-key-id` example sets the specified CMK as the default CMK for EBS encryption for your AWS account in the current Region.

```
aws ec2 modify-ebs-default-kms-key-id \  
--kms-key-id alias/my-cmk
```

Output:

```
{  
  "KmsKeyId": "arn:aws:kms:us-  
west-2:123456789012:key/0ea3fef3-80a7-4778-9d8c-1c0c6EXAMPLE"  
}
```

- For API details, see [ModifyEbsDefaultKmsKeyId](#) in *AWS CLI Command Reference*.

modify-fleet

The following code example shows how to use `modify-fleet`.

AWS CLI

To scale an EC2 Fleet

The following `modify-fleet` example modifies the target capacity of the specified EC2 Fleet. If the specified value is greater than the current capacity, the EC2 Fleet launches additional instances. If the specified value is less than the current capacity, the EC2 Fleet cancels any open requests and if the termination policy is `terminate`, the EC2 fleet terminates any instances that exceed the new target capacity.

```
aws ec2 modify-fleet \  
  --fleet-ids fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE \  
  --target-capacity-specification TotalTargetCapacity=5
```

Output:

```
{  
  "Return": true  
}
```

For more information, see [Managing an EC2 Fleet](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [ModifyFleet](#) in *AWS CLI Command Reference*.

modify-fpga-image-attribute

The following code example shows how to use `modify-fpga-image-attribute`.

AWS CLI

To modify the attributes of an Amazon FPGA image

This example adds load permissions for account ID 123456789012 for the specified AFI.

Command:

```
aws ec2 modify-fpga-image-attribute --attribute loadPermission --fpga-image-id  
  afi-0d123e123bfc85abc --load-permission Add=[{UserId=123456789012}]
```

Output:

```
{
  "FpgaImageAttribute": {
    "FpgaImageId": "afi-0d123e123bfc85abc",
    "LoadPermissions": [
      {
        "UserId": "123456789012"
      }
    ]
  }
}
```

- For API details, see [ModifyFpgaImageAttribute](#) in *AWS CLI Command Reference*.

modify-hosts

The following code example shows how to use `modify-hosts`.

AWS CLI

Example 1: To enable auto-placement for a Dedicated Host

The following `modify-hosts` example enables auto-placement for a Dedicated Host so that it accepts any untargeted instance launches that match its instance type configuration.

```
aws ec2 modify-hosts \
  --host-id h-06c2f189b4EXAMPLE \
  --auto-placement on
```

Output:

```
{
  "Successful": [
    "h-06c2f189b4EXAMPLE"
  ],
  "Unsuccessful": []
}
```

Example 2: To enable host recovery for a Dedicated Host

The following `modify-hosts` example enables host recovery for the specified Dedicated Host.

```
aws ec2 modify-hosts \  
  --host-id h-06c2f189b4EXAMPLE \  
  --host-recovery on
```

Output:

```
{  
  "Successful": [  
    "h-06c2f189b4EXAMPLE"  
  ],  
  "Unsuccessful": []  
}
```

For more information, see [Modifying Dedicated Host Auto-Placement](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [ModifyHosts](#) in *AWS CLI Command Reference*.

modify-id-format

The following code example shows how to use `modify-id-format`.

AWS CLI

To enable the longer ID format for a resource

The following `modify-id-format` example enables the longer ID format for the instance resource type.

```
aws ec2 modify-id-format \  
  --resource instance \  
  --use-long-ids
```

To disable the longer ID format for a resource

The following `modify-id-format` example disables the longer ID format for the instance resource type.

```
aws ec2 modify-id-format \  
  --resource instance \  
  --use-long-ids
```

```
--no-use-long-ids
```

The following `modify-id-format` example enables the longer ID format for all supported resource types that are within their opt-in period.

```
aws ec2 modify-id-format \  
  --resource all-current \  
  --use-long-ids
```

- For API details, see [ModifyIdFormat](#) in *AWS CLI Command Reference*.

modify-identity-id-format

The following code example shows how to use `modify-identity-id-format`.

AWS CLI

To enable an IAM role to use longer IDs for a resource

The following `modify-identity-id-format` example enables the IAM role `EC2Role` in your AWS account to use long ID format for the `instance` resource type.

```
aws ec2 modify-identity-id-format \  
  --principal-arn arn:aws:iam::123456789012:role/EC2Role \  
  --resource instance \  
  --use-long-ids
```

To enable an IAM user to use longer IDs for a resource

The following `modify-identity-id-format` example enables the IAM user `AdminUser` in your AWS account to use the longer ID format for the `volume` resource type.

```
aws ec2 modify-identity-id-format \  
  --principal-arn arn:aws:iam::123456789012:user/AdminUser \  
  --resource volume \  
  --use-long-ids
```

The following `modify-identity-id-format` example enables the IAM user `AdminUser` in your AWS account to use the longer ID format for all supported resource types that are within their opt-in period.

```
aws ec2 modify-identity-id-format \  
  --principal-arn arn:aws:iam::123456789012:user/AdminUser \  
  --resource all-current \  
  --use-long-ids
```

- For API details, see [ModifyIdentityIdFormat](#) in *AWS CLI Command Reference*.

modify-image-attribute

The following code example shows how to use `modify-image-attribute`.

AWS CLI

Example 1: To make an AMI public

The following `modify-instance-attribute` example makes the specified AMI public.

```
aws ec2 modify-image-attribute \  
  --image-id ami-5731123e \  
  --launch-permission "Add=[{Group=all}]"
```

This command produces no output.

Example 2: To make an AMI private

The following `modify-instance-attribute` example makes the specified AMI private.

```
aws ec2 modify-image-attribute \  
  --image-id ami-5731123e \  
  --launch-permission "Remove=[{Group=all}]"
```

This command produces no output.

Example 3: To grant launch permission to an AWS account

The following `modify-instance-attribute` example grants launch permissions to the specified AWS account.

```
aws ec2 modify-image-attribute \  
  --image-id ami-5731123e \  
  --launch-permission "Add=[{UserId=123456789012}]"
```

This command produces no output.

Example 4: To remove launch permission from an AWS account

The following `modify-instance-attribute` example removes launch permissions from the specified AWS account.

```
aws ec2 modify-image-attribute \  
  --image-id ami-5731123e \  
  --launch-permission "Remove=[{UserId=123456789012}]"
```

- For API details, see [ModifyImageAttribute](#) in *AWS CLI Command Reference*.

modify-instance-attribute

The following code example shows how to use `modify-instance-attribute`.

AWS CLI

Example 1: To modify the instance type

The following `modify-instance-attribute` example modifies the instance type of the specified instance. The instance must be in the stopped state.

```
aws ec2 modify-instance-attribute \  
  --instance-id i-1234567890abcdef0 \  
  --instance-type "{\"Value\": \"m1.small\"}"
```

This command produces no output.

Example 2: To enable enhanced networking on an instance

The following `modify-instance-attribute` example enables enhanced networking for the specified instance. The instance must be in the stopped state.

```
aws ec2 modify-instance-attribute \  
  --instance-id i-1234567890abcdef0 \  
  --sriov-net-support simple
```

This command produces no output.

Example 3: To modify the sourceDestCheck attribute

The following `modify-instance-attribute` example sets the `sourceDestCheck` attribute of the specified instance to `true`. The instance must be in a VPC.

```
aws ec2 modify-instance-attribute --instance-id i-1234567890abcdef0 --source-dest-check "{\"Value\": true}"
```

This command produces no output.

Example 4: To modify the `deleteOnTermination` attribute of the root volume

The following `modify-instance-attribute` example sets the `deleteOnTermination` attribute for the root volume of the specified Amazon EBS-backed instance to `false`. By default, this attribute is `true` for the root volume.

Command:

```
aws ec2 modify-instance-attribute \
  --instance-id i-1234567890abcdef0 \
  --block-device-mappings "[{\"DeviceName\": \"/dev/sda1\", \"Ebs\": {\"DeleteOnTermination\": false}}]"
```

This command produces no output.

Example 5: To modify the user data attached to an instance

The following `modify-instance-attribute` example adds the contents of the file `UserData.txt` as the `UserData` for the specified instance.

Contents of original file `UserData.txt`:

```
#!/bin/bash
yum update -y
service httpd start
chkconfig httpd on
```

The contents of the file must be base64 encoded. The first command converts the text file to base64 and saves it as a new file.

Linux/macOS version of the command:

```
base64 UserData.txt > UserData.base64.txt
```

This command produces no output.

Windows version of the command:

```
certutil -encode UserData.txt tmp.b64 && findstr /v /c:- tmp.b64 >
UserData.base64.txt
```

Output:

```
Input Length = 67
Output Length = 152
CertUtil: -encode command completed successfully.
```

Now you can reference that file in the CLI command that follows:

```
aws ec2 modify-instance-attribute \
  --instance-id=i-09b5a14dbca622e76 \
  --attribute userData --value file://UserData.base64.txt
```

This command produces no output.

For more information, see [User Data and the AWS CLI](#) in the *EC2 User Guide*.

- For API details, see [ModifyInstanceAttribute](#) in *AWS CLI Command Reference*.

modify-instance-capacity-reservation-attributes

The following code example shows how to use `modify-instance-capacity-reservation-attributes`.

AWS CLI

Example 1: To modify an instance's capacity reservation targeting settings

The following `modify-instance-capacity-reservation-attributes` example modifies a stopped instance to target a specific capacity reservation.

```
aws ec2 modify-instance-capacity-reservation-attributes \
  --instance-id i-EXAMPLE8765abcd4e \
  --capacity-reservation-specification
  'CapacityReservationTarget={CapacityReservationId= cr-1234abcd56EXAMPLE }'
```

Output:

```
{
  "Return": true
}
```

Example 2: To modify an instance's capacity reservation targeting settings

The following `modify-instance-capacity-reservation-attributes` example modifies a stopped instance that targets the specified capacity reservation to launch in any capacity reservation that has matching attributes (instance type, platform, Availability Zone) and that has open instance matching criteria.

```
aws ec2 modify-instance-capacity-reservation-attributes \
  --instance-id i-EXAMPLE8765abcd4e \
  --capacity-reservation-specification 'CapacityReservationPreference=open'
```

Output:

```
{
  "Return": true
}
```

For more information, see [Modifying an Instance's Capacity Reservation Settings](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [ModifyInstanceCapacityReservationAttributes](#) in *AWS CLI Command Reference*.

modify-instance-credit-specification

The following code example shows how to use `modify-instance-credit-specification`.

AWS CLI**To modify the credit option for CPU usage of an instance**

This example modifies the credit option for CPU usage of the specified instance in the specified region to "unlimited". Valid credit options are "standard" and "unlimited".

Command:

```
aws ec2 modify-instance-credit-specification --instance-credit-specification
"InstanceId=i-1234567890abcdef0,CpuCredits=unlimited"
```

Output:

```
{
  "SuccessfulInstanceCreditSpecifications": [
    {
      "InstanceId": "i-1234567890abcdef0"
    }
  ],
  "UnsuccessfulInstanceCreditSpecifications": []
}
```

- For API details, see [ModifyInstanceCreditSpecification](#) in *AWS CLI Command Reference*.

modify-instance-event-start-time

The following code example shows how to use `modify-instance-event-start-time`.

AWS CLI

To modify the event start time for an instance

The following `modify-instance-event-start-time` command shows how to modify the event start time for the specified instance. Specify the event ID by using the `--instance-event-id` parameter. Specify the new date and time by using the `--not-before` parameter.

```
aws ec2 modify-instance-event-start-time --instance-id i-1234567890abcdef0
--instance-event-id instance-event-0abcdef1234567890 --not-before
2019-03-25T10:00:00.000
```

Output:

```
"Event": {
  "InstanceEventId": "instance-event-0abcdef1234567890",
  "Code": "system-reboot",
  "Description": "scheduled reboot",
  "NotAfter": "2019-03-25T12:00:00.000Z",
  "NotBefore": "2019-03-25T10:00:00.000Z",
```

```
"NotBeforeDeadline": "2019-04-22T21:00:00.000Z"  
}
```

For more information, see [Working with Instances Scheduled for Reboot](#) in the *Amazon Elastic Compute Cloud User Guide*

- For API details, see [ModifyInstanceEventStartTime](#) in *AWS CLI Command Reference*.

modify-instance-event-window

The following code example shows how to use `modify-instance-event-window`.

AWS CLI

Example 1: To modify the time range of an event window

The following `modify-instance-event-window` example modifies the time range of an event window. Specify the `time-range` parameter to modify the time range. You can't also specify the `cron-expression` parameter.

```
aws ec2 modify-instance-event-window \  
  --region us-east-1 \  
  --instance-event-window-id iew-0abcdef1234567890  
  --time-range StartWeekDay=monday,StartHour=2,EndWeekDay=wednesday,EndHour=8
```

Output:

```
{  
  "InstanceEventWindow": {  
    "InstanceEventWindowId": "iew-0abcdef1234567890",  
    "TimeRanges": [  
      {  
        "StartWeekDay": "monday",  
        "StartHour": 2,  
        "EndWeekDay": "wednesday",  
        "EndHour": 8  
      }  
    ],  
    "Name": "myEventWindowName",  
    "AssociationTarget": {  
      "InstanceIds": [  

```

```

        "i-0abcdef1234567890",
        "i-0be35f9acb8ba01f0"
    ],
    "Tags": [],
    "DedicatedHostIds": []
},
"State": "creating",
"Tags": [
    {
        "Key": "K1",
        "Value": "V1"
    }
]
}
}

```

For event window constraints, see [Considerations](#) in the Scheduled Events section of the *Amazon EC2 User Guide*.

Example 2: To modify a set of time ranges for an event window

The following `modify-instance-event-window` example modifies the time range of an event window. Specify the `time-range` parameter to modify the time range. You can't also specify the `cron-expression` parameter.

```

aws ec2 modify-instance-event-window \
  --region us-east-1 \
  --instance-event-window-id iew-0abcdef1234567890 \
  --time-range '[{"StartWeekDay": "monday", "StartHour": 2, "EndWeekDay":
"wednesday", "EndHour": 8},
    {"StartWeekDay": "thursday", "StartHour": 2, "EndWeekDay": "friday",
"EndHour": 8}]'

```

Output:

```

{
  "InstanceEventWindow": {
    "InstanceEventWindowId": "iew-0abcdef1234567890",
    "TimeRanges": [
      {
        "StartWeekDay": "monday",
        "StartHour": 2,

```

```

        "EndWeekDay": "wednesday",
        "EndHour": 8
    },
    {
        "StartWeekDay": "thursday",
        "StartHour": 2,
        "EndWeekDay": "friday",
        "EndHour": 8
    }
],
"Name": "myEventWindowName",
"AssociationTarget": {
    "InstanceIds": [
        "i-0abcdef1234567890",
        "i-0be35f9acb8ba01f0"
    ],
    "Tags": [],
    "DedicatedHostIds": []
},
"State": "creating",
"Tags": [
    {
        "Key": "K1",
        "Value": "V1"
    }
]
}
}

```

For event window constraints, see [Considerations](#) in the Scheduled Events section of the *Amazon EC2 User Guide*.

Example 3: To modify the cron expression of an event window

The following `modify-instance-event-window` example modifies the cron expression of an event window. Specify the `cron-expression` parameter to modify the cron expression. You can't also specify the `time-range` parameter.

```

aws ec2 modify-instance-event-window \
  --region us-east-1 \
  --instance-event-window-id iew-0abcdef1234567890 \
  --cron-expression "* 21-23 * * 2,3"

```

Output:

```
{
  "InstanceEventWindow": {
    "InstanceEventWindowId": "iew-0abcdef1234567890",
    "Name": "myEventWindowName",
    "CronExpression": "* 21-23 * * 2,3",
    "AssociationTarget": {
      "InstanceIds": [
        "i-0abcdef1234567890",
        "i-0be35f9acb8ba01f0"
      ],
      "Tags": [],
      "DedicatedHostIds": []
    },
    "State": "creating",
    "Tags": [
      {
        "Key": "K1",
        "Value": "V1"
      }
    ]
  }
}
```

For event window constraints, see [Considerations](#) in the Scheduled Events section of the *Amazon EC2 User Guide*.

- For API details, see [ModifyInstanceEventWindow](#) in *AWS CLI Command Reference*.

modify-instance-maintenance-options

The following code example shows how to use `modify-instance-maintenance-options`.

AWS CLI**Example 1: To disable the recovery behavior of an instance**

The following `modify-instance-maintenance-options` example disables simplified automatic recovery for a running or stopped instance.

```
aws ec2 modify-instance-maintenance-options \
  --instance-id i-0abcdef1234567890 \
```



```
--auto-recovery disabled
```

Output:

```
{
  "InstanceId": "i-0abcdef1234567890",
  "AutoRecovery": "disabled"
}
```

For more information, see [Recover your instance](#) in the *Amazon EC2 User Guide for Linux Instances*.

Example 2: To set the recovery behavior of an instance to default

The following `modify-instance-maintenance-options` example sets the automatic recovery behavior to default which enables simplified automatic recovery for supported instance types.

```
aws ec2 modify-instance-maintenance-options \
  --instance-id i-0abcdef1234567890 \
  --auto-recovery default
```

Output:

```
{
  "InstanceId": "i-0abcdef1234567890",
  "AutoRecovery": "default"
}
```

For more information, see [Recover your instance](#) in the *Amazon EC2 User Guide for Linux Instances*.

- For API details, see [ModifyInstanceMaintenanceOptions](#) in *AWS CLI Command Reference*.

modify-instance-metadata-options

The following code example shows how to use `modify-instance-metadata-options`.

AWS CLI**Example 1: To enable IMDSv2**

The following `modify-instance-metadata-options` example configures the use of IMDSv2 on the specified instance.

```
aws ec2 modify-instance-metadata-options \  
  --instance-id i-1234567898abcdef0 \  
  --http-tokens required \  
  --http-endpoint enabled
```

Output:

```
{  
  "InstanceId": "i-1234567898abcdef0",  
  "InstanceMetadataOptions": {  
    "State": "pending",  
    "HttpTokens": "required",  
    "HttpPutResponseHopLimit": 1,  
    "HttpEndpoint": "enabled"  
  }  
}
```

For more information, see [Instance metadata and user data](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

Example 2: To disable instance metadata

The following `modify-instance-metadata-options` example disables the use of all versions of instance metadata on the specified instance.

```
aws ec2 modify-instance-metadata-options \  
  --instance-id i-1234567898abcdef0 \  
  --http-endpoint disabled
```

Output:

```
{  
  "InstanceId": "i-1234567898abcdef0",  
  "InstanceMetadataOptions": {  
    "State": "pending",  
    "HttpTokens": "required",  
    "HttpPutResponseHopLimit": 1,  
    "HttpEndpoint": "disabled"  
  }  
}
```

```
}
```

For more information, see [Instance metadata and user data](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

Example 3: To enable instance metadata IPv6 endpoint for your instance

The following `modify-instance-metadata-options` example shows you how to turn on the IPv6 endpoint for the instance metadata service.

```
aws ec2 modify-instance-metadata-options \
  --instance-id i-1234567898abcdef0 \
  --http-protocol-ipv6 enabled \
  --http-endpoint enabled
```

Output:

```
{
  "InstanceId": "i-1234567898abcdef0",
  "InstanceMetadataOptions": {
    "State": "pending",
    "HttpTokens": "required",
    "HttpPutResponseHopLimit": 1,
    "HttpEndpoint": "enabled",
    "HttpProtocolIpv6": "enabled"
  }
}
```

By default, the IPv6 endpoint is disabled. This is true even if you have launched an instance into an IPv6-only subnet. The IPv6 endpoint for IMDS is only accessible on instances built on the Nitro System. For more information, see [Instance metadata and user data](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [ModifyInstanceMetadataOptions](#) in *AWS CLI Command Reference*.

modify-instance-placement

The following code example shows how to use `modify-instance-placement`.

AWS CLI

Example 1: To remove an instance's affinity with a Dedicated Host

The following `modify-instance-placement` example removes an instance's affinity with a Dedicated Host and enables it to launch on any available Dedicated Host in your account that supports its instance type.

```
aws ec2 modify-instance-placement \  
  --instance-id i-0e6ddf6187EXAMPLE \  
  --affinity default
```

Output:

```
{  
  "Return": true  
}
```

Example 2: To establish affinity between an instance and the specified Dedicated Host

The following `modify-instance-placement` example establishes a launch relationship between an instance and a Dedicated Host. The instance is only able to run on the specified Dedicated Host.

```
aws ec2 modify-instance-placement \  
  --instance-id i-0e6ddf6187EXAMPLE \  
  --affinity host \  
  --host-id i-0e6ddf6187EXAMPLE
```

Output:

```
{  
  "Return": true  
}
```

For more information, see [Modifying Instance Tenancy and Affinity](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

Example 3: To move an instance to a placement group

The following `modify-instance-placement` example moves an instance to a placement group, stop the instance, modify the instance placement, and then restart the instance.

```
aws ec2 stop-instances \  
  --instance-ids i-0e6ddf6187EXAMPLE
```

```
--instance-ids i-0123a456700123456

aws ec2 modify-instance-placement \
  --instance-id i-0123a456700123456 \
  --group-name MySpreadGroup

aws ec2 start-instances \
  --instance-ids i-0123a456700123456
```

For more information, see [Changing the Placement Group for an Instance](#) in the *Amazon Elastic Compute Cloud Users Guide*.

Example 4: To remove an instance from a placement group

The following `modify-instance-placement` example removes an instance from a placement group by stopping the instance, modifying the instance placement, and then restarting the instance. The following example specifies an empty string ("") for the placement group name to indicate that the instance is not to be located in a placement group.

Stop the instance:

```
aws ec2 stop-instances \
  --instance-ids i-0123a456700123456
```

Modify the placement (Windows Command Prompt, Linux, and macOS):

```
aws ec2 modify-instance-placement \
  --instance-id i-0123a456700123456 \
  --group-name ""
```

Modify the placement (Windows PowerShell):

```
aws ec2 modify-instance-placement `
  --instance-id i-0123a456700123456 `
  --group-name ""
```

Restart the instance:

```
aws ec2 start-instances \
  --instance-ids i-0123a456700123456
```

Output:

```
{
  "Return": true
}
```

For more information, see [Modifying Instance Tenancy and Affinity](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [ModifyInstancePlacement](#) in *AWS CLI Command Reference*.

modify-ipam-pool

The following code example shows how to use `modify-ipam-pool`.

AWS CLI**To modify an IPAM pool**

The following `modify-ipam-pool` example modifies an IPAM pool.

(Linux):

```
aws ec2 modify-ipam-pool \
  --ipam-pool-id ipam-pool-0533048da7d823723 \
  --add-allocation-resource-tags "Key=Owner,Value=Build Team" \
  --clear-allocation-default-netmask-length \
  --allocation-min-netmask-length 14
```

(Windows):

```
aws ec2 modify-ipam-pool ^
  --ipam-pool-id ipam-pool-0533048da7d823723 ^
  --add-allocation-resource-tags "Key=Owner,Value=Build Team" ^
  --clear-allocation-default-netmask-length ^
  --allocation-min-netmask-length 14
```

Output:

```
{
  "IpamPool": {
```

```

    "OwnerId": "123456789012",
    "IpamPoolId": "ipam-pool-0533048da7d823723",
    "IpamPoolArn": "arn:aws:ec2::123456789012:ipam-pool/ipam-
pool-0533048da7d823723",
    "IpamScopeArn": "arn:aws:ec2::123456789012:ipam-scope/ipam-
scope-02fc38cd4c48e7d38",
    "IpamScopeType": "private",
    "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",
    "IpamRegion": "us-east-1",
    "Locale": "None",
    "PoolDepth": 1,
    "State": "modify-complete",
    "AutoImport": true,
    "AddressFamily": "ipv4",
    "AllocationMinNetmaskLength": 14,
    "AllocationMaxNetmaskLength": 26,
    "AllocationResourceTags": [
      {
        "Key": "Environment",
        "Value": "Preprod"
      },
      {
        "Key": "Owner",
        "Value": "Build Team"
      }
    ]
  }
}

```

For more information, see [Edit a pool](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [ModifyIpamPool](#) in *AWS CLI Command Reference*.

modify-ipam-resource-cidr

The following code example shows how to use `modify-ipam-resource-cidr`.

AWS CLI

To modify the CIDR allocated to a resource

The following `modify-ipam-resource-cidr` example modifies a resource CIDR.

(Linux):

```
aws ec2 modify-ipam-resource-cidr \  
  --current-ipam-scope-id ipam-scope-02fc38cd4c48e7d38 \  
  --destination-ipam-scope-id ipam-scope-0da34c61fd189a141 \  
  --resource-id vpc-010e1791024eb0af9 \  
  --resource-cidr 10.0.1.0/24 \  
  --resource-region us-east-1 \  
  --monitored
```

(Windows):

```
aws ec2 modify-ipam-resource-cidr ^  
  --current-ipam-scope-id ipam-scope-02fc38cd4c48e7d38 ^  
  --destination-ipam-scope-id ipam-scope-0da34c61fd189a141 ^  
  --resource-id vpc-010e1791024eb0af9 ^  
  --resource-cidr 10.0.1.0/24 ^  
  --resource-region us-east-1 ^  
  --monitored
```

Output:

```
{  
  "IpamResourceCidr": {  
    "IpamId": "ipam-08440e7a3acde3908",  
    "IpamScopeId": "ipam-scope-0da34c61fd189a141",  
    "IpamPoolId": "ipam-pool-0533048da7d823723",  
    "ResourceRegion": "us-east-1",  
    "ResourceOwnerId": "123456789012",  
    "ResourceId": "vpc-010e1791024eb0af9",  
    "ResourceCidr": "10.0.1.0/24",  
    "ResourceType": "vpc",  
    "ResourceTags": [  
      {  
        "Key": "Environment",  
        "Value": "Preprod"  
      },  
      {  
        "Key": "Owner",  
        "Value": "Build Team"  
      }  
    ],  
    "IpUsage": 0.0,  
    "ComplianceStatus": "noncompliant",
```



```
    "ManagementState": "managed",
    "OverlapStatus": "overlapping",
    "VpcId": "vpc-010e1791024eb0af9"
  }
}
```

For more information on moving resources, see [Move resource CIDRs between scopes](#) in the *Amazon VPC IPAM User Guide*.

For more information on changing monitoring states, see [Change the monitoring state of resource CIDRs](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [ModifyIpamResourceCidr](#) in *AWS CLI Command Reference*.

modify-ipam-resource-discovery

The following code example shows how to use `modify-ipam-resource-discovery`.

AWS CLI

To modify the operating regions of a resource discovery

In this example, you're an IPAM delegated admin who wants to modify the operating regions of a resource discovery.

To complete this request:

You cannot modify a default resource discovery and you must be the owner of the resource discovery. You need the resource discovery ID, which you can get with [describe-ipam-resource-discoveries](#).

The following `modify-ipam-resource-discovery` example modifies a non-default resource discovery in your AWS account.

```
aws ec2 modify-ipam-resource-discovery \
  --ipam-resource-discovery-id ipam-res-disco-0f4ef577a9f37a162 \
  --add-operating-regions RegionName='us-west-1' \
  --remove-operating-regions RegionName='us-east-2' \
  --region us-east-1
```

Output:

```
{
  "IpamResourceDiscovery": {
    "OwnerId": "149977607591",
    "IpamResourceDiscoveryId": "ipam-res-disco-0365d2977fc1672fe",
    "IpamResourceDiscoveryArn": "arn:aws:ec2::149977607591:ipam-resource-
discovery/ipam-res-disco-0365d2977fc1672fe",
    "IpamResourceDiscoveryRegion": "us-east-1",
    "Description": "Example",
    "OperatingRegions": [
      {
        "RegionName": "us-east-1"
      },
      {
        "RegionName": "us-west-1"
      }
    ],
    "IsDefault": false,
    "State": "modify-in-progress"
  }
}
```

For more information, see [Work with resource discoveries](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [ModifyIpamResourceDiscovery](#) in *AWS CLI Command Reference*.

modify-ipam-scope

The following code example shows how to use `modify-ipam-scope`.

AWS CLI

To modify the description of a scope

In this scenario, you're an IPAM delegated admin who wants to modify the description of an IPAM scope.

To complete this request, you'll need the scope ID, which you can get with [describe-ipam-scopes](#).

The following `modify-ipam-scope` example updates the description of the scope.

```
aws ec2 modify-ipam-scope \
  --ipam-scope-id ipam-scope-0d3539a30b57dcdd1 \
```

```
--description example \  
--region us-east-1
```

Output:

```
{  
  "IpamScope": {  
    "OwnerId": "320805250157",  
    "IpamScopeId": "ipam-scope-0d3539a30b57dcdd1",  
    "IpamScopeArn": "arn:aws:ec2::320805250157:ipam-scope/ipam-  
scope-0d3539a30b57dcdd1",  
    "IpamArn": "arn:aws:ec2::320805250157:ipam/ipam-005f921c17ebd5107",  
    "IpamRegion": "us-east-1",  
    "IpamScopeType": "public",  
    "IsDefault": true,  
    "Description": "example",  
    "PoolCount": 1,  
    "State": "modify-in-progress"  
  }  
}
```

For more information about scopes, see [How IPAM works](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [ModifyIpamScope](#) in *AWS CLI Command Reference*.

modify-ipam

The following code example shows how to use `modify-ipam`.

AWS CLI

To modify an IPAM

The following `modify-ipam` example modifies an IPAM by adding an Operating Region.

(Linux):

```
aws ec2 modify-ipam \  
  --ipam-id ipam-08440e7a3acde3908 \  
  --add-operating-regions RegionName=us-west-2
```

(Windows):

```
aws ec2 modify-ipam ^
  --ipam-id ipam-08440e7a3acde3908 ^
  --add-operating-regions RegionName=us-west-2
```

Output:

```
{
  "Ipam": {
    "OwnerId": "123456789012",
    "IpamId": "ipam-08440e7a3acde3908",
    "IpamArn": "arn:aws:ec2::123456789012:ipam/ipam-08440e7a3acde3908",
    "IpamRegion": "us-east-1",
    "PublicDefaultScopeId": "ipam-scope-0b9eed026396dbc16",
    "PrivateDefaultScopeId": "ipam-scope-02fc38cd4c48e7d38",
    "ScopeCount": 3,
    "OperatingRegions": [
      {
        "RegionName": "us-east-1"
      },
      {
        "RegionName": "us-east-2"
      },
      {
        "RegionName": "us-west-1"
      },
      {
        "RegionName": "us-west-2"
      }
    ],
    "State": "modify-in-progress"
  }
}
```

- For API details, see [ModifyIpam](#) in *AWS CLI Command Reference*.

modify-launch-template

The following code example shows how to use `modify-launch-template`.

AWS CLI

To change the default launch template version

This example specifies version 2 of the specified launch template as the default version.

Command:

```
aws ec2 modify-launch-template --launch-template-id lt-0abcd290751193123 --default-version 2
```

Output:

```
{
  "LaunchTemplate": {
    "LatestVersionNumber": 2,
    "LaunchTemplateId": "lt-0abcd290751193123",
    "LaunchTemplateName": "WebServers",
    "DefaultVersionNumber": 2,
    "CreatedBy": "arn:aws:iam::123456789012:root",
    "CreateTime": "2017-12-01T13:35:46.000Z"
  }
}
```

- For API details, see [ModifyLaunchTemplate](#) in *AWS CLI Command Reference*.

modify-managed-prefix-list

The following code example shows how to use `modify-managed-prefix-list`.

AWS CLI

To modify a prefix list

The following `modify-managed-prefix-list` example adds an entry to the specified prefix list.

```
aws ec2 modify-managed-prefix-list \
  --prefix-list-id pl-0123456abcabcabc1 \
  --add-entries Cidr=10.1.0.0/16,Description=vpc-c \
  --current-version 1
```

Output:

```
{
```

```
"PrefixList": {
  "PrefixListId": "pl-0123456abcabcabc1",
  "AddressFamily": "IPv4",
  "State": "modify-in-progress",
  "PrefixListArn": "arn:aws:ec2:us-west-2:123456789012:prefix-list/
pl-0123456abcabcabc1",
  "PrefixListName": "vpc-cidrs",
  "MaxEntries": 10,
  "Version": 1,
  "OwnerId": "123456789012"
}
```

For more information, see [Managed prefix lists](#) in the *Amazon VPC User Guide*.

- For API details, see [ModifyManagedPrefixList](#) in *AWS CLI Command Reference*.

modify-network-interface-attribute

The following code example shows how to use `modify-network-interface-attribute`.

AWS CLI

To modify the attachment attribute of a network interface

This example command modifies the attachment attribute of the specified network interface.

Command:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --
attachment AttachmentId=eni-attach-43348162,DeleteOnTermination=false
```

To modify the description attribute of a network interface

This example command modifies the description attribute of the specified network interface.

Command:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --
description "My description"
```

To modify the groupSet attribute of a network interface

This example command modifies the groupSet attribute of the specified network interface.

Command:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --groups sg-903004f8 sg-1a2b3c4d
```

To modify the sourceDestCheck attribute of a network interface

This example command modifies the sourceDestCheck attribute of the specified network interface.

Command:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --no-source-dest-check
```

- For API details, see [ModifyNetworkInterfaceAttribute](#) in *AWS CLI Command Reference*.

modify-private-dns-name-options

The following code example shows how to use modify-private-dns-name-options.

AWS CLI

To modify the options for instance hostnames

The following modify-private-dns-name-options example disables the option to respond to DNS queries for instance hostnames with DNS A records.

```
aws ec2 modify-private-dns-name-options \
  --instance-id i-1234567890abcdef0 \
  --no-enable-resource-name-dns-a-record
```

Output:

```
{
  "Return": true
}
```

For more information, see [Amazon EC2 instance hostname types](#) in the *Amazon EC2 User Guide*.

- For API details, see [ModifyPrivateDnsNameOptions](#) in *AWS CLI Command Reference*.

modify-reserved-instances

The following code example shows how to use `modify-reserved-instances`.

AWS CLI

To modify Reserved Instances

This example command moves a Reserved Instance to another Availability Zone in the same region.

Command:

```
aws ec2 modify-reserved-instances --reserved-instances-ids b847fa93-e282-4f55-
b59a-1342f5bd7c02 --target-configurations AvailabilityZone=us-west-1c,Platform=EC2-
Classic,InstanceCount=10
```

Output:

```
{
  "ReservedInstancesModificationId": "rimod-d3ed4335-b1d3-4de6-ab31-0f13aaf46687"
}
```

To modify the network platform of Reserved Instances

This example command converts EC2-Classic Reserved Instances to EC2-VPC.

Command:

```
aws ec2 modify-reserved-instances --reserved-instances-ids f127bd27-edb7-44c9-
a0eb-0d7e09259af0 --target-configurations AvailabilityZone=us-west-1c,Platform=EC2-
VPC,InstanceCount=5
```

Output:

```
{
  "ReservedInstancesModificationId": "rimod-82fa9020-668f-4fb6-945d-61537009d291"
}
```



```
}
```

For more information, see *Modifying Your Reserved Instances* in the *Amazon EC2 User Guide*.

To modify the instance size of Reserved Instances

This example command modifies a Reserved Instance that has 10 m1.small Linux/UNIX instances in us-west-1c so that 8 m1.small instances become 2 m1.large instances, and the remaining 2 m1.small become 1 m1.medium instance in the same Availability Zone. Command:

```
aws ec2 modify-reserved-instances --reserved-instances-ids
1ba8e2e3-3556-4264-949e-63ee671405a9 --target-configurations AvailabilityZone=us-
west-1c,Platform=EC2-Classic,InstanceCount=2,InstanceType=m1.large
AvailabilityZone=us-west-1c,Platform=EC2-
Classic,InstanceCount=1,InstanceType=m1.medium
```

Output:

```
{
  "ReservedInstancesModificationId": "rimod-acc5f240-080d-4717-b3e3-1c6b11fa00b6"
}
```

For more information, see *Modifying the Instance Size of Your Reservations* in the *Amazon EC2 User Guide*.

- For API details, see [ModifyReservedInstances](#) in *AWS CLI Command Reference*.

modify-security-group-rules

The following code example shows how to use `modify-security-group-rules`.

AWS CLI

To modify a security group rules to update the rule description, the IP protocol, and the CidrIpv4 address range

The following `modify-security-group-rules` example updates the description, the IP protocol, and the IPV4 CIDR range of a specified security group rule. Use the `security-group-rules` parameter to enter the updates for the specified security group rules. `-1` specifies all protocols.

```
aws ec2 modify-security-group-rules \  
  --group-id sg-1234567890abcdef0 \  
  --security-group-rules SecurityGroupId=sg-  
abcdef01234567890,SecurityGroupRule='{Description=test,IpProtocol=-1,CidrIpv4=0.0.0.0/0}'
```

Output:

```
{  
  "Return": true  
}
```

For more information about security group rules, see [Security group rules](#) in the *Amazon EC2 User Guide*.

- For API details, see [ModifySecurityGroupRules](#) in *AWS CLI Command Reference*.

modify-snapshot-attribute

The following code example shows how to use `modify-snapshot-attribute`.

AWS CLI

Example 1: To modify a snapshot attribute

The following `modify-snapshot-attribute` example updates the `createVolumePermission` attribute for the specified snapshot, removing volume permissions for the specified user.

```
aws ec2 modify-snapshot-attribute \  
  --snapshot-id snap-1234567890abcdef0 \  
  --attribute createVolumePermission \  
  --operation-type remove \  
  --user-ids 123456789012
```

Example 2: To make a snapshot public

The following `modify-snapshot-attribute` example makes the specified snapshot public.

```
aws ec2 modify-snapshot-attribute \  
  --snapshot-id snap-1234567890abcdef0 \  
  --attribute createVolumePermission \  
  --operation-type public
```

```
--operation-type add \  
--group-names all
```

- For API details, see [ModifySnapshotAttribute](#) in *AWS CLI Command Reference*.

modify-snapshot-tier

The following code example shows how to use `modify-snapshot-tier`.

AWS CLI

Example 1: To archive a snapshot

The following `modify-snapshot-tier` example archives the specified snapshot.

```
aws ec2 modify-snapshot-tier \  
  --snapshot-id snap-01234567890abcdef \  
  --storage-tier archive
```

Output:

```
{  
  "SnapshotId": "snap-01234567890abcdef",  
  "TieringStartTime": "2021-09-15T16:44:37.574Z"  
}
```

The `TieringStartTime` response parameter indicates the date and time at which the archive process was started, in UTC time format (YYYY-MM-DDTHH:MM:SSZ).

For more information about snapshot archiving, see [Archive Amazon EBS snapshots](#) in the *Amazon EC2 User Guide*.

- For API details, see [ModifySnapshotTier](#) in *AWS CLI Command Reference*.

modify-spot-fleet-request

The following code example shows how to use `modify-spot-fleet-request`.

AWS CLI

To modify a Spot fleet request

This example command updates the target capacity of the specified Spot fleet request.

Command:

```
aws ec2 modify-spot-fleet-request --target-capacity 20 --spot-fleet-request-id
sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

Output:

```
{
  "Return": true
}
```

This example command decreases the target capacity of the specified Spot fleet request without terminating any Spot Instances as a result.

Command:

```
aws ec2 modify-spot-fleet-request --target-capacity 10 --excess-capacity-
termination-policy NoTermination --spot-fleet-request-ids sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE
```

Output:

```
{
  "Return": true
}
```

- For API details, see [ModifySpotFleetRequest](#) in *AWS CLI Command Reference*.

modify-subnet-attribute

The following code example shows how to use `modify-subnet-attribute`.

AWS CLI

To change a subnet's public IPv4 addressing behavior

This example modifies `subnet-1a2b3c4d` to specify that all instances launched into this subnet are assigned a public IPv4 address. If the command succeeds, no output is returned.

Command:

```
aws ec2 modify-subnet-attribute --subnet-id subnet-1a2b3c4d --map-public-ip-on-launch
```

To change a subnet's IPv6 addressing behavior

This example modifies subnet-1a2b3c4d to specify that all instances launched into this subnet are assigned an IPv6 address from the range of the subnet.

Command:

```
aws ec2 modify-subnet-attribute --subnet-id subnet-1a2b3c4d --assign-ipv6-address-on-creation
```

For more information, see IP Addressing in Your VPC in the *AWS Virtual Private Cloud User Guide*.

- For API details, see [ModifySubnetAttribute](#) in *AWS CLI Command Reference*.

modify-traffic-mirror-filter-network-services

The following code example shows how to use `modify-traffic-mirror-filter-network-services`.

AWS CLI**To add network services to a Traffic Mirror filter**

The following `modify-traffic-mirror-filter-network-services` example adds the Amazon DNS network services to the specified filter.

```
aws ec2 modify-traffic-mirror-filter-network-services \
  --traffic-mirror-filter-id tmf-04812ff784EXAMPLE \
  --add-network-service amazon-dns
```

Output:

```
{
  "TrafficMirrorFilter": {
    "Tags": [
```

```

        {
            "Key": "Name",
            "Value": "Production"
        }
    ],
    "EgressFilterRules": [],
    "NetworkServices": [
        "amazon-dns"
    ],
    "TrafficMirrorFilterId": "tmf-04812ff784EXAMPLE",
    "IngressFilterRules": [
        {
            "SourceCidrBlock": "0.0.0.0/0",
            "RuleNumber": 1,
            "DestinationCidrBlock": "0.0.0.0/0",
            "Description": "TCP Rule",
            "Protocol": 6,
            "TrafficDirection": "ingress",
            "TrafficMirrorFilterId": "tmf-04812ff784EXAMPLE",
            "RuleAction": "accept",
            "TrafficMirrorFilterRuleId": "tmf-04812ff784EXAMPLE"
        }
    ]
}
}
}

```

For more information, see [Modify Traffic Mirror Filter Network Services](#) in the *AWS Traffic Mirroring Guide*.

- For API details, see [ModifyTrafficMirrorFilterNetworkServices](#) in *AWS CLI Command Reference*.

modify-traffic-mirror-filter-rule

The following code example shows how to use `modify-traffic-mirror-filter-rule`.

AWS CLI

To modify a traffic mirror filter rule

The following `modify-traffic-mirror-filter-rule` example modifies the description of the specified traffic mirror filter rule.

```
aws ec2 modify-traffic-mirror-filter-rule \
```

```
--traffic-mirror-filter-rule-id tmfr-0ca76e0e08EXAMPLE \  
--description "TCP Rule"
```

Output:

```
{  
  "TrafficMirrorFilterRule": {  
    "TrafficMirrorFilterRuleId": "tmfr-0ca76e0e08EXAMPLE",  
    "TrafficMirrorFilterId": "tmf-0293f26e86EXAMPLE",  
    "TrafficDirection": "ingress",  
    "RuleNumber": 100,  
    "RuleAction": "accept",  
    "Protocol": 6,  
    "DestinationCidrBlock": "10.0.0.0/24",  
    "SourceCidrBlock": "10.0.0.0/24",  
    "Description": "TCP Rule"  
  }  
}
```

For more information, see [Modify Your Traffic Mirror Filter Rules](#) in the *AWS Traffic Mirroring Guide*.

- For API details, see [ModifyTrafficMirrorFilterRule](#) in *AWS CLI Command Reference*.

modify-traffic-mirror-session

The following code example shows how to use `modify-traffic-mirror-session`.

AWS CLI

To modify a traffic mirror session

The following `modify-traffic-mirror-session` example changes the traffic mirror session description and the number of packets to mirror.

```
aws ec2 modify-traffic-mirror-session \  
  --description "Change packet length" \  
  --traffic-mirror-session-id tms-08a33b1214EXAMPLE \  
  --remove-fields "packet-length"
```

Output:

```
{
  "TrafficMirrorSession": {
    "TrafficMirrorSessionId": "tms-08a33b1214EXAMPLE",
    "TrafficMirrorTargetId": "tmt-07f75d8feeEXAMPLE",
    "TrafficMirrorFilterId": "tmf-04812ff784EXAMPLE",
    "NetworkInterfaceId": "eni-070203f901EXAMPLE",
    "OwnerId": "111122223333",
    "SessionNumber": 1,
    "VirtualNetworkId": 7159709,
    "Description": "Change packet length",
    "Tags": []
  }
}
```

For more information, see [Modify your traffic mirror session](#) in the *Traffic Mirroring Guide*.

- For API details, see [ModifyTrafficMirrorSession](#) in *AWS CLI Command Reference*.

modify-transit-gateway-prefix-list-reference

The following code example shows how to use `modify-transit-gateway-prefix-list-reference`.

AWS CLI

To modify a reference to a prefix list

The following `modify-transit-gateway-prefix-list-reference` example modifies the prefix list reference in the specified route table by changing the attachment to which traffic is routed.

```
aws ec2 modify-transit-gateway-prefix-list-reference \
  --transit-gateway-route-table-id tgw-rtb-0123456789abcd123 \
  --prefix-list-id pl-11111122222222333 \
  --transit-gateway-attachment-id tgw-attach-aabbccddaabbccaab
```

Output:

```
{
  "TransitGatewayPrefixListReference": {
    "TransitGatewayRouteTableId": "tgw-rtb-0123456789abcd123",
```



```

    "PrefixListId": "pl-11111122222222333",
    "PrefixListOwnerId": "123456789012",
    "State": "modifying",
    "Blackhole": false,
    "TransitGatewayAttachment": {
      "TransitGatewayAttachmentId": "tgw-attach-aabbccddaabbccaab",
      "ResourceType": "vpc",
      "ResourceId": "vpc-112233445566aabbcc"
    }
  }
}

```

For more information, see [Prefix list references](#) in the *Transit Gateways Guide*.

- For API details, see [ModifyTransitGatewayPrefixListReference](#) in *AWS CLI Command Reference*.

modify-transit-gateway-vpc-attachment

The following code example shows how to use `modify-transit-gateway-vpc-attachment`.

AWS CLI

To modify a transit gateway VPC attachment

The following `modify-transit-gateway-vpc-attachment` example adds a subnet to the specified transit gateway VPC attachment.

```

aws ec2 modify-transit-gateway-vpc-attachment \
  --transit-gateway-attachment-id tgw-attach-09fbd47ddfEXAMPLE \
  --add-subnet-ids subnet-0e51f45802EXAMPLE

```

Output:

```

{
  "TransitGatewayVpcAttachment": {
    "TransitGatewayAttachmentId": "tgw-attach-09fbd47ddfEXAMPLE",
    "TransitGatewayId": "tgw-0560315ccfEXAMPLE",
    "VpcId": "vpc-5eccc927",
    "VpcOwnerId": "111122223333",
    "State": "modifying",
    "SubnetIds": [
      "subnet-0e51f45802EXAMPLE",
    ]
  }
}

```

```

        "subnet-1EXAMPLE"
    ],
    "CreationTime": "2019-08-08T16:47:38.000Z",
    "Options": {
        "DnsSupport": "enable",
        "Ipv6Support": "disable"
    }
}
}

```

For more information, see [Transit gateway attachments to a VPC](#) in the *Transit Gateways Guide*.

- For API details, see [ModifyTransitGatewayVpcAttachment](#) in *AWS CLI Command Reference*.

modify-transit-gateway

The following code example shows how to use `modify-transit-gateway`.

AWS CLI

To modify a transit gateway

The following `modify-transit-gateway` example modifies the specified transit gateway by enabling ECMP support for VPN attachments.

```

aws ec2 modify-transit-gateway \
  --transit-gateway-id tgw-111111222222aaaaa \
  --options VpnEcmpSupport=enable

```

Output:

```

{
  "TransitGateway": {
    "TransitGatewayId": "tgw-111111222222aaaaa",
    "TransitGatewayArn": "64512",
    "State": "modifying",
    "OwnerId": "123456789012",
    "CreationTime": "2020-04-30T08:41:37.000Z",
    "Options": {
      "AmazonSideAsn": 64512,
      "AutoAcceptSharedAttachments": "disable",
      "DefaultRouteTableAssociation": "enable",

```

```

        "AssociationDefaultRouteTableId": "tgw-rtb-0123456789abcd123",
        "DefaultRouteTablePropagation": "enable",
        "PropagationDefaultRouteTableId": "tgw-rtb-0123456789abcd123",
        "VpnEcmpSupport": "enable",
        "DnsSupport": "enable"
    }
}
}

```

For more information, see [Transit gateways](#) in the *Transit Gateways Guide*.

- For API details, see [ModifyTransitGateway](#) in *AWS CLI Command Reference*.

modify-verified-access-endpoint-policy

The following code example shows how to use `modify-verified-access-endpoint-policy`.

AWS CLI

To configure the Verified Access policy for an endpoint

The following `modify-verified-access-endpoint-policy` example adds the specified Verified Access policy to the specified Verified Access endpoint.

```

aws ec2 modify-verified-access-endpoint-policy \
  --verified-access-endpoint-id vae-066fac616d4d546f2 \
  --policy-enabled \
  --policy-document file://policy.txt

```

Contents of `policy.txt`:

```

permit(principal,action,resource)
when {
  context.identity.groups.contains("finance") &&
  context.identity.email.verified == true
};

```

Output:

```

{
  "PolicyEnabled": true,

```

```
"PolicyDocument": "permit(principal,action,resource)\nwhen
{\n  context.identity.groups.contains(\"finance\") &&\n
context.identity.email_verified == true\n};"
}
```

For more information, see [Verified Access policies](#) in the *AWS Verified Access User Guide*.

- For API details, see [ModifyVerifiedAccessEndpointPolicy](#) in *AWS CLI Command Reference*.

modify-verified-access-endpoint

The following code example shows how to use `modify-verified-access-endpoint`.

AWS CLI

To modify the configuration of a Verified Access endpoint

The following `modify-verified-access-endpoint` example adds the specified description to the specified Verified Access endpoint.

```
aws ec2 modify-verified-access-endpoint \
  --verified-access-endpoint-id vae-066fac616d4d546f2 \
  --description "Testing Verified Access"
```

Output:

```
{
  "VerifiedAccessEndpoint": {
    "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
    "VerifiedAccessGroupId": "vagr-0dbe967baf14b7235",
    "VerifiedAccessEndpointId": "vae-066fac616d4d546f2",
    "ApplicationDomain": "example.com",
    "EndpointType": "network-interface",
    "AttachmentType": "vpc",
    "DomainCertificateArn": "arn:aws:acm:us-east-2:123456789012:certificate/
eb065ea0-26f9-4e75-a6ce-0a1a7EXAMPLE",
    "EndpointDomain": "my-ava-
app.edge-00c3372d53b1540bb.vai-0ce000c0b7643abea.prod.verified-access.us-
east-2.amazonaws.com",
    "SecurityGroupIds": [
      "sg-004915970c4c8f13a"
    ],
  },
}
```

```
    "NetworkInterfaceOptions": {
      "NetworkInterfaceId": "eni-0aec70418c8d87a0f",
      "Protocol": "https",
      "Port": 443
    },
    "Status": {
      "Code": "updating"
    },
    "Description": "Testing Verified Access",
    "CreationTime": "2023-08-25T20:54:43",
    "LastUpdatedTime": "2023-08-25T22:46:32"
  }
}
```

For more information, see [Verified Access endpoints](#) in the *AWS Verified Access User Guide*.

- For API details, see [ModifyVerifiedAccessEndpoint](#) in *AWS CLI Command Reference*.

modify-verified-access-group-policy

The following code example shows how to use `modify-verified-access-group-policy`.

AWS CLI

To configure a Verified Access policy for a group

The following `modify-verified-access-group-policy` example adds the specified Verified Access policy to the specified Verified Access group.

```
aws ec2 modify-verified-access-group-policy \
  --verified-access-group-id vagr-0dbe967baf14b7235 \
  --policy-enabled \
  --policy-document file://policy.txt
```

Contents of `policy.txt`:

```
permit(principal,action,resource)
when {
  context.identity.groups.contains("finance") &&
  context.identity.email.verified == true
};
```

Output:

```
{
  "PolicyEnabled": true,
  "PolicyDocument": "permit(principal,action,resource)\nwhen
{\n  context.identity.groups.contains(\"finance\") &&\n
context.identity.email_verified == true\n};"
}
```

For more information, see [Verified Access groups](#) in the *AWS Verified Access User Guide*.

- For API details, see [ModifyVerifiedAccessGroupPolicy](#) in *AWS CLI Command Reference*.

modify-verified-access-group

The following code example shows how to use `modify-verified-access-group`.

AWS CLI**To modify the configuration of a Verified Access group**

The following `modify-verified-access-group` example adds the specified description to the specified Verified Access group.

```
aws ec2 modify-verified-access-group \
  --verified-access-group-id vagr-0dbe967baf14b7235 \
  --description "Testing Verified Access"
```

Output:

```
{
  "VerifiedAccessGroup": {
    "VerifiedAccessGroupId": "vagr-0dbe967baf14b7235",
    "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",
    "Description": "Testing Verified Access",
    "Owner": "123456789012",
    "VerifiedAccessGroupArn": "arn:aws:ec2:us-east-2:123456789012:verified-
access-group/vagr-0dbe967baf14b7235",
    "CreationTime": "2023-08-25T19:55:19",
    "LastUpdatedTime": "2023-08-25T22:17:25"
  }
}
```

For more information, see [Verified Access groups](#) in the *AWS Verified Access User Guide*.

- For API details, see [ModifyVerifiedAccessGroup](#) in *AWS CLI Command Reference*.

modify-verified-access-instance-logging-configuration

The following code example shows how to use `modify-verified-access-instance-logging-configuration`.

AWS CLI

To enable logging for a Verified Access instance

The following `modify-verified-access-instance-logging-configuration` example enables access logging for the specified Verified Access instance. The logs will be delivered to the specified CloudWatch Logs log group.

```
aws ec2 modify-verified-access-instance-logging-configuration \  
  --verified-access-instance-id vai-0ce000c0b7643abea \  
  --access-logs CloudWatchLogs={Enabled=true,LogGroup=my-log-group}
```

Output:

```
{  
  "LoggingConfiguration": {  
    "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",  
    "AccessLogs": {  
      "S3": {  
        "Enabled": false  
      },  
      "CloudWatchLogs": {  
        "Enabled": true,  
        "DeliveryStatus": {  
          "Code": "success"  
        },  
        "LogGroup": "my-log-group"  
      },  
      "KinesisDataFirehose": {  
        "Enabled": false  
      },  
      "LogVersion": "ocsf-1.0.0-rc.2",  
      "IncludeTrustContext": false  
    }  
  }  
}
```

```
    }  
  }  
}
```

For more information, see [Verified Access logs](#) in the *AWS Verified Access User Guide*.

- For API details, see [ModifyVerifiedAccessInstanceLoggingConfiguration](#) in *AWS CLI Command Reference*.

modify-verified-access-instance

The following code example shows how to use `modify-verified-access-instance`.

AWS CLI

To modify the configuration of a Verified Access instance

The following `modify-verified-access-instance` example adds the specified description to the specified Verified Access instance.

```
aws ec2 modify-verified-access-instance \  
  --verified-access-instance-id vai-0ce000c0b7643abea \  
  --description "Testing Verified Access"
```

Output:

```
{  
  "VerifiedAccessInstance": {  
    "VerifiedAccessInstanceId": "vai-0ce000c0b7643abea",  
    "Description": "Testing Verified Access",  
    "VerifiedAccessTrustProviders": [  
      {  
        "VerifiedAccessTrustProviderId": "vatp-0bb32de759a3e19e7",  
        "TrustProviderType": "user",  
        "UserTrustProviderType": "iam-identity-center"  
      }  
    ],  
    "CreationTime": "2023-08-25T18:27:56",  
    "LastUpdatedTime": "2023-08-25T22:41:04"  
  }  
}
```


For more information, see [Verified Access instances](#) in the *AWS Verified Access User Guide*.

- For API details, see [ModifyVerifiedAccessInstance](#) in *AWS CLI Command Reference*.

modify-verified-access-trust-provider

The following code example shows how to use `modify-verified-access-trust-provider`.

AWS CLI

To modify the configuration of a Verified Access trust provider

The following `modify-verified-access-trust-provider` example adds the specified description to the specified Verified Access trust provider.

```
aws ec2 modify-verified-access-trust-provider \  
  --verified-access-trust-provider-id vatp-0bb32de759a3e19e7 \  
  --description "Testing Verified Access"
```

Output:

```
{  
  "VerifiedAccessTrustProvider": {  
    "VerifiedAccessTrustProviderId": "vatp-0bb32de759a3e19e7",  
    "Description": "Testing Verified Access",  
    "TrustProviderType": "user",  
    "UserTrustProviderType": "iam-identity-center",  
    "PolicyReferenceName": "idc",  
    "CreationTime": "2023-08-25T19:00:38",  
    "LastUpdatedTime": "2023-08-25T19:18:21"  
  }  
}
```

For more information, see [Trust providers for Verified Access](#) in the *AWS Verified Access User Guide*.

- For API details, see [ModifyVerifiedAccessTrustProvider](#) in *AWS CLI Command Reference*.

modify-volume-attribute

The following code example shows how to use `modify-volume-attribute`.

AWS CLI

To modify a volume attribute

This example sets the `autoEnableIo` attribute of the volume with the ID `vol-1234567890abcdef0` to `true`. If the command succeeds, no output is returned.

Command:

```
aws ec2 modify-volume-attribute --volume-id vol-1234567890abcdef0 --auto-enable-io
```

- For API details, see [ModifyVolumeAttribute](#) in *AWS CLI Command Reference*.

modify-volume

The following code example shows how to use `modify-volume`.

AWS CLI

Example 1: To modify a volume by changing its size

The following `modify-volume` example changes the size of the specified volume to 150GB.

Command:

```
aws ec2 modify-volume --size 150 --volume-id vol-1234567890abcdef0
```

Output:

```
{
  "VolumeModification": {
    "TargetSize": 150,
    "TargetVolumeType": "io1",
    "ModificationState": "modifying",
    "VolumeId": " vol-1234567890abcdef0",
    "TargetIops": 100,
    "StartTime": "2019-05-17T11:27:19.000Z",
    "Progress": 0,
    "OriginalVolumeType": "io1",
    "OriginalIops": 100,
    "OriginalSize": 100
  }
}
```

```
}
```

Example 2: To modify a volume by changing its type, size, and IOPS value

The following `modify-volume` example changes the volume type to Provisioned IOPS SSD, sets the target IOPS rate to 10000, and sets the volume size to 350GB.

```
aws ec2 modify-volume \  
  --volume-type io1 \  
  --iops 10000 \  
  --size 350 \  
  --volume-id vol-1234567890abcdef0
```

Output:

```
{  
  "VolumeModification": {  
    "TargetSize": 350,  
    "TargetVolumeType": "io1",  
    "ModificationState": "modifying",  
    "VolumeId": "vol-0721c1a9d08c93bf6",  
    "TargetIops": 10000,  
    "StartTime": "2019-05-17T11:38:57.000Z",  
    "Progress": 0,  
    "OriginalVolumeType": "gp2",  
    "OriginalIops": 150,  
    "OriginalSize": 50  
  }  
}
```

- For API details, see [ModifyVolume](#) in *AWS CLI Command Reference*.

modify-vpc-attribute

The following code example shows how to use `modify-vpc-attribute`.

AWS CLI

To modify the `enableDnsSupport` attribute

This example modifies the `enableDnsSupport` attribute. This attribute indicates whether DNS resolution is enabled for the VPC. If this attribute is `true`, the Amazon DNS server resolves DNS

hostnames for your instances to their corresponding IP addresses; otherwise, it does not. If the command succeeds, no output is returned.

Command:

```
aws ec2 modify-vpc-attribute --vpc-id vpc-a01106c2 --enable-dns-support "{\"Value\n\":false}"
```

To modify the `enableDnsHostnames` attribute

This example modifies the `enableDnsHostnames` attribute. This attribute indicates whether instances launched in the VPC get DNS hostnames. If this attribute is `true`, instances in the VPC get DNS hostnames; otherwise, they do not. If the command succeeds, no output is returned.

Command:

```
aws ec2 modify-vpc-attribute --vpc-id vpc-a01106c2 --enable-dns-hostnames "{\"Value\n\":false}"
```

- For API details, see [ModifyVpcAttribute](#) in *AWS CLI Command Reference*.

modify-vpc-endpoint-connection-notification

The following code example shows how to use `modify-vpc-endpoint-connection-notification`.

AWS CLI

To modify an endpoint connection notification

This example changes the SNS topic for the specified endpoint connection notification.

Command:

```
aws ec2 modify-vpc-endpoint-connection-notification --connection-notification-id vpce-nfn-008776de7e03f5abc --connection-events Accept Reject --connection-notification-arn arn:aws:sns:us-east-2:123456789012:mytopic
```

Output:

```
{
  "ReturnValue": true
}
```

- For API details, see [ModifyVpcEndpointConnectionNotification](#) in *AWS CLI Command Reference*.

modify-vpc-endpoint-service-configuration

The following code example shows how to use `modify-vpc-endpoint-service-configuration`.

AWS CLI

To modify an endpoint service configuration

This example changes the acceptance requirement for the specified endpoint service.

Command:

```
aws ec2 modify-vpc-endpoint-service-configuration --service-id vpce-
svc-09222513e6e77dc86 --no-acceptance-required
```

Output:

```
{
  "ReturnValue": true
}
```

- For API details, see [ModifyVpcEndpointServiceConfiguration](#) in *AWS CLI Command Reference*.

modify-vpc-endpoint-service-payer-responsibility

The following code example shows how to use `modify-vpc-endpoint-service-payer-responsibility`.

AWS CLI

To modify the payer responsibility

The following `modify-vpc-endpoint-service-payer-responsibility` example modifies the payer responsibility of the specified endpoint service.

```
aws ec2 modify-vpc-endpoint-service-payer-responsibility \  
  --service-id vpce-svc-071afff70666e61e0 \  
  --payer-responsibility ServiceOwner
```

This command produces no output.

- For API details, see [ModifyVpcEndpointServicePayerResponsibility](#) in *AWS CLI Command Reference*.

modify-vpc-endpoint-service-permissions

The following code example shows how to use `modify-vpc-endpoint-service-permissions`.

AWS CLI

To modify endpoint service permissions

This example adds permission for an AWS account to connect to the specified endpoint service.

Command:

```
aws ec2 modify-vpc-endpoint-service-permissions --service-id vpce-  
svc-03d5ebb7d9579a2b3 --add-allowed-principals '["arn:aws:iam::123456789012:root"]'
```

Output:

```
{  
  "ReturnValue": true  
}
```

This example adds permission for a specific IAM user (admin) to connect to the specified endpoint service.

Command:

```
aws ec2 modify-vpc-endpoint-service-permissions --service-id vpce-  
svc-03d5ebb7d9579a2b3 --add-allowed-principals '["arn:aws:iam::123456789012:user/  
admin"]'
```

- For API details, see [ModifyVpcEndpointServicePermissions](#) in *AWS CLI Command Reference*.

modify-vpc-endpoint

The following code example shows how to use `modify-vpc-endpoint`.

AWS CLI

To modify a gateway endpoint

This example modifies gateway endpoint `vpce-1a2b3c4d` by associating route table `rtb-aaa222bb` with the endpoint, and resetting the policy document.

Command:

```
aws ec2 modify-vpc-endpoint --vpc-endpoint-id vpce-1a2b3c4d --add-route-table-ids
rtb-aaa222bb --reset-policy
```

Output:

```
{
  "Return": true
}
```

To modify an interface endpoint

This example modifies interface endpoint `vpce-0fe5b17a0707d6fa5` by adding subnet `subnet-d6fcaa8d` to the endpoint.

Command:

```
aws ec2 modify-vpc-endpoint --vpc-endpoint-id vpce-0fe5b17a0707d6fa5 --add-subnet-id
subnet-d6fcaa8d
```

Output:

```
{
  "Return": true
}
```

- For API details, see [ModifyVpcEndpoint](#) in *AWS CLI Command Reference*.

modify-vpc-peering-connection-options

The following code example shows how to use `modify-vpc-peering-connection-options`.

AWS CLI

To enable communication over a VPC peering connection from your local ClassicLink connection

In this example, for peering connection `pcx-aaaabbbb`, the owner of the requester VPC modifies the VPC peering connection options to enable a local ClassicLink connection to communicate with the peer VPC.

Command:

```
aws ec2 modify-vpc-peering-connection-options --vpc-peering-connection-id pcx-aaaabbbb --requester-peering-connection-options AllowEgressFromLocalClassicLinkToRemoteVpc=true
```

Output:

```
{
  "RequesterPeeringConnectionOptions": {
    "AllowEgressFromLocalClassicLinkToRemoteVpc": true
  }
}
```

To enable communication over a VPC peering connection from your local VPC to a remote ClassicLink connection

In this example, the owner of the accepter VPC modifies the VPC peering connection options to enable the local VPC to communicate with the ClassicLink connection in the peer VPC.

Command:

```
aws ec2 modify-vpc-peering-connection-options --vpc-peering-connection-id pcx-aaaabbbb --accepter-peering-connection-options AllowEgressFromLocalVpcToRemoteClassicLink=true
```

Output:


```
{
  "AccepterPeeringConnectionOptions": {
    "AllowEgressFromLocalVpcToRemoteClassicLink": true
  }
}
```

To enable DNS resolution support for the VPC peering connection

In this example, the owner of the requester VPC modifies the VPC peering connection options for `pcx-aaaabbbb` to enable the local VPC to resolve public DNS hostnames to private IP addresses when queried from instances in the peer VPC.

Command:

```
aws ec2 modify-vpc-peering-connection-options --vpc-peering-connection-id pcx-aaaabbbb --requester-peering-connection-options AllowDnsResolutionFromRemoteVpc=true
```

Output:

```
{
  "RequesterPeeringConnectionOptions": {
    "AllowDnsResolutionFromRemoteVpc": true
  }
}
```

- For API details, see [ModifyVpcPeeringConnectionOptions](#) in *AWS CLI Command Reference*.

modify-vpc-tenancy

The following code example shows how to use `modify-vpc-tenancy`.

AWS CLI

To modify the tenancy of a VPC

This example modifies the tenancy of VPC `vpc-1a2b3c4d` to `default`.

Command:

```
aws ec2 modify-vpc-tenancy --vpc-id vpc-1a2b3c4d --instance-tenancy default
```

Output:

```
{
  "Return": true
}
```

- For API details, see [ModifyVpnTenancy](#) in *AWS CLI Command Reference*.

modify-vpn-connection-options

The following code example shows how to use `modify-vpn-connection-options`.

AWS CLI**To modify your VPN connection options**

The following `modify-vpn-connection-options` example modifies the local IPv4 CIDR on the customer gateway side of the specified VPN connection.

```
aws ec2 modify-vpn-connection-options \
  --vpn-connection-id vpn-1122334455aabbccd \
  --local-ipv4-network-cidr 10.0.0.0/16
```

Output:

```
{
  "VpnConnections": [
    {
      "CustomerGatewayConfiguration": "...configuration information...",
      "CustomerGatewayId": "cgw-01234567abcde1234",
      "Category": "VPN",
      "State": "modifying",
      "Type": "ipsec.1",
      "VpnConnectionId": "vpn-1122334455aabbccd",
      "TransitGatewayId": "tgw-00112233445566aab",
      "Options": {
        "EnableAcceleration": false,
        "StaticRoutesOnly": true,
        "LocalIpv4NetworkCidr": "10.0.0.0/16",
        "RemoteIpv4NetworkCidr": "0.0.0.0/0",
        "TunnelInsideIpVersion": "ipv4"
      }
    }
  ]
}
```

```

    },
    "Routes": [],
    "Tags": [
      {
        "Key": "Name",
        "Value": "CanadaVPN"
      }
    ],
    "VgwTelemetry": [
      {
        "AcceptedRouteCount": 0,
        "LastStatusChange": "2020-07-29T10:35:11.000Z",
        "OutsideIpAddress": "203.0.113.3",
        "Status": "DOWN",
        "StatusMessage": ""
      },
      {
        "AcceptedRouteCount": 0,
        "LastStatusChange": "2020-09-02T09:09:33.000Z",
        "OutsideIpAddress": "203.0.113.5",
        "Status": "UP",
        "StatusMessage": ""
      }
    ]
  }
]
}

```

For more information, see [Modifying Site-to-Site VPN connection options](#) in the *AWS Site-to-Site VPN User Guide*.

- For API details, see [ModifyVpnConnectionOptions](#) in *AWS CLI Command Reference*.

modify-vpn-connection

The following code example shows how to use `modify-vpn-connection`.

AWS CLI

To modify a VPN connection

The following `modify-vpn-connection` example changes the target gateway for VPN connection `vpn-12345678901234567` to virtual private gateway `vgw-11223344556677889`:

```
aws ec2 modify-vpn-connection \  
  --vpn-connection-id vpn-12345678901234567 \  
  --vpn-gateway-id vgw-11223344556677889
```

Output:

```
{  
  "VpnConnection": {  
    "CustomerGatewayConfiguration": "...configuration information...",  
    "CustomerGatewayId": "cgw-aabbccdde1122334",  
    "Category": "VPN",  
    "State": "modifying",  
    "Type": "ipsec.1",  
    "VpnConnectionId": "vpn-12345678901234567",  
    "VpnGatewayId": "vgw-11223344556677889",  
    "Options": {  
      "StaticRoutesOnly": false  
    },  
    "VgwTelemetry": [  
      {  
        "AcceptedRouteCount": 0,  
        "LastStatusChange": "2019-07-17T07:34:00.000Z",  
        "OutsideIpAddress": "18.210.3.222",  
        "Status": "DOWN",  
        "StatusMessage": "IPSEC IS DOWN"  
      },  
      {  
        "AcceptedRouteCount": 0,  
        "LastStatusChange": "2019-07-20T21:20:16.000Z",  
        "OutsideIpAddress": "34.193.129.33",  
        "Status": "DOWN",  
        "StatusMessage": "IPSEC IS DOWN"  
      }  
    ]  
  }  
}
```

- For API details, see [ModifyVpnConnection](#) in *AWS CLI Command Reference*.

modify-vpn-tunnel-certificate

The following code example shows how to use `modify-vpn-tunnel-certificate`.

AWS CLI

To rotate a VPN tunnel certificate

The following `modify-vpn-tunnel-certificate` example rotates the certificate for the specified tunnel for a VPN connection

```
aws ec2 modify-vpn-tunnel-certificate \  
  --vpn-tunnel-outside-ip-address 203.0.113.17 \  
  --vpn-connection-id vpn-12345678901234567
```

Output:

```
{  
  "VpnConnection": {  
    "CustomerGatewayConfiguration": "...configuration information...",  
    "CustomerGatewayId": "cgw-aabbccdde1122334",  
    "Category": "VPN",  
    "State": "modifying",  
    "Type": "ipsec.1",  
    "VpnConnectionId": "vpn-12345678901234567",  
    "VpnGatewayId": "vgw-11223344556677889",  
    "Options": {  
      "StaticRoutesOnly": false  
    },  
    "VgwTelemetry": [  
      {  
        "AcceptedRouteCount": 0,  
        "LastStatusChange": "2019-09-11T17:27:14.000Z",  
        "OutsideIpAddress": "203.0.113.17",  
        "Status": "DOWN",  
        "StatusMessage": "IPSEC IS DOWN",  
        "CertificateArn": "arn:aws:acm:us-east-1:123456789101:certificate/  
c544d8ce-20b8-4fff-98b0-example"  
      },  
      {  
        "AcceptedRouteCount": 0,  
        "LastStatusChange": "2019-09-11T17:26:47.000Z",  
        "OutsideIpAddress": "203.0.114.18",  
        "Status": "DOWN",  
        "StatusMessage": "IPSEC IS DOWN",  
        "CertificateArn": "arn:aws:acm:us-  
east-1:123456789101:certificate/5ab64566-761b-4ad3-b259-example"      }  
    ]  
  }  
}
```

```

    }
  ]
}
}

```

- For API details, see [ModifyVpnTunnelCertificate](#) in *AWS CLI Command Reference*.

modify-vpn-tunnel-options

The following code example shows how to use `modify-vpn-tunnel-options`.

AWS CLI

To modify the tunnel options for a VPN connection

The following `modify-vpn-tunnel-options` example updates the Diffie-Hellman groups that are permitted for the specified tunnel and VPN connection.

```

aws ec2 modify-vpn-tunnel-options \
  --vpn-connection-id vpn-12345678901234567 \
  --vpn-tunnel-outside-ip-address 203.0.113.17 \
  --tunnel-options Phase1DHGroupNumbers=[{Value=14},{Value=15},{Value=16},
{Value=17},{Value=18}],Phase2DHGroupNumbers=[{Value=14},{Value=15},{Value=16},
{Value=17},{Value=18}]

```

Output:

```

{
  "VpnConnection": {
    "CustomerGatewayConfiguration": "...configuration information...",
    "CustomerGatewayId": "cgw-aabbccdde1122334",
    "Category": "VPN",
    "State": "available",
    "Type": "ipsec.1",
    "VpnConnectionId": "vpn-12345678901234567",
    "VpnGatewayId": "vgw-11223344556677889",
    "Options": {
      "StaticRoutesOnly": false,
      "TunnelOptions": [
        {
          "OutsideIpAddress": "203.0.113.17",
          "Phase1DHGroupNumbers": [

```

```
        {
            "Value": 14
        },
        {
            "Value": 15
        },
        {
            "Value": 16
        },
        {
            "Value": 17
        },
        {
            "Value": 18
        }
    ],
    "Phase2DHGroupNumbers": [
        {
            "Value": 14
        },
        {
            "Value": 15
        },
        {
            "Value": 16
        },
        {
            "Value": 17
        },
        {
            "Value": 18
        }
    ]
},
{
    "OutsideIpAddress": "203.0.114.19"
}
],
},
"VgwTelemetry": [
    {
        "AcceptedRouteCount": 0,
        "LastStatusChange": "2019-09-10T21:56:54.000Z",
        "OutsideIpAddress": "203.0.113.17",
```

```

        "Status": "DOWN",
        "StatusMessage": "IPSEC IS DOWN"
    },
    {
        "AcceptedRouteCount": 0,
        "LastStatusChange": "2019-09-10T21:56:43.000Z",
        "OutsideIpAddress": "203.0.114.19",
        "Status": "DOWN",
        "StatusMessage": "IPSEC IS DOWN"
    }
]
}
}

```

- For API details, see [ModifyVpnTunnelOptions](#) in *AWS CLI Command Reference*.

monitor-instances

The following code example shows how to use `monitor-instances`.

AWS CLI

To enable detailed monitoring for an instance

This example command enables detailed monitoring for the specified instance.

Command:

```
aws ec2 monitor-instances --instance-ids i-1234567890abcdef0
```

Output:

```

{
  "InstanceMonitorings": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "Monitoring": {
        "State": "pending"
      }
    }
  ]
}

```



```
}
```

- For API details, see [MonitorInstances](#) in *AWS CLI Command Reference*.

move-address-to-vpc

The following code example shows how to use `move-address-to-vpc`.

AWS CLI

To move an address to EC2-VPC

This example moves Elastic IP address 54.123.4.56 to the EC2-VPC platform.

Command:

```
aws ec2 move-address-to-vpc --public-ip 54.123.4.56
```

Output:

```
{  
  "Status": "MoveInProgress"  
}
```

- For API details, see [MoveAddressToVpc](#) in *AWS CLI Command Reference*.

move-byoip-cidr-to-ipam

The following code example shows how to use `move-byoip-cidr-to-ipam`.

AWS CLI

To transfer a BYOIP CIDR to IPAM

The following `move-byoip-cidr-to-ipam` example transfers a BYOIP CIDR to IPAM.

(Linux):

```
aws ec2 move-byoip-cidr-to-ipam \  
  --region us-west-2 \  
  \
```

```
--ipam-pool-id ipam-pool-0a03d430ca3f5c035 \  
--ipam-pool-owner 111111111111 \  
--cidr 130.137.249.0/24
```

(Windows):

```
aws ec2 move-byoip-cidr-to-ipam ^  
  --region us-west-2 ^  
  --ipam-pool-id ipam-pool-0a03d430ca3f5c035 ^  
  --ipam-pool-owner 111111111111 ^  
  --cidr 130.137.249.0/24
```

Output:

```
{  
  "ByoipCidr": {  
    "Cidr": "130.137.249.0/24",  
    "State": "pending-transfer"  
  }  
}
```

For more information, see [Tutorial: Transfer an existing BYOIP IPv4 CIDR to IPAM](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [MoveByoipCidrToIpam](#) in *AWS CLI Command Reference*.

network-insights-access-scope

The following code example shows how to use `network-insights-access-scope`.

AWS CLI

To create Network Insights access scopes

The following `create-network-insights-access-scope` example creates a network insights access scope in your AWS account.

```
aws ec2 create-network-insights-access-scope \  
  --cli-input-json file://access-scope-file.json
```

Contents of `access-scope-file.json`:

```
{
  {
    "MatchPaths": [
      {
        "Source": {
          "ResourceStatement": {
            "Resources": [
              "vpc-abcd12e3"
            ]
          }
        }
      }
    ],
    "ExcludePaths": [
      {
        "Source": {
          "ResourceStatement": {
            "ResourceTypes": [
              "AWS::EC2::InternetGateway"
            ]
          }
        }
      }
    ]
  }
}
```

Output:

```
{
  "NetworkInsightsAccessScopeAnalysisId": "nisa-123456789111"
}{
  "NetworkInsightsAccessScope": {
    "NetworkInsightsAccessScopeId": "nis-123456789222",
    "NetworkInsightsAccessScopeArn": "arn:aws:ec2:us-
east-1:123456789222:network-insights-access-scope/nis-123456789222",
    "CreateDate": "2022-01-25T19:20:28.796000+00:00",
    "UpdateDate": "2022-01-25T19:20:28.797000+00:00"
  },
  "NetworkInsightsAccessScopeContent": {
    "NetworkInsightsAccessScopeId": "nis-04c0c0fbca737c404",
    "MatchPaths": [
      {
```

```

        "Source": {
            "ResourceStatement": {
                "Resources": [
                    "vpc-abcd12e3"
                ]
            }
        }
    ],
    "ExcludePaths": [
        {
            "Source": {
                "ResourceStatement": {
                    "ResourceTypes": [
                        "AWS::EC2::InternetGateway"
                    ]
                }
            }
        }
    ]
}

```

For more information, see [Getting started with Network Access Analyzer using the AWS CLI](#) in the *Network Access Analyzer Guide*.

- For API details, see [NetworkInsightsAccessScope](#) in *AWS CLI Command Reference*.

provision-byoip-cidr

The following code example shows how to use `provision-byoip-cidr`.

AWS CLI

To provision an address range

The following `provision-byoip-cidr` example provisions a public IP address range for use with AWS.

```

aws ec2 provision-byoip-cidr \
  --cidr 203.0.113.25/24 \
  --cidr-authorization-context Message="$text_message",Signature="$signed_message"

```

Output:

```
{
  "ByoipCidr": {
    "Cidr": "203.0.113.25/24",
    "State": "pending-provision"
  }
}
```

For more information about creating the messages strings for the authorization context, see [Bring Your Own IP Addresses](#) in the *Amazon EC2 User Guide*.

- For API details, see [ProvisionByoipCidr](#) in *AWS CLI Command Reference*.

provision-ipam-pool-cidr

The following code example shows how to use `provision-ipam-pool-cidr`.

AWS CLI**To provision a CIDR to an IPAM pool**

The following `provision-ipam-pool-cidr` example provisions a CIDR to an IPAM pool.

(Linux):

```
aws ec2 provision-ipam-pool-cidr \
  --ipam-pool-id ipam-pool-0533048da7d823723 \
  --cidr 10.0.0.0/24
```

(Windows):

```
aws ec2 provision-ipam-pool-cidr ^
  --ipam-pool-id ipam-pool-0533048da7d823723 ^
  --cidr 10.0.0.0/24
```

Output:

```
{
  "IpamPoolCidr": {
```

```
    "Cidr": "10.0.0.0/24",
    "State": "pending-provision"
  }
}
```

For more information, see [Provision CIDRs to a pool](#) in the *Amazon VPC IPAM User Guide*.

- For API details, see [ProvisionIpamPoolCidr](#) in *AWS CLI Command Reference*.

purchase-host-reservation

The following code example shows how to use `purchase-host-reservation`.

AWS CLI

To purchase a Dedicated Host Reservation

This example purchases the specified Dedicated Host Reservation offering for the specified Dedicated Host in your account.

Command:

```
aws ec2 purchase-host-reservation --offering-id hro-03f707bf363b6b324 --host-id-set
h-013abcd2a00cbd123
```

Output:

```
{
  "TotalHourlyPrice": "1.499",
  "Purchase": [
    {
      "HourlyPrice": "1.499",
      "InstanceFamily": "m4",
      "PaymentOption": "NoUpfront",
      "HostIdSet": [
        "h-013abcd2a00cbd123"
      ],
      "HostReservationId": "hr-0d418a3a4ffc669ae",
      "UpfrontPrice": "0.000",
      "Duration": 31536000
    }
  ],
}
```

```
"TotalUpfrontPrice": "0.000"  
}
```

- For API details, see [PurchaseHostReservation](#) in *AWS CLI Command Reference*.

purchase-reserved-instances-offering

The following code example shows how to use `purchase-reserved-instances-offering`.

AWS CLI

To purchase a Reserved Instance offering

This example command illustrates a purchase of a Reserved Instances offering, specifying an offering ID and instance count.

Command:

```
aws ec2 purchase-reserved-instances-offering --reserved-instances-offering-id  
ec06327e-dd07-46ee-9398-75b5fexample --instance-count 3
```

Output:

```
{  
  "ReservedInstancesId": "af9f760e-6f91-4559-85f7-4980eexample"  
}
```

- For API details, see [PurchaseReservedInstancesOffering](#) in *AWS CLI Command Reference*.

purchase-scheduled-instances

The following code example shows how to use `purchase-scheduled-instances`.

AWS CLI

To purchase a Scheduled Instance

This example purchases a Scheduled Instance.

Command:

```
aws ec2 purchase-scheduled-instances --purchase-requests file://purchase-
request.json
```

Purchase-request.json:

```
[
  {
    "PurchaseToken": "eyJ2IjoiMSIsInMiOjEsImMiOi...",
    "InstanceCount": 1
  }
]
```

Output:

```
{
  "ScheduledInstanceSet": [
    {
      "AvailabilityZone": "us-west-2b",
      "ScheduledInstanceId": "sci-1234-1234-1234-1234-123456789012",
      "HourlyPrice": "0.095",
      "CreateDate": "2016-01-25T21:43:38.612Z",
      "Recurrence": {
        "OccurrenceDaySet": [
          1
        ],
        "Interval": 1,
        "Frequency": "Weekly",
        "OccurrenceRelativeToEnd": false,
        "OccurrenceUnit": ""
      },
      "Platform": "Linux/UNIX",
      "TermEndDate": "2017-01-31T09:00:00Z",
      "InstanceCount": 1,
      "SlotDurationInHours": 32,
      "TermStartDate": "2016-01-31T09:00:00Z",
      "NetworkPlatform": "EC2-VPC",
      "TotalScheduledInstanceHours": 1696,
      "NextSlotStartTime": "2016-01-31T09:00:00Z",
      "InstanceType": "c4.large"
    }
  ]
}
```


- For API details, see [PurchaseScheduledInstances](#) in *AWS CLI Command Reference*.

reboot-instances

The following code example shows how to use `reboot-instances`.

AWS CLI

To reboot an Amazon EC2 instance

This example reboots the specified instance. If the command succeeds, no output is returned.

Command:

```
aws ec2 reboot-instances --instance-ids i-1234567890abcdef5
```

For more information, see *Reboot Your Instance* in the *Amazon Elastic Compute Cloud User Guide*.

- For API details, see [RebootInstances](#) in *AWS CLI Command Reference*.

register-image

The following code example shows how to use `register-image`.

AWS CLI

Example 1: To register an AMI using a manifest file

The following `register-image` example registers an AMI using the specified manifest file in Amazon S3.

```
aws ec2 register-image \  
  --name my-image \  
  --image-location my-s3-bucket/myimage/image.manifest.xml
```

Output:

```
{  
  "ImageId": "ami-1234567890EXAMPLE"  
}
```

For more information, see [Amazon Machine Images \(AMI\)](#) in the *Amazon EC2 User Guide*.

Example 2: To register an AMI using a snapshot of a root device

The following `register-image` example registers an AMI using the specified snapshot of an EBS root volume as device `/dev/xvda`. The block device mapping also includes an empty 100 GiB EBS volume as device `/dev/xvdf`.

```
aws ec2 register-image \  
  --name my-image \  
  --root-device-name /dev/xvda \  
  --block-device-mappings DeviceName=/dev/  
xvda,Ebs={SnapshotId=snap-0db2cf683925d191f} DeviceName=/dev/  
xvdf,Ebs={VolumeSize=100}
```

Output:

```
{  
  "ImageId": "ami-1a2b3c4d5eEXAMPLE"  
}
```

For more information, see [Amazon Machine Images \(AMI\)](#) in the *Amazon EC2 User Guide*.

- For API details, see [RegisterImage](#) in *AWS CLI Command Reference*.

register-instance-event-notification-attributes

The following code example shows how to use `register-instance-event-notification-attributes`.

AWS CLI

Example 1: To include all tags in event notifications

The following `register-instance-event-notification-attributes` example includes all tags in event notifications.

```
aws ec2 register-instance-event-notification-attributes \  
  --instance-tag-attribute IncludeAllTagsOfInstance=true
```

Output:

```
{
  "InstanceTagAttribute": {
    "InstanceTagKeys": [],
    "IncludeAllTagsOfInstance": true
  }
}
```

For more information, see [Scheduled events for your instances](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

Example 2: To include specific tags in event notifications

The following `register-instance-event-notification-attributes` example includes the specified tags in event notifications. You cannot specify tags if `IncludeAllTagsOfInstance` is `true`.

```
aws ec2 register-instance-event-notification-attributes \
  --instance-tag-attribute InstanceTagKeys="tag-key1","tag-key2"
```

Output:

```
{
  "InstanceTagAttribute": {
    "InstanceTagKeys": [
      "tag-key1",
      "tag-key2"
    ],
    "IncludeAllTagsOfInstance": false
  }
}
```

For more information, see [Scheduled events for your instances](#) in the *Amazon Elastic Compute Cloud User Guide for Linux Instances*.

- For API details, see [RegisterInstanceEventNotificationAttributes](#) in *AWS CLI Command Reference*.

register-transit-gateway-multicase-group-sources

The following code example shows how to use `register-transit-gateway-multicase-group-sources`.

AWS CLI

To register a source with a transit gateway multicast group.

The following `register-transit-gateway-multicast-group-sources` example registers the specified network interface group source with a multicast group.

```
aws ec2 register-transit-gateway-multicast-group-sources \
  --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef79d6e597 \
  --group-ip-address 224.0.1.0 \
  --network-interface-ids eni-07f290fc3c090cbae
```

Output:

```
{
  "RegisteredMulticastGroupSources": {
    "TransitGatewayMulticastDomainId": "tgw-mcast-domain-0c4905cef79d6e597",
    "RegisteredNetworkInterfaceIds": [
      "eni-07f290fc3c090cbae"
    ],
    "GroupIpAddress": "224.0.1.0"
  }
}
```

For more information, see [Register Sources with a Multicast Group](#) in the *AWS Transit Gateways User Guide*.

- For API details, see [RegisterTransitGatewayMulticastGroupSources](#) in *AWS CLI Command Reference*.

register-transit-gateway-multicast-group-members

The following code example shows how to use `register-transit-gateway-multicast-group-members`.

AWS CLI

To view the information about the transit gateway multicast domain associations

The following `register-transit-gateway-multicast-group-members` example returns the associations for the specified multicast domain.

```
aws ec2 register-transit-gateway-multicast-group-members \  
  --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef79d6e597 \  
  --group-ip-address 224.0.1.0 \  
  --network-interface-ids eni-0e246d32695012e81
```

Output:

```
{  
  "RegisteredMulticastGroupMembers": {  
    "TransitGatewayMulticastDomainId": "tgw-mcast-domain-0c4905cef79d6e597",  
    "RegisteredNetworkInterfaceIds": [  
      "eni-0e246d32695012e81"  
    ],  
    "GroupIpAddress": "224.0.1.0"  
  }  
}
```

For more information, see [Managing multicast domains](#) in the *Transit Gateways User Guide*.

- For API details, see [RegisterTransitGatewayMulticastGroupMembers](#) in *AWS CLI Command Reference*.

register-transit-gateway-multicast-group-sources

The following code example shows how to use `register-transit-gateway-multicast-group-sources`.

AWS CLI

To register a source with a transit gateway multicast group.

The following `register-transit-gateway-multicast-group-sources` example registers the specified network interface group source with a multicast group.

```
aws ec2 register-transit-gateway-multicast-group-sources \  
  --transit-gateway-multicast-domain-id tgw-mcast-domain-0c4905cef79d6e597 \  
  --group-ip-address 224.0.1.0 \  
  --network-interface-ids eni-07f290fc3c090cbae
```

Output:

```
{
  "RegisteredMulticastGroupSources": {
    "TransitGatewayMulticastDomainId": "tgw-mcast-domain-0c4905cef79d6e597",
    "RegisteredNetworkInterfaceIds": [
      "eni-07f290fc3c090cbae"
    ],
    "GroupIpAddress": "224.0.1.0"
  }
}
```

For more information, see [Managing multicast domains](#) in the *Transit Gateways Guide*.

- For API details, see [RegisterTransitGatewayMulticastGroupSources](#) in *AWS CLI Command Reference*.

reject-transit-gateway-peering-attachment

The following code example shows how to use `reject-transit-gateway-peering-attachment`.

AWS CLI

To reject a transit gateway peering attachment

The following `reject-transit-gateway-peering-attachment` example rejects the specified transit gateway peering attachment request. The `--region` parameter specifies the Region that the accepter transit gateway is located in.

```
aws ec2 reject-transit-gateway-peering-attachment \
  --transit-gateway-attachment-id tgw-attach-4455667788aabbccd \
  --region us-east-2
```

Output:

```
{
  "TransitGatewayPeeringAttachment": {
    "TransitGatewayAttachmentId": "tgw-attach-4455667788aabbccd",
    "RequesterTgwInfo": {
      "TransitGatewayId": "tgw-123abc05e04123abc",
      "OwnerId": "123456789012",
      "Region": "us-west-2"
    }
  }
}
```

```

    },
    "AccepterTgwInfo": {
      "TransitGatewayId": "tgw-11223344aabbcc112",
      "OwnerId": "123456789012",
      "Region": "us-east-2"
    },
    "State": "rejecting",
    "CreationTime": "2019-12-09T11:50:31.000Z"
  }
}

```

For more information, see [Transit Gateway Peering Attachments](#) in the *Transit Gateways Guide*.

- For API details, see [RejectTransitGatewayPeeringAttachment](#) in *AWS CLI Command Reference*.

reject-transit-gateway-vpc-attachment

The following code example shows how to use `reject-transit-gateway-vpc-attachment`.

AWS CLI

To reject a transit gateway VPC attachment

The following `reject-transit-gateway-vpc-attachment` example rejects the specified transit gateway VPC attachment.

```

aws ec2 reject-transit-gateway-vpc-attachment \
  --transit-gateway-attachment-id tgw-attach-0a34fe6b4fEXAMPLE

```

Output:

```

{
  "TransitGatewayVpcAttachment": {
    "TransitGatewayAttachmentId": "tgw-attach-0a34fe6b4fEXAMPLE",
    "TransitGatewayId": "tgw-0262a0e521EXAMPLE",
    "VpcId": "vpc-07e8ffd50fEXAMPLE",
    "VpcOwnerId": "111122223333",
    "State": "pending",
    "SubnetIds": [
      "subnet-0752213d59EXAMPLE"
    ],
    "CreationTime": "2019-07-10T17:33:46.000Z",
  }
}

```

```
    "Options": {
      "DnsSupport": "enable",
      "Ipv6Support": "disable"
    }
  }
}
```

For more information, see [Transit gateway attachments to a VPC](#) in the *Transit Gateways Guide*.

- For API details, see [RejectTransitGatewayVpcAttachment](#) in *AWS CLI Command Reference*.

reject-transit-gateway-vpc-attachments

The following code example shows how to use `reject-transit-gateway-vpc-attachments`.

AWS CLI

To reject a transit gateway VPC attachment

The following `reject-transit-gateway-vpc-attachment` example rejects the specified transit gateway VPC attachment.

```
aws ec2 reject-transit-gateway-vpc-attachment \
  --transit-gateway-attachment-id tgw-attach-0a34fe6b4fEXAMPLE
```

Output:

```
{
  "TransitGatewayVpcAttachment": {
    "TransitGatewayAttachmentId": "tgw-attach-0a34fe6b4fEXAMPLE",
    "TransitGatewayId": "tgw-0262a0e521EXAMPLE",
    "VpcId": "vpc-07e8ffd50fEXAMPLE",
    "VpcOwnerId": "111122223333",
    "State": "pending",
    "SubnetIds": [
      "subnet-0752213d59EXAMPLE"
    ],
    "CreationTime": "2019-07-10T17:33:46.000Z",
    "Options": {
      "DnsSupport": "enable",
      "Ipv6Support": "disable"
    }
  }
}
```



```
}  
}
```

For more information, see [Transit gateway attachments to a VPC](#) in the *Transit Gateways Guide*.

- For API details, see [RejectTransitGatewayVpcAttachments](#) in *AWS CLI Command Reference*.

reject-vpc-endpoint-connections

The following code example shows how to use `reject-vpc-endpoint-connections`.

AWS CLI

To reject an interface endpoint connection request

This example rejects the specified endpoint connection request for the specified endpoint service.

Command:

```
aws ec2 reject-vpc-endpoint-connections --service-id vpce-svc-03d5ebb7d9579a2b3 --  
vpc-endpoint-ids vpce-0c1308d7312217abc
```

Output:

```
{  
  "Unsuccessful": []  
}
```

- For API details, see [RejectVpcEndpointConnections](#) in *AWS CLI Command Reference*.

reject-vpc-peering-connection

The following code example shows how to use `reject-vpc-peering-connection`.

AWS CLI

To reject a VPC peering connection

This example rejects the specified VPC peering connection request.

Command:

```
aws ec2 reject-vpc-peering-connection --vpc-peering-connection-id pcx-1a2b3c4d
```

Output:

```
{
  "Return": true
}
```

- For API details, see [RejectVpcPeeringConnection](#) in *AWS CLI Command Reference*.

release-address

The following code example shows how to use `release-address`.

AWS CLI

To release an Elastic IP addresses for EC2-Classic

This example releases an Elastic IP address for use with instances in EC2-Classic. If the command succeeds, no output is returned.

Command:

```
aws ec2 release-address --public-ip 198.51.100.0
```

To release an Elastic IP address for EC2-VPC

This example releases an Elastic IP address for use with instances in a VPC. If the command succeeds, no output is returned.

Command:

```
aws ec2 release-address --allocation-id eipalloc-64d5890a
```

- For API details, see [ReleaseAddress](#) in *AWS CLI Command Reference*.

release-hosts

The following code example shows how to use `release-hosts`.

AWS CLI

To release a Dedicated host from your account

To release a Dedicated host from your account. Instances that are on the host must be stopped or terminated before the host can be released.

Command:

```
aws ec2 release-hosts --host-id=h-0029d6e3cacf1b3da
```

Output:

```
{
  "Successful": [
    "h-0029d6e3cacf1b3da"
  ],
  "Unsuccessful": []
}
```

- For API details, see [ReleaseHosts](#) in *AWS CLI Command Reference*.

release-ipam-pool-allocation

The following code example shows how to use `release-ipam-pool-allocation`.

AWS CLI

To release an IPAM pool allocation

In this example, you're an IPAM delegated admin who tried to delete an IPAM pool but received an error that you cannot delete the pool while the pool has allocations. You are using this command to release a pool allocation.

Note the following:

You can only use this command for custom allocations. To remove an allocation for a resource without deleting the resource, set its monitored state to false using [modify-ipam-resource-cidr](#). To complete this request, you'll need the IPAM pool ID, which you can get with [describe-ipam-pools](#). You'll also need the allocation ID, which you can get with [get-ipam-pool-](#)

[allocations](#). If you do not want to remove allocations one by one, you can use the `--cascade` option when you delete an IPAM pool to automatically release any allocations in the pool before deleting it. There are a number of prerequisites before running this command. For more information, see [Release an allocation](#) in the *Amazon VPC IPAM User Guide*. The `--region` in which you run this command must be the locale of the IPAM pool where the allocation is.

The following `release-ipam-pool-allocation` example releases an IPAM pool allocation.

```
aws ec2 release-ipam-pool-allocation \
  --ipam-pool-id ipam-pool-07bdd12d7c94e4693 \
  --cidr 10.0.0.0/23 \
  --ipam-pool-allocation-id ipam-pool-alloc-0e66a1f730da54791b99465b79e7d1e89 \
  --region us-west-1
```

Output:

```
{
  "Success": true
}
```

Once you release an allocation, you may want to run [delete-ipam-pool](#).

- For API details, see [ReleaseIpamPoolAllocation](#) in *AWS CLI Command Reference*.

replace-iam-instance-profile-association

The following code example shows how to use `replace-iam-instance-profile-association`.

AWS CLI

To replace an IAM instance profile for an instance

This example replaces the IAM instance profile represented by the association `iip-assoc-060bae234aac2e7fa` with the IAM instance profile named `AdminRole`.

```
aws ec2 replace-iam-instance-profile-association \
  --iam-instance-profile Name=AdminRole \
  --association-id iip-assoc-060bae234aac2e7fa
```

Output:

```
{
  "IamInstanceProfileAssociation": {
    "InstanceId": "i-087711ddaf98f9489",
    "State": "associating",
    "AssociationId": "iip-assoc-0b215292fab192820",
    "IamInstanceProfile": {
      "Id": "AIPAJLNLDX3AMYZWNWYYAY",
      "Arn": "arn:aws:iam::123456789012:instance-profile/AdminRole"
    }
  }
}
```

- For API details, see [ReplacelamInstanceProfileAssociation](#) in *AWS CLI Command Reference*.

replace-network-acl-association

The following code example shows how to use `replace-network-acl-association`.

AWS CLI

To replace the network ACL associated with a subnet

This example associates the specified network ACL with the subnet for the specified network ACL association.

Command:

```
aws ec2 replace-network-acl-association --association-id aclassoc-e5b95c8c --
network-acl-id acl-5fb85d36
```

Output:

```
{
  "NewAssociationId": "aclassoc-3999875b"
}
```

- For API details, see [ReplaceNetworkAclAssociation](#) in *AWS CLI Command Reference*.

replace-network-acl-entry

The following code example shows how to use `replace-network-acl-entry`.

AWS CLI

To replace a network ACL entry

This example replaces an entry for the specified network ACL. The new rule 100 allows ingress traffic from 203.0.113.12/24 on UDP port 53 (DNS) into any associated subnet.

Command:

```
aws ec2 replace-network-acl-entry --network-acl-id acl-5fb85d36 --ingress --rule-number 100 --protocol udp --port-range From=53,To=53 --cidr-block 203.0.113.12/24 --rule-action allow
```

- For API details, see [ReplaceNetworkAclEntry](#) in *AWS CLI Command Reference*.

replace-route-table-association

The following code example shows how to use `replace-route-table-association`.

AWS CLI

To replace the route table associated with a subnet

This example associates the specified route table with the subnet for the specified route table association.

Command:

```
aws ec2 replace-route-table-association --association-id rtbassoc-781d0d1a --route-table-id rtb-22574640
```

Output:

```
{
  "NewAssociationId": "rtbassoc-3a1f0f58"
}
```

- For API details, see [ReplaceRouteTableAssociation](#) in *AWS CLI Command Reference*.

replace-route

The following code example shows how to use `replace-route`.

AWS CLI

To replace a route

This example replaces the specified route in the specified route table. The new route matches the specified CIDR and sends the traffic to the specified virtual private gateway. If the command succeeds, no output is returned.

Command:

```
aws ec2 replace-route --route-table-id rtb-22574640 --destination-cidr-block 10.0.0.0/16 --gateway-id vgw-9a4cacf3
```

- For API details, see [ReplaceRoute](#) in *AWS CLI Command Reference*.

replace-transit-gateway-route

The following code example shows how to use `replace-transit-gateway-route`.

AWS CLI

To replace the specified route in the specified transit gateway route table

The following `replace-transit-gateway-route` example replaces the route in the specified transit gateway route table.

```
aws ec2 replace-transit-gateway-route \
  --destination-cidr-block 10.0.2.0/24 \
  --transit-gateway-attachment-id tgw-attach-09b52ccdb5EXAMPLE \
  --transit-gateway-route-table-id tgw-rtb-0a823edbdeEXAMPLE
```

Output:

```
{
  "Route": {
    "DestinationCidrBlock": "10.0.2.0/24",
    "TransitGatewayAttachments": [
      {
        "ResourceId": "vpc-4EXAMPLE",
        "TransitGatewayAttachmentId": "tgw-attach-09b52ccdb5EXAMPLE",
        "ResourceType": "vpc"
      }
    ]
  }
}
```

```
    }  
  ],  
  "Type": "static",  
  "State": "active"  
}  
}
```

For more information, see [Transit gateway route tables](#) in the *Transit Gateways Guide*.

- For API details, see [ReplaceTransitGatewayRoute](#) in *AWS CLI Command Reference*.

report-instance-status

The following code example shows how to use `report-instance-status`.

AWS CLI

To report status feedback for an instance

This example command reports status feedback for the specified instance.

Command:

```
aws ec2 report-instance-status --instances i-1234567890abcdef0 --status impaired --  
reason-codes unresponsive
```

- For API details, see [ReportInstanceStatus](#) in *AWS CLI Command Reference*.

request-spot-fleet

The following code example shows how to use `request-spot-fleet`.

AWS CLI

To request a Spot fleet in the subnet with the lowest price

This example command creates a Spot fleet request with two launch specifications that differ only by subnet. The Spot fleet launches the instances in the specified subnet with the lowest price. If the instances are launched in a default VPC, they receive a public IP address by default. If the instances are launched in a nondefault VPC, they do not receive a public IP address by default.

Note that you can't specify different subnets from the same Availability Zone in a Spot fleet request.

Command:

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

Config.json:

```
{
  "SpotPrice": "0.04",
  "TargetCapacity": 2,
  "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role",
  "LaunchSpecifications": [
    {
      "ImageId": "ami-1a2b3c4d",
      "KeyName": "my-key-pair",
      "SecurityGroups": [
        {
          "GroupId": "sg-1a2b3c4d"
        }
      ],
      "InstanceType": "m3.medium",
      "SubnetId": "subnet-1a2b3c4d, subnet-3c4d5e6f",
      "IamInstanceProfile": {
        "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
      }
    }
  ]
}
```

Output:

```
{
  "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE"
}
```

To request a Spot fleet in the Availability Zone with the lowest price

This example command creates a Spot fleet request with two launch specifications that differ only by Availability Zone. The Spot fleet launches the instances in the specified Availability Zone with the lowest price. If your account supports EC2-VPC only, Amazon EC2 launches the Spot

instances in the default subnet of the Availability Zone. If your account supports EC2-Classic, Amazon EC2 launches the instances in EC2-Classic in the Availability Zone.

Command:

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

Config.json:

```
{
  "SpotPrice": "0.04",
  "TargetCapacity": 2,
  "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role",
  "LaunchSpecifications": [
    {
      "ImageId": "ami-1a2b3c4d",
      "KeyName": "my-key-pair",
      "SecurityGroups": [
        {
          "GroupId": "sg-1a2b3c4d"
        }
      ],
      "InstanceType": "m3.medium",
      "Placement": {
        "AvailabilityZone": "us-west-2a, us-west-2b"
      },
      "IamInstanceProfile": {
        "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
      }
    }
  ]
}
```

To launch Spot instances in a subnet and assign them public IP addresses

This example command assigns public addresses to instances launched in a nondefault VPC. Note that when you specify a network interface, you must include the subnet ID and security group ID using the network interface.

Command:

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

Config.json:

```
{
  "SpotPrice": "0.04",
  "TargetCapacity": 2,
  "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role",
  "LaunchSpecifications": [
    {
      "ImageId": "ami-1a2b3c4d",
      "KeyName": "my-key-pair",
      "InstanceType": "m3.medium",
      "NetworkInterfaces": [
        {
          "DeviceIndex": 0,
          "SubnetId": "subnet-1a2b3c4d",
          "Groups": [ "sg-1a2b3c4d" ],
          "AssociatePublicIpAddress": true
        }
      ],
      "IamInstanceProfile": {
        "Arn": "arn:aws:iam::880185128111:instance-profile/my-iam-role"
      }
    }
  ]
}
```

To request a Spot fleet using the diversified allocation strategy

This example command creates a Spot fleet request that launches 30 instances using the diversified allocation strategy. The launch specifications differ by instance type. The Spot fleet distributes the instances across the launch specifications such that there are 10 instances of each type.

Command:

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

Config.json:

```
{
  "SpotPrice": "0.70",
  "TargetCapacity": 30,
```

```
"AllocationStrategy": "diversified",
"IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role",
"LaunchSpecifications": [
  {
    "ImageId": "ami-1a2b3c4d",
    "InstanceType": "c4.2xlarge",
    "SubnetId": "subnet-1a2b3c4d"
  },
  {
    "ImageId": "ami-1a2b3c4d",
    "InstanceType": "m3.2xlarge",
    "SubnetId": "subnet-1a2b3c4d"
  },
  {
    "ImageId": "ami-1a2b3c4d",
    "InstanceType": "r3.2xlarge",
    "SubnetId": "subnet-1a2b3c4d"
  }
]
}
```

For more information, see Spot Fleet Requests in the *Amazon Elastic Compute Cloud User Guide*.

- For API details, see [RequestSpotFleet](#) in *AWS CLI Command Reference*.

request-spot-instances

The following code example shows how to use `request-spot-instances`.

AWS CLI

To request Spot Instances

This example command creates a one-time Spot Instance request for five instances in the specified Availability Zone. If your account supports EC2-VPC only, Amazon EC2 launches the instances in the default subnet of the specified Availability Zone. If your account supports EC2-Classic, Amazon EC2 launches the instances in EC2-Classic in the specified Availability Zone.

Command:

```
aws ec2 request-spot-instances --spot-price "0.03" --instance-count 5 --type "one-time" --launch-specification file://specification.json
```

Specification.json:

```
{
  "ImageId": "ami-1a2b3c4d",
  "KeyName": "my-key-pair",
  "SecurityGroupIds": [ "sg-1a2b3c4d" ],
  "InstanceType": "m3.medium",
  "Placement": {
    "AvailabilityZone": "us-west-2a"
  },
  "IamInstanceProfile": {
    "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}
```

Output:

```
{
  "SpotInstanceRequests": [
    {
      "Status": {
        "UpdateTime": "2014-03-25T20:54:21.000Z",
        "Code": "pending-evaluation",
        "Message": "Your Spot request has been submitted for review, and is
pending evaluation."
      },
      "ProductDescription": "Linux/UNIX",
      "SpotInstanceRequestId": "sir-df6f405d",
      "State": "open",
      "LaunchSpecification": {
        "Placement": {
          "AvailabilityZone": "us-west-2a"
        },
        "ImageId": "ami-1a2b3c4d",
        "KeyName": "my-key-pair",
        "SecurityGroups": [
          {
            "GroupName": "my-security-group",
            "GroupId": "sg-1a2b3c4d"
          }
        ],
        "Monitoring": {
          "Enabled": false
        }
      }
    }
  ]
}
```

```

        },
        "IamInstanceProfile": {
            "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
        },
        "InstanceType": "m3.medium"
    },
    "Type": "one-time",
    "CreateTime": "2014-03-25T20:54:20.000Z",
    "SpotPrice": "0.050000"
},
...
]
}

```

This example command creates a one-time Spot Instance request for five instances in the specified subnet. Amazon EC2 launches the instances in the specified subnet. If the VPC is a nondefault VPC, the instances do not receive a public IP address by default.

Command:

```
aws ec2 request-spot-instances --spot-price "0.050" --instance-count 5 --type "one-time" --launch-specification file://specification.json
```

Specification.json:

```

{
  "ImageId": "ami-1a2b3c4d",
  "SecurityGroupIds": [ "sg-1a2b3c4d" ],
  "InstanceType": "m3.medium",
  "SubnetId": "subnet-1a2b3c4d",
  "IamInstanceProfile": {
    "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}

```

Output:

```

{
  "SpotInstanceRequests": [
    {
      "Status": {
        "UpdateTime": "2014-03-25T22:21:58.000Z",

```

```

        "Code": "pending-evaluation",
        "Message": "Your Spot request has been submitted for review, and is
pending evaluation."
    },
    "ProductDescription": "Linux/UNIX",
    "SpotInstanceRequestId": "sir-df6f405d",
    "State": "open",
    "LaunchSpecification": {
        "Placement": {
            "AvailabilityZone": "us-west-2a"
        }
        "ImageId": "ami-1a2b3c4d"
        "SecurityGroups": [
            {
                "GroupName": "my-security-group",
                "GroupID": "sg-1a2b3c4d"
            }
        ]
        "SubnetId": "subnet-1a2b3c4d",
        "Monitoring": {
            "Enabled": false
        },
        "IamInstanceProfile": {
            "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
        },
        "InstanceType": "m3.medium",
    },
    "Type": "one-time",
    "CreateTime": "2014-03-25T22:21:58.000Z",
    "SpotPrice": "0.050000"
},
...
]
}

```

This example assigns a public IP address to the Spot Instances that you launch in a nondefault VPC. Note that when you specify a network interface, you must include the subnet ID and security group ID using the network interface.

Command:

```
aws ec2 request-spot-instances --spot-price "0.050" --instance-count 1 --type "one-time" --launch-specification file://specification.json
```

Specification.json:

```
{
  "ImageId": "ami-1a2b3c4d",
  "KeyName": "my-key-pair",
  "InstanceType": "m3.medium",
  "NetworkInterfaces": [
    {
      "DeviceIndex": 0,
      "SubnetId": "subnet-1a2b3c4d",
      "Groups": [ "sg-1a2b3c4d" ],
      "AssociatePublicIpAddress": true
    }
  ],
  "IamInstanceProfile": {
    "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}
```

- For API details, see [RequestSpotInstances](#) in *AWS CLI Command Reference*.

reset-address-attribute

The following code example shows how to use `reset-address-attribute`.

AWS CLI

To reset the domain name attribute associated with an elastic IP address

The following `reset-address-attribute` examples reset the domain name attribute of an elastic IP address.

Linux:

```
aws ec2 reset-address-attribute \
  --allocation-id eipalloc-abcdef01234567890 \
  --attribute domain-name
```

Windows:

```
aws ec2 reset-address-attribute ^
  --allocation-id eipalloc-abcdef01234567890 ^
```



```
--attribute domain-name
```

Output:

```
{
  "Addresses": [
    {
      "PublicIp": "192.0.2.0",
      "AllocationId": "eipalloc-abcdef01234567890",
      "PtrRecord": "example.com."
      "PtrRecordUpdate": {
        "Value": "example.net.",
        "Status": "PENDING"
      }
    }
  ]
}
```

To monitor the pending change, see [describe-addresses-attribute](#) in the *AWS CLI Command Reference*.

- For API details, see [ResetAddressAttribute](#) in *AWS CLI Command Reference*.

reset-ebs-default-kms-key-id

The following code example shows how to use `reset-ebs-default-kms-key-id`.

AWS CLI

To reset your default CMK for EBS encryption

The following `reset-ebs-default-kms-key-id` example resets the default CMK for EBS encryption for your AWS account in the current Region.

```
aws ec2 reset-ebs-default-kms-key-id
```

Output:

```
{
  "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/8c5b2c63-b9bc-45a3-a87a-5513eEXAMPLE"
}
```

- For API details, see [ResetEbsDefaultKmsKeyId](#) in *AWS CLI Command Reference*.

reset-fpga-image-attribute

The following code example shows how to use `reset-fpga-image-attribute`.

AWS CLI

To reset the attributes of an Amazon FPGA image

This example resets the load permissions for the specified AFI.

Command:

```
aws ec2 reset-fpga-image-attribute --fpga-image-id afi-0d123e123bfc85abc --attribute loadPermission
```

Output:

```
{
  "Return": true
}
```

- For API details, see [ResetFpgaImageAttribute](#) in *AWS CLI Command Reference*.

reset-image-attribute

The following code example shows how to use `reset-image-attribute`.

AWS CLI

To reset the launchPermission attribute

This example resets the `launchPermission` attribute for the specified AMI to its default value. By default, AMIs are private. If the command succeeds, no output is returned.

Command:

```
aws ec2 reset-image-attribute --image-id ami-5731123e --attribute launchPermission
```

- For API details, see [ResetImageAttribute](#) in *AWS CLI Command Reference*.

reset-instance-attribute

The following code example shows how to use `reset-instance-attribute`.

AWS CLI

To reset the `sourceDestCheck` attribute

This example resets the `sourceDestCheck` attribute of the specified instance. The instance must be in a VPC. If the command succeeds, no output is returned.

Command:

```
aws ec2 reset-instance-attribute --instance-id i-1234567890abcdef0 --attribute
sourceDestCheck
```

To reset the `kernel` attribute

This example resets the `kernel` attribute of the specified instance. The instance must be in the stopped state. If the command succeeds, no output is returned.

Command:

```
aws ec2 reset-instance-attribute --instance-id i-1234567890abcdef0 --attribute
kernel
```

To reset the `ramdisk` attribute

This example resets the `ramdisk` attribute of the specified instance. The instance must be in the stopped state. If the command succeeds, no output is returned.

Command:

```
aws ec2 reset-instance-attribute --instance-id i-1234567890abcdef0 --attribute
ramdisk
```

- For API details, see [ResetInstanceAttribute](#) in *AWS CLI Command Reference*.

reset-network-interface-attribute

The following code example shows how to use `reset-network-interface-attribute`.

AWS CLI

To reset a network interface attribute

The following `reset-network-interface-attribute` example resets the value of the source/destination checking attribute to `true`.

```
aws ec2 reset-network-interface-attribute \  
  --network-interface-id eni-686ea200 \  
  --source-dest-check
```

This command produces no output.

- For API details, see [ResetNetworkInterfaceAttribute](#) in *AWS CLI Command Reference*.

reset-snapshot-attribute

The following code example shows how to use `reset-snapshot-attribute`.

AWS CLI

To reset a snapshot attribute

This example resets the create volume permissions for snapshot `snap-1234567890abcdef0`. If the command succeeds, no output is returned.

Command:

```
aws ec2 reset-snapshot-attribute --snapshot-id snap-1234567890abcdef0 --attribute  
createVolumePermission
```

- For API details, see [ResetSnapshotAttribute](#) in *AWS CLI Command Reference*.

restore-address-to-classic

The following code example shows how to use `restore-address-to-classic`.

AWS CLI

To restore an address to EC2-Classic

This example restores Elastic IP address 198.51.100.0 to the EC2-Classic platform.

Command:

```
aws ec2 restore-address-to-classic --public-ip 198.51.100.0
```

Output:

```
{
  "Status": "MoveInProgress",
  "PublicIp": "198.51.100.0"
}
```

- For API details, see [RestoreAddressToClassic](#) in *AWS CLI Command Reference*.

restore-image-from-recycle-bin

The following code example shows how to use `restore-image-from-recycle-bin`.

AWS CLI

To restore an image from the Recycle Bin

The following `restore-image-from-recycle-bin` example restores AMI `ami-0111222333444abcd` from the Recycle Bin.

```
aws ec2 restore-image-from-recycle-bin \
  --image-id ami-0111222333444abcd
```

Output:

```
{
  "Return": true
}
```

For more information, see [Recover AMIs from the Recycle Bin](#) in the *Amazon Elastic Compute Cloud User Guide*.

- For API details, see [RestoreImageFromRecycleBin](#) in *AWS CLI Command Reference*.

restore-managed-prefix-list-version

The following code example shows how to use `restore-managed-prefix-list-version`.

AWS CLI

us-west-2**To restore a prefix list version**

The following `restore-managed-prefix-list-version` restores the entries from version 1 of the specified prefix list.

```
aws ec2 restore-managed-prefix-list-version \
  --prefix-list-id pl-0123456abcabcabc1 \
  --current-version 2 \
  --previous-version 1
```

Output:

```
{
  "PrefixList": {
    "PrefixListId": "pl-0123456abcabcabc1",
    "AddressFamily": "IPv4",
    "State": "restore-in-progress",
    "PrefixListArn": "arn:aws:ec2:us-west-2:123456789012:prefix-list/
pl-0123456abcabcabc1",
    "PrefixListName": "vpc-cidrs",
    "MaxEntries": 10,
    "Version": 2,
    "OwnerId": "123456789012"
  }
}
```

For more information, see [Managed prefix lists](#) in the *Amazon VPC User Guide*.

- For API details, see [RestoreManagedPrefixListVersion](#) in *AWS CLI Command Reference*.

restore-snapshot-from-recycle-bin

The following code example shows how to use `restore-snapshot-from-recycle-bin`.

AWS CLI

To restore snapshots from the Recycle Bin

The following `restore-snapshot-from-recycle-bin` example restores a snapshot from the Recycle Bin. When you restore a snapshot from the Recycle Bin, the snapshot is immediately available for use, and it is removed from the Recycle Bin. You can use a restored snapshot in the same way that you use any other snapshot in your account.

```
aws ec2 restore-snapshot-from-recycle-bin \  
  --snapshot-id snap-01234567890abcdef
```

This command produces no output.

For more information about Recycle Bin for Amazon EBS, see [Recover snapshots from the Recycle Bin](#) in the *Amazon EC2 User Guide*.

- For API details, see [RestoreSnapshotFromRecycleBin](#) in *AWS CLI Command Reference*.

restore-snapshot-tier

The following code example shows how to use `restore-snapshot-tier`.

AWS CLI

Example 1: To permanently restore an archived snapshot

The following `restore-snapshot-tier` example permanently restores the specified snapshot. Specify the `--snapshot-id` and include the `permanent-restore` option.

```
aws ec2 restore-snapshot-tier \  
  --snapshot-id snap-01234567890abcdef \  
  --permanent-restore
```

Output:

```
{  
  "SnapshotId": "snap-01234567890abcdef",  
  "IsPermanentRestore": true  
}
```

For more information about snapshot archiving, see [Archive Amazon EBS snapshots <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/snapshot-archive.html>](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/snapshot-archive.html) in the *Amazon EC2 User Guide*.

Example 2: To temporarily restore an archived snapshot

The following `restore-snapshot-tier` example temporarily restores the specified snapshot. Omit the `--permanent-restore` option. Specify the `--snapshot-id` and, for `temporary-restore-days`, specify the number of days for which to restore the snapshot. `temporary-restore-days` must be specified in days. The allowed range is 1 to 180. If you do not specify a value, it defaults to 1 day.

```
aws ec2 restore-snapshot-tier \  
  --snapshot-id snap-01234567890abcdef \  
  --temporary-restore-days 5
```

Output:

```
{  
  "SnapshotId": "snap-01234567890abcdef",  
  "RestoreDuration": 5,  
  "IsPermanentRestore": false  
}
```

For more information about snapshot archiving, see [Archive Amazon EBS snapshots](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/snapshot-archive.html) <<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/snapshot-archive.html>> in the *Amazon EC2 User Guide*.

Example 3: To modify the restore period

The following `restore-snapshot-tier` example changes the restore period for the specified snapshot to 10 days.

```
aws ec2 restore-snapshot-tier \  
  --snapshot-id snap-01234567890abcdef  
  --temporary-restore-days 10
```

Output:

```
{  
  "SnapshotId": "snap-01234567890abcdef",  
  "RestoreDuration": 10,  
  "IsPermanentRestore": false  
}
```


For more information about snapshot archiving, see Archive Amazon EBS snapshots <<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/snapshot-archive.html>> in the *Amazon EC2 User Guide*.

Example 4: To modify the restore type

The following `restore-snapshot-tier` example changes the restore type for the specified snapshot from temporary to permanent.

```
aws ec2 restore-snapshot-tier \  
  --snapshot-id snap-01234567890abcdef \  
  --permanent-restore
```

Output:

```
{  
  "SnapshotId": "snap-01234567890abcdef",  
  "IsPermanentRestore": true  
}
```

For more information about snapshot archiving, see Archive Amazon EBS snapshots <<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/snapshot-archive.html>> in the *Amazon EC2 User Guide*.

- For API details, see [RestoreSnapshotTier](#) in *AWS CLI Command Reference*.

revoke-client-vpn-ingress

The following code example shows how to use `revoke-client-vpn-ingress`.

AWS CLI

To revoke an authorization rule for a Client VPN endpoint

The following `revoke-client-vpn-ingress` example revokes a rule for internet access (`0.0.0.0/0`) for all groups.

```
aws ec2 revoke-client-vpn-ingress \  
  --client-vpn-endpoint-id cvpn-endpoint-123456789123abcde \  
  --target-network-cidr 0.0.0.0/0 --revoke-all-groups
```

Output:

```
{
  "Status": {
    "Code": "revoking"
  }
}
```

For more information, see [Authorization Rules](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [RevokeClientVpnIngress](#) in *AWS CLI Command Reference*.

revoke-security-group-egress

The following code example shows how to use `revoke-security-group-egress`.

AWS CLI

Example 1: To remove the rule that allows outbound traffic to a specific address range

The following `revoke-security-group-egress` example command removes the rule that grants access to the specified address ranges on TCP port 80.

```
aws ec2 revoke-security-group-egress \
  --group-id sg-026c12253ce15eff7 \
  --ip-permissions
  [{IpProtocol=tcp,FromPort=80,ToPort=80,IpRanges=[{CidrIp=10.0.0.0/16}]}
```

This command produces no output.

For more information, see [Security groups](#) in the *Amazon EC2 User Guide*.

Example 2: To remove the rule that allows outbound traffic to a specific security group

The following `revoke-security-group-egress` example command removes the rule that grants access to the specified security group on TCP port 80.

```
aws ec2 revoke-security-group-egress \
  --group-id sg-026c12253ce15eff7 \
  --ip-permissions '[{"IpProtocol": "tcp", "FromPort": 443, "ToPort":
  443, "UserIdGroupPairs": [{"GroupId": "sg-06df23a01ff2df86d"}]}'
```

This command produces no output.

For more information, see [Security groups](#) in the *Amazon EC2 User Guide*.

- For API details, see [RevokeSecurityGroupEgress](#) in *AWS CLI Command Reference*.

revoke-security-group-ingress

The following code example shows how to use `revoke-security-group-ingress`.

AWS CLI

Example 1: To remove a rule from a security group

The following `revoke-security-group-ingress` example removes TCP port 22 access for the `203.0.113.0/24` address range from the specified security group for a default VPC.

```
aws ec2 revoke-security-group-ingress \  
  --group-name mySecurityGroup \  
  --protocol tcp \  
  --port 22 \  
  --cidr 203.0.113.0/24
```

This command produces no output if it succeeds.

For more information, see [Security groups](#) in the *Amazon EC2 User Guide*.

Example 2: To remove a rule using the IP permissions set

The following `revoke-security-group-ingress` example uses the `ip-permissions` parameter to remove an inbound rule that allows the ICMP message Destination Unreachable: Fragmentation Needed and Don't Fragment was Set (Type 3, Code 4).

```
aws ec2 revoke-security-group-ingress \  
  --group-id sg-026c12253ce15eff7 \  
  --ip-permissions \  
  IpProtocol=icmp,FromPort=3,ToPort=4,IpRanges=[{CidrIp=0.0.0.0/0}]
```

This command produces no output if it succeeds.

For more information, see [Security groups](#) in the *Amazon EC2 User Guide*.

- For API details, see [RevokeSecurityGroupIngress](#) in *AWS CLI Command Reference*.

run-instances

The following code example shows how to use `run-instances`.

AWS CLI

Example 1: To launch an instance into a default subnet

The following `run-instances` example launches a single instance of type `t2.micro` into the default subnet for the current Region and associates it with the default subnet for the default VPC for the Region. The key pair is optional if you do not plan to connect to your instance using SSH (Linux) or RDP (Windows).

```
aws ec2 run-instances \  
  --image-id ami-0abcdef1234567890 \  
  --instance-type t2.micro \  
  --key-name MyKeyPair
```

Output:

```
{  
  "Instances": [  
    {  
      "AmiLaunchIndex": 0,  
      "ImageId": "ami-0abcdef1234567890",  
      "InstanceId": "i-1231231230abcdef0",  
      "InstanceType": "t2.micro",  
      "KeyName": "MyKeyPair",  
      "LaunchTime": "2018-05-10T08:05:20.000Z",  
      "Monitoring": {  
        "State": "disabled"  
      },  
      "Placement": {  
        "AvailabilityZone": "us-east-2a",  
        "GroupName": "",  
        "Tenancy": "default"  
      },  
      "PrivateDnsName": "ip-10-0-0-157.us-east-2.compute.internal",  
      "PrivateIpAddress": "10.0.0.157",  
      "ProductCodes": [],  
      "PublicDnsName": "",  
      "State": {  
        "Code": 0,
```

```
    "Name": "pending"
  },
  "StateTransitionReason": "",
  "SubnetId": "subnet-04a636d18e83cfac",
  "VpcId": "vpc-1234567890abcdef0",
  "Architecture": "x86_64",
  "BlockDeviceMappings": [],
  "ClientToken": "",
  "EbsOptimized": false,
  "Hypervisor": "xen",
  "NetworkInterfaces": [
    {
      "Attachment": {
        "AttachTime": "2018-05-10T08:05:20.000Z",
        "AttachmentId": "eni-attach-0e325c07e928a0405",
        "DeleteOnTermination": true,
        "DeviceIndex": 0,
        "Status": "attaching"
      },
      "Description": "",
      "Groups": [
        {
          "GroupName": "MySecurityGroup",
          "GroupId": "sg-0598c7d356eba48d7"
        }
      ],
      "Ipv6Addresses": [],
      "MacAddress": "0a:ab:58:e0:67:e2",
      "NetworkInterfaceId": "eni-0c0a29997760baee7",
      "OwnerId": "123456789012",
      "PrivateDnsName": "ip-10-0-0-157.us-east-2.compute.internal",
      "PrivateIpAddress": "10.0.0.157",
      "PrivateIpAddresses": [
        {
          "Primary": true,
          "PrivateDnsName": "ip-10-0-0-157.us-
east-2.compute.internal",
          "PrivateIpAddress": "10.0.0.157"
        }
      ],
      "SourceDestCheck": true,
      "Status": "in-use",
      "SubnetId": "subnet-04a636d18e83cfac",
      "VpcId": "vpc-1234567890abcdef0",
```

```

        "InterfaceType": "interface"
      }
    ],
    "RootDeviceName": "/dev/xvda",
    "RootDeviceType": "ebs",
    "SecurityGroups": [
      {
        "GroupName": "MySecurityGroup",
        "GroupId": "sg-0598c7d356eba48d7"
      }
    ],
    "SourceDestCheck": true,
    "StateReason": {
      "Code": "pending",
      "Message": "pending"
    },
    "Tags": [],
    "VirtualizationType": "hvm",
    "CpuOptions": {
      "CoreCount": 1,
      "ThreadsPerCore": 1
    },
    "CapacityReservationSpecification": {
      "CapacityReservationPreference": "open"
    },
    "MetadataOptions": {
      "State": "pending",
      "HttpTokens": "optional",
      "HttpPutResponseHopLimit": 1,
      "HttpEndpoint": "enabled"
    }
  }
],
"OwnerId": "123456789012",
"ReservationId": "r-02a3f596d91211712"
}

```

Example 2: To launch an instance into a non-default subnet and add a public IP address

The following `run-instances` example requests a public IP address for an instance that you're launching into a nondefault subnet. The instance is associated with the specified security group.

```
aws ec2 run-instances \
```

```
--image-id ami-0abcdef1234567890 \  
--instance-type t2.micro \  
--subnet-id subnet-08fc749671b2d077c \  
--security-group-ids sg-0b0384b66d7d692f9 \  
--associate-public-ip-address \  
--key-name MyKeyPair
```

For an example of the output for `run-instances`, see Example 1.

Example 3: To launch an instance with additional volumes

The following `run-instances` example uses a block device mapping, specified in `mapping.json`, to attach additional volumes at launch. A block device mapping can specify EBS volumes, instance store volumes, or both EBS volumes and instance store volumes.

```
aws ec2 run-instances \  
  --image-id ami-0abcdef1234567890 \  
  --instance-type t2.micro \  
  --subnet-id subnet-08fc749671b2d077c \  
  --security-group-ids sg-0b0384b66d7d692f9 \  
  --key-name MyKeyPair \  
  --block-device-mappings file://mapping.json
```

Contents of `mapping.json`. This example adds `/dev/sdh` an empty EBS volume with a size of 100 GiB.

```
[  
  {  
    "DeviceName": "/dev/sdh",  
    "Ebs": {  
      "VolumeSize": 100  
    }  
  }  
]
```

Contents of `mapping.json`. This example adds `ephemeral1` as an instance store volume.

```
[  
  {  
    "DeviceName": "/dev/sdc",  
    "VirtualName": "ephemeral1"  
  }  
]
```

```
}  
]
```

For an example of the output for `run-instances`, see Example 1.

For more information about block device mappings, see [Block device mapping](#) in the *Amazon EC2 User Guide*.

Example 4: To launch an instance and add tags on creation

The following `run-instances` example adds a tag with a key of `webserver` and value of `production` to the instance. The command also applies a tag with a key of `cost-center` and a value of `cc123` to any EBS volume that's created (in this case, the root volume).

```
aws ec2 run-instances \  
  --image-id ami-0abcdef1234567890 \  
  --instance-type t2.micro \  
  --count 1 \  
  --subnet-id subnet-08fc749671b2d077c \  
  --key-name MyKeyPair \  
  --security-group-ids sg-0b0384b66d7d692f9 \  
  --tag-specifications  
  'ResourceType=instance,Tags=[{Key=webserver,Value=production}]'  
  'ResourceType=volume,Tags=[{Key=cost-center,Value=cc123}]'
```

For an example of the output for `run-instances`, see Example 1.

Example 5: To launch an instance with user data

The following `run-instances` example passes user data in a file called `my_script.txt` that contains a configuration script for your instance. The script runs at launch.

```
aws ec2 run-instances \  
  --image-id ami-0abcdef1234567890 \  
  --instance-type t2.micro \  
  --count 1 \  
  --subnet-id subnet-08fc749671b2d077c \  
  --key-name MyKeyPair \  
  --security-group-ids sg-0b0384b66d7d692f9 \  
  --user-data file://my_script.txt
```

For an example of the output for `run-instances`, see Example 1.

For more information about instance user data, see [Working with instance user data](#) in the *Amazon EC2 User Guide*.

Example 6: To launch a burstable performance instance

The following `run-instances` example launches a `t2.micro` instance with the `unlimited` credit option. When you launch a T2 instance, if you do not specify `--credit-specification`, the default is the standard credit option. When you launch a T3 instance, the default is the `unlimited` credit option.

```
aws ec2 run-instances \  
  --image-id ami-0abcdef1234567890 \  
  --instance-type t2.micro \  
  --count 1 \  
  --subnet-id subnet-08fc749671b2d077c \  
  --key-name MyKeyPair \  
  --security-group-ids sg-0b0384b66d7d692f9 \  
  --credit-specification CpuCredits=unlimited
```

For an example of the output for `run-instances`, see Example 1.

For more information about burstable performance instances, see [Burstable performance instances](#) in the *Amazon EC2 User Guide*.

- For API details, see [RunInstances](#) in *AWS CLI Command Reference*.

run-scheduled-instances

The following code example shows how to use `run-scheduled-instances`.

AWS CLI

To launch a Scheduled Instance

This example launches the specified Scheduled Instance in a VPC.

Command:

```
aws ec2 run-scheduled-instances --scheduled-instance-id  
sci-1234-1234-1234-1234-123456789012 --instance-count 1 --launch-specification  
file://launch-specification.json
```

Launch-specification.json:

```
{
  "ImageId": "ami-12345678",
  "KeyName": "my-key-pair",
  "InstanceType": "c4.large",
  "NetworkInterfaces": [
    {
      "DeviceIndex": 0,
      "SubnetId": "subnet-12345678",
      "AssociatePublicIpAddress": true,
      "Groups": ["sg-12345678"]
    }
  ],
  "IamInstanceProfile": {
    "Name": "my-iam-role"
  }
}
```

Output:

```
{
  "InstanceIdSet": [
    "i-1234567890abcdef0"
  ]
}
```

This example launches the specified Scheduled Instance in EC2-Classic.

Command:

```
aws ec2 run-scheduled-instances --scheduled-instance-id
sci-1234-1234-1234-1234-123456789012 --instance-count 1 --launch-specification
file://launch-specification.json
```

Launch-specification.json:

```
{
  "ImageId": "ami-12345678",
  "KeyName": "my-key-pair",
  "SecurityGroupIds": ["sg-12345678"],
```

```
"InstanceType": "c4.large",
"Placement": {
  "AvailabilityZone": "us-west-2b"
}
"IamInstanceProfile": {
  "Name": "my-iam-role"
}
}
```

Output:

```
{
  "InstanceIdSet": [
    "i-1234567890abcdef0"
  ]
}
```

- For API details, see [RunScheduledInstances](#) in *AWS CLI Command Reference*.

search-local-gateway-routes

The following code example shows how to use `search-local-gateway-routes`.

AWS CLI

To search for routes in a local gateway route table

The following `search-local-gateway-routes` example searches for static routes in the specified local gateway route table.

```
aws ec2 search-local-gateway-routes \
  --local-gateway-route-table-id lgw-rtb-059615ef7dEXAMPLE \
  --filters "Name=type,Values=static"
```

Output:

```
{
  "Route": {
    "DestinationCidrBlock": "0.0.0.0/0",
    "LocalGatewayVirtualInterfaceGroupId": "lgw-vif-grp-07145b276bEXAMPLE",
    "Type": "static",
```

```

    "State": "deleted",
    "LocalGatewayRouteTableId": "lgw-rtb-059615ef7EXAMPLE"
  }
}

```

- For API details, see [SearchLocalGatewayRoutes](#) in *AWS CLI Command Reference*.

search-transit-gateway-multicast-groups

The following code example shows how to use `search-transit-gateway-multicast-groups`.

AWS CLI

To search one or more transit gateway multicast groups and return the group membership information

The following `search-transit-gateway-multicast-groups` example returns the group membership of the specified multicast group.

```

aws ec2 search-transit-gateway-multicast-groups \
  --transit-gateway-multicast-domain-id tgw-mcast-domain-000fb24d04EXAMPLE

```

Output:

```

{
  "MulticastGroups": [
    {
      "GroupIpAddress": "224.0.1.0",
      "TransitGatewayAttachmentId": "tgw-attach-0372e72386EXAMPLE",
      "SubnetId": "subnet-0187aff814EXAMPLE",
      "ResourceId": "vpc-0065acced4EXAMPLE",
      "ResourceType": "vpc",
      "NetworkInterfaceId": "eni-03847706f6EXAMPLE",
      "GroupMember": false,
      "GroupSource": true,
      "SourceType": "static"
    }
  ]
}

```

For more information, see [Managing multicast groups](#) in the *Transit Gateways Guide*.

- For API details, see [SearchTransitGatewayMulticastGroups](#) in *AWS CLI Command Reference*.

search-transit-gateway-routes

The following code example shows how to use `search-transit-gateway-routes`.

AWS CLI

To search for routes in the specified transit gateway route table

The following `search-transit-gateway-routes` example returns all the routes that are of type `static` in the specified route table.

```
aws ec2 search-transit-gateway-routes \  
  --transit-gateway-route-table-id tgw-rtb-0a823eddbdeEXAMPLE \  
  --filters "Name=type,Values=static"
```

Output:

```
{  
  "Routes": [  
    {  
      "DestinationCidrBlock": "10.0.2.0/24",  
      "TransitGatewayAttachments": [  
        {  
          "ResourceId": "vpc-4EXAMPLE",  
          "TransitGatewayAttachmentId": "tgw-attach-09b52ccdb5EXAMPLE",  
          "ResourceType": "vpc"  
        }  
      ],  
      "Type": "static",  
      "State": "active"  
    },  
    {  
      "DestinationCidrBlock": "10.1.0.0/24",  
      "TransitGatewayAttachments": [  
        {  
          "ResourceId": "vpc-4EXAMPLE",  
          "TransitGatewayAttachmentId": "tgw-attach-09b52ccdb5EXAMPLE",  
          "ResourceType": "vpc"  
        }  
      ],  
    }  
  ],  
}
```

```
        "Type": "static",
        "State": "active"
    }
],
"AdditionalRoutesAvailable": false
}
```

For more information, see [Transit gateway route tables](#) in the *Transit Gateways Guide*.

- For API details, see [SearchTransitGatewayRoutes](#) in *AWS CLI Command Reference*.

send-diagnostic-interrupt

The following code example shows how to use `send-diagnostic-interrupt`.

AWS CLI

To send a diagnostic interrupt

The following `send-diagnostic-interrupt` example sends a diagnostic interrupt to the specified instance.

```
aws ec2 send-diagnostic-interrupt \
  --instance-id i-1234567890abcdef0
```

This command produces no output.

- For API details, see [SendDiagnosticInterrupt](#) in *AWS CLI Command Reference*.

start-instances

The following code example shows how to use `start-instances`.

AWS CLI

To start an Amazon EC2 instance

This example starts the specified Amazon EBS-backed instance.

Command:

```
aws ec2 start-instances --instance-ids i-1234567890abcdef0
```

Output:

```
{
  "StartingInstances": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "CurrentState": {
        "Code": 0,
        "Name": "pending"
      },
      "PreviousState": {
        "Code": 80,
        "Name": "stopped"
      }
    }
  ]
}
```

For more information, see *Stop and Start Your Instance* in the *Amazon Elastic Compute Cloud User Guide*.

- For API details, see [StartInstances](#) in *AWS CLI Command Reference*.

start-network-insights-access-scope-analysis

The following code example shows how to use `start-network-insights-access-scope-analysis`.

AWS CLI**To start a Network Insights access scope analysis**

The following `start-network-insights-access-scope-analysis` example starts the scope analysis in your AWS account.

```
aws ec2 start-network-insights-access-scope-analysis \
  --region us-east-1 \
  --network-insights-access-scope-id nis-123456789111
```

Output:

```
{
```

```
"NetworkInsightsAccessScopeAnalysis": {
  "NetworkInsightsAccessScopeAnalysisId": "nisa-123456789222",
  "NetworkInsightsAccessScopeAnalysisArn": "arn:aws:ec2:us-
east-1:123456789012:network-insights-access-scope-analysis/nisa-123456789222",
  "NetworkInsightsAccessScopeId": "nis-123456789111",
  "Status": "running",
  "StartDate": "2022-01-26T00:47:06.814000+00:00"
}
}
```

For more information, see [Getting started with Network Access Analyzer using the AWS CLI](#) in the *Network Access Analyzer Guide*.

- For API details, see [StartNetworkInsightsAccessScopeAnalysis](#) in *AWS CLI Command Reference*.

start-network-insights-analysis

The following code example shows how to use `start-network-insights-analysis`.

AWS CLI

To analyze a path

The following `start-network-insights-analysis` example analyzes the path between the source and destination. To view the results of the path analysis, use the `describe-network-insights-analyses` command.

```
aws ec2 start-network-insights-analysis \
  --network-insights-path-id nip-0b26f224f1d131fa8
```

Output:

```
{
  "NetworkInsightsAnalysis": {
    "NetworkInsightsAnalysisId": "nia-02207aa13eb480c7a",
    "NetworkInsightsAnalysisArn": "arn:aws:ec2:us-east-1:123456789012:network-
insights-analysis/nia-02207aa13eb480c7a",
    "NetworkInsightsPathId": "nip-0b26f224f1d131fa8",
    "StartDate": "2021-01-20T22:58:37.495Z",
    "Status": "running"
  }
}
```



```
}
```

For more information, see [Getting started using the AWS CLI](#) in the *Reachability Analyzer Guide*.

- For API details, see [StartNetworkInsightsAnalysis](#) in *AWS CLI Command Reference*.

start-vpc-endpoint-service-private-dns-verification

The following code example shows how to use `start-vpc-endpoint-service-private-dns-verification`.

AWS CLI

To initiate the DNS verification process

The following `start-vpc-endpoint-service-private-dns-verification` example initiates the DNS verification process for the specified endpoint service.

```
aws ec2 start-vpc-endpoint-service-private-dns-verification \  
  --service-id vpce-svc-071afff70666e61e0
```

This command produces no output.

For more information, see [Manage DNS names](#) in the *AWS PrivateLink User Guide*.

- For API details, see [StartVpcEndpointServicePrivateDnsVerification](#) in *AWS CLI Command Reference*.

stop-instances

The following code example shows how to use `stop-instances`.

AWS CLI

Example 1: To stop an Amazon EC2 instance

The following `stop-instances` example stops the specified Amazon EBS-backed instance.

```
aws ec2 stop-instances \  
  --instance-ids i-1234567890abcdef0
```

Output:

```
{
  "StoppingInstances": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "CurrentState": {
        "Code": 64,
        "Name": "stopping"
      },
      "PreviousState": {
        "Code": 16,
        "Name": "running"
      }
    }
  ]
}
```

For more information, see [Stop and Start Your Instance](#) in the *Amazon Elastic Compute Cloud User Guide*.

Example 2: To hibernate an Amazon EC2 instance

The following `stop-instances` example hibernates Amazon EBS-backed instance if the instance is enabled for hibernation and meets the hibernation prerequisites. After the instance is put into hibernation the instance is stopped.

```
aws ec2 stop-instances \
  --instance-ids i-1234567890abcdef0 \
  --hibernate
```

Output:

```
{
  "StoppingInstances": [
    {
      "CurrentState": {
        "Code": 64,
        "Name": "stopping"
      },
      "InstanceId": "i-1234567890abcdef0",
      "PreviousState": {
```

```

        "Code": 16,
        "Name": "running"
      }
    ]
  }
}

```

For more information, see [Hibernate your On-Demand Linux instance](#) in the *Amazon Elastic Cloud Compute User Guide*.

- For API details, see [StopInstances](#) in *AWS CLI Command Reference*.

terminate-client-vpn-connections

The following code example shows how to use `terminate-client-vpn-connections`.

AWS CLI

To terminate a connection to a Client VPN endpoint

The following `terminate-client-vpn-connections` example terminates the specified connection to the Client VPN endpoint.

```

aws ec2 terminate-client-vpn-connections \
  --client-vpn-endpoint-id vpn-endpoint-123456789123abcde \
  --connection-id cvpn-connection-04edd76f5201e0cb8

```

Output:

```

{
  "ClientVpnEndpointId": "vpn-endpoint-123456789123abcde",
  "ConnectionStatuses": [
    {
      "ConnectionId": "cvpn-connection-04edd76f5201e0cb8",
      "PreviousStatus": {
        "Code": "active"
      },
      "CurrentStatus": {
        "Code": "terminating"
      }
    }
  ]
}

```

```
}
```

For more information, see [Client Connections](#) in the *AWS Client VPN Administrator Guide*.

- For API details, see [TerminateClientVpnConnections](#) in *AWS CLI Command Reference*.

terminate-instances

The following code example shows how to use `terminate-instances`.

AWS CLI

To terminate an Amazon EC2 instance

This example terminates the specified instance.

Command:

```
aws ec2 terminate-instances --instance-ids i-1234567890abcdef0
```

Output:

```
{
  "TerminatingInstances": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "CurrentState": {
        "Code": 32,
        "Name": "shutting-down"
      },
      "PreviousState": {
        "Code": 16,
        "Name": "running"
      }
    }
  ]
}
```

For more information, see *Using Amazon EC2 Instances* in the *AWS Command Line Interface User Guide*.

- For API details, see [TerminateInstances](#) in *AWS CLI Command Reference*.

unassign-ipv6-addresses

The following code example shows how to use `unassign-ipv6-addresses`.

AWS CLI

To unassign an IPv6 address from a network interface

This example unassigns the specified IPv6 address from the specified network interface.

Command:

```
aws ec2 unassign-ipv6-addresses --ipv6-addresses
2001:db8:1234:1a00:3304:8879:34cf:4071 --network-interface-id eni-23c49b68
```

Output:

```
{
  "NetworkInterfaceId": "eni-23c49b68",
  "UnassignedIpv6Addresses": [
    "2001:db8:1234:1a00:3304:8879:34cf:4071"
  ]
}
```

- For API details, see [UnassignIpv6Addresses](#) in *AWS CLI Command Reference*.

unassign-private-ip-addresses

The following code example shows how to use `unassign-private-ip-addresses`.

AWS CLI

To unassign a secondary private IP address from a network interface

This example unassigns the specified private IP address from the specified network interface. If the command succeeds, no output is returned.

Command:

```
aws ec2 unassign-private-ip-addresses --network-interface-id eni-e5aa89a3 --private-
ip-addresses 10.0.0.82
```

- For API details, see [UnassignPrivateIpAddresses](#) in *AWS CLI Command Reference*.

unassign-private-nat-gateway-address

The following code example shows how to use `unassign-private-nat-gateway-address`.

AWS CLI

To unassign a private IP address from your private NAT gateway

The following `unassign-private-nat-gateway-address` example unassigns the specified IP address from the specified private NAT gateway.

```
aws ec2 unassign-private-nat-gateway-address \
  --nat-gateway-id nat-1234567890abcdef0 \
  --private-ip-addresses 10.0.20.197
```

Output:

```
{
  "NatGatewayId": "nat-0ee3edd182361f662",
  "NatGatewayAddresses": [
    {
      "NetworkInterfaceId": "eni-0065a61b324d1897a",
      "PrivateIp": "10.0.20.197",
      "IsPrimary": false,
      "Status": "unassigning"
    }
  ]
}
```

For more information, see [NAT gateways](#) in the *Amazon VPC User Guide*.

- For API details, see [UnassignPrivateNatGatewayAddress](#) in *AWS CLI Command Reference*.

unmonitor-instances

The following code example shows how to use `unmonitor-instances`.

AWS CLI

To disable detailed monitoring for an instance

This example command disables detailed monitoring for the specified instance.

Command:

```
aws ec2 unmonitor-instances --instance-ids i-1234567890abcdef0
```

Output:

```
{
  "InstanceMonitorings": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "Monitoring": {
        "State": "disabling"
      }
    }
  ]
}
```

- For API details, see [UnmonitorInstances](#) in *AWS CLI Command Reference*.

update-security-group-rule-descriptions-egress

The following code example shows how to use `update-security-group-rule-descriptions-egress`.

AWS CLI

To update the description of an outbound security group rule

The following `update-security-group-rule-descriptions-egress` example updates the description for the security group rule for the specified port and IPv4 address range. The description 'Outbound HTTP access to server 2' replaces any existing description for the rule.

```
aws ec2 update-security-group-rule-descriptions-egress \
  --group-id sg-02f0d35a850ba727f \
  --ip-permissions
  IpProtocol=tcp,FromPort=80,ToPort=80,IpRanges=[{CidrIp=203.0.113.0/24,Description="Outbound
  HTTP access to server 2"}]
```

Output:

```
{
  "Return": true
}
```

For more information, see [Security group rules](#) in the *Amazon EC2 User Guide*.

- For API details, see [UpdateSecurityGroupRuleDescriptionsEgress](#) in *AWS CLI Command Reference*.

update-security-group-rule-descriptions-ingress

The following code example shows how to use `update-security-group-rule-descriptions-ingress`.

AWS CLI**Example 1: To update the description of an inbound security group rule with a CIDR source**

The following `update-security-group-rule-descriptions-ingress` example updates the description for the security group rule for the specified port and IPv4 address range. The description 'SSH access from ABC office' replaces any existing description for the rule.

```
aws ec2 update-security-group-rule-descriptions-ingress \
  --group-id sg-02f0d35a850ba727f \
  --ip-permissions
  IpProtocol=tcp,FromPort=22,ToPort=22,IpRanges='[{"CidrIp=203.0.113.0/16,Description="SSH
  access from corpnet"}]'
```

Output:

```
{
  "Return": true
}
```

For more information, see [Security group rules](#) in the *Amazon EC2 User Guide*.

Example 2: To update the description of an inbound security group rule with a prefix list source

The following `update-security-group-rule-descriptions-ingress` example updates the description for the security group rule for the specified port and prefix list. The description 'SSH access from ABC office' replaces any existing description for the rule.

```
aws ec2 update-security-group-rule-descriptions-ingress \  
  --group-id sg-02f0d35a850ba727f \  
  --ip-permissions  
  IpProtocol=tcp,FromPort=22,ToPort=22,PrefixListIds='[{"PrefixListId=pl-12345678,Description=  
  access from corpnet"}]'
```

Output:

```
{  
  "Return": true  
}
```

For more information, see [Security group rules](#) in the *Amazon EC2 User Guide*.

- For API details, see [UpdateSecurityGroupRuleDescriptionsIngress](#) in *AWS CLI Command Reference*.

withdraw-byoip-cidr

The following code example shows how to use `withdraw-byoip-cidr`.

AWS CLI

To stop advertising an address range

The following `withdraw-byoip-cidr` example stops advertising the specified address range.

```
aws ec2 withdraw-byoip-cidr  
  --cidr 203.0.113.25/24
```

Output:

```
{  
  "ByoipCidr": {  
    "Cidr": "203.0.113.25/24",  
    "StatusMessage": "ipv4pool-ec2-1234567890abcdef0",  
    "State": "advertised"  
  }  
}
```

```
}  
}
```

- For API details, see [WithdrawByoipCidr](#) in *AWS CLI Command Reference*.

Amazon EC2 Instance Connect examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon EC2 Instance Connect.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

send-ssh-public-key

The following code example shows how to use `send-ssh-public-key`.

AWS CLI

To send a an SSH public key to an instance

The following `send-ssh-public-key` example sends the specified SSH public key to the specified instance. The key is used to authenticate the specified user.

```
aws ec2-instance-connect send-ssh-public-key \  
  --instance-id i-1234567890abcdef0 \  
  --instance-os-user ec2-user \  
  --availability-zone us-east-2b \  
  --public-key-path /path/to/public-key
```

```
--ssh-public-key file://path/my-rsa-key.pub
```

This command produces no output.

- For API details, see [SendSshPublicKey](#) in *AWS CLI Command Reference*.

Amazon ECR examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon ECR.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

batch-check-layer-availability

The following code example shows how to use `batch-check-layer-availability`.

AWS CLI

To check the availability of a layer

The following `batch-check-layer-availability` example checks the availability of a layer with the digest

`sha256:6171c7451a50945f8ddd72f7732cc04d7a0d1f48138a426b2e64387fdeb834ed`
in the `cluster-autoscaler` repository.

```
aws ecr batch-check-layer-availability \  
  --repository-name cluster-autoscaler \  
  --digest sha256:6171c7451a50945f8ddd72f7732cc04d7a0d1f48138a426b2e64387fdeb834ed
```

```
--layer-digests
sha256:6171c7451a50945f8ddd72f7732cc04d7a0d1f48138a426b2e64387fdeb834ed
```

Output:

```
{
  "layers": [
    {
      "layerDigest":
"sha256:6171c7451a50945f8ddd72f7732cc04d7a0d1f48138a426b2e64387fdeb834ed",
      "layerAvailability": "AVAILABLE",
      "layerSize": 2777,
      "mediaType": "application/vnd.docker.container.image.v1+json"
    }
  ],
  "failures": []
}
```

- For API details, see [BatchCheckLayerAvailability](#) in *AWS CLI Command Reference*.

batch-delete-image

The following code example shows how to use `batch-delete-image`.

AWS CLI**Example 1: To delete an image**

The following `batch-delete-image` example deletes an image with the tag `precise` in the specified repository in the default registry for an account.

```
aws ecr batch-delete-image \
  --repository-name ubuntu \
  --image-ids imageTag=precise
```

Output:

```
{
  "failures": [],
  "imageIds": [
    {
      "imageTag": "precise",
```

```
        "imageDigest":
          "sha256:19665f1e6d1e504117a1743c0a3d3753086354a38375961f2e665416ef4b1b2f"
        }
      ]
    }
  }
```

Example 2: To delete multiple images

The following `batch-delete-image` example deletes all images tagged with `prod` and `team1` in the specified repository.

```
aws ecr batch-delete-image \
  --repository-name MyRepository \
  --image-ids imageTag=prod imageTag=team1
```

Output:

```
{
  "imageIds": [
    {
      "imageDigest": "sha256:123456789012",
      "imageTag": "prod"
    },
    {
      "imageDigest": "sha256:567890121234",
      "imageTag": "team1"
    }
  ],
  "failures": []
}
```

For more information, see [Deleting an Image](#) in the *Amazon ECR User Guide*.

- For API details, see [BatchDeleteImage](#) in *AWS CLI Command Reference*.

batch-get-image

The following code example shows how to use `batch-get-image`.

AWS CLI

Example 1: To get an image

The following `batch-get-image` example gets an image with the tag `v1.13.6` in a repository called `cluster-autoscaler` in the default registry for an account.

```
aws ecr batch-get-image \
  --repository-name cluster-autoscaler \
  --image-ids imageTag=v1.13.6
```

Output:

```
{
  "images": [
    {
      "registryId": "012345678910",
      "repositoryName": "cluster-autoscaler",
      "imageId": {
        "imageDigest":
"sha256:4a1c6567c38904384ebc64e35b7eeddd8451110c299e3368d2210066487d97e5",
        "imageTag": "v1.13.6"
      },
      "imageManifest": "{\n  \"schemaVersion\": 2,\n
\n  \"mediaType\": \"application/vnd.docker.distribution.manifest.v2+json
\n\",\n  \"config\": {\n    \"mediaType\": \"application/
vnd.docker.container.image.v1+json\",\n    \"size\": 2777,\n    \"digest
\": \"sha256:6171c7451a50945f8ddd72f7732cc04d7a0d1f48138a426b2e64387fdeb834ed
\n\n  },\n  \"layers\": [\n    {\n      \"mediaType
\": \"application/vnd.docker.image.rootfs.diff.tar.gzip
\n\",\n      \"size\": 17743696,\n      \"digest\":
\n\"sha256:39fafc05754f195f134ca11ecdb1c9a691ab0848c697fffeb5a85f900caaf6e1\"\n
\n    },\n    {\n      \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n      \"size\": 2565026,\n
\n      \"digest\":
\n\"sha256:8c8a779d3a537b767ae1091fe6e00c2590afd16767aa6096d1b318d75494819f
\n\n    },\n    {\n      \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n      \"size\": 28005981,\n
\n      \"digest\":
\n\"sha256:c44ba47496991c9982ee493b47fd25c252caabf2b4ae7dd679c9a27b6a3c8fb7\"\n
\n    },\n    {\n      \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\",\n      \"size\": 775,\n
\n      \"digest
\": \"sha256:e2c388b44226544363ca007be7b896bcce1baebea04da23cbd165eac30be650f\"\n
\n    }\n  ]\n}"
    },
    {
      "failures": []
    }
  ]
}
```

```
}
```

Example 2: To get multiple images

The following `batch-get-image` example displays details of all images tagged with `prod` and `team1` in the specified repository.

```
aws ecr batch-get-image \  
  --repository-name MyRepository \  
  --image-ids imageTag=prod imageTag=team1
```

Output:

```
{  
  "images": [  
    {  
      "registryId": "123456789012",  
      "repositoryName": "MyRepository",  
      "imageId": {  
        "imageDigest": "sha256:123456789012",  
        "imageTag": "prod"  
      },  
      "imageManifest": "manifestExample1"  
    },  
    {  
      "registryId": "567890121234",  
      "repositoryName": "MyRepository",  
      "imageId": {  
        "imageDigest": "sha256:123456789012",  
        "imageTag": "team1"  
      },  
      "imageManifest": "manifestExample2"  
    }  
  ],  
  "failures": []  
}
```

For more information, see [Images](#) in the *Amazon ECR User Guide*.

- For API details, see [BatchGetImage](#) in *AWS CLI Command Reference*.

complete-layer-upload

The following code example shows how to use `complete-layer-upload`.

AWS CLI

To complete an image layer upload

The following `complete-layer-upload` example completes an image layer upload to the `layer-test` repository.

```
aws ecr complete-layer-upload \  
  --repository-name layer-test \  
  --upload-id 6cb64b8a-9378-0e33-2ab1-b780fab8a9e9 \  
  --layer-digests 6cb64b8a-9378-0e33-2ab1-  
b780fab8a9e9:48074e6d3a68b39aad8ccc002cdad912d4148c0f92b3729323e
```

Output:

```
{  
  "uploadId": "6cb64b8a-9378-0e33-2ab1-b780fab8a9e9",  
  "layerDigest":  
    "sha256:9a77f85878aa1906f2020a0ecdf7a7e962d57e882250acd773383224b3fe9a02",  
  "repositoryName": "layer-test",  
  "registryId": "130757420319"  
}
```

- For API details, see [CompleteLayerUpload](#) in *AWS CLI Command Reference*.

create-repository

The following code example shows how to use `create-repository`.

AWS CLI

Example 1: To create a repository

The following `create-repository` example creates a repository inside the specified namespace in the default registry for an account.

```
aws ecr create-repository \  
  --namespace-name my-namespace
```



```
--repository-name project-a/nginx-web-app
```

Output:

```
{
  "repository": {
    "registryId": "123456789012",
    "repositoryName": "sample-repo",
    "repositoryArn": "arn:aws:ecr:us-west-2:123456789012:repository/project-a/nginx-web-app"
  }
}
```

For more information, see [Creating a Repository](#) in the *Amazon ECR User Guide*.

Example 2: To create a repository configured with image tag immutability

The following `create-repository` example creates a repository configured for tag immutability in the default registry for an account.

```
aws ecr create-repository \
  --repository-name sample-repo \
  --image-tag-mutability IMMUTABLE
```

Output:

```
{
  "repository": {
    "registryId": "123456789012",
    "repositoryName": "sample-repo",
    "repositoryArn": "arn:aws:ecr:us-west-2:123456789012:repository/sample-repo",
    "imageTagMutability": "IMMUTABLE"
  }
}
```

For more information, see [Image Tag Mutability](#) in the *Amazon ECR User Guide*.

Example 3: To create a repository configured with a scanning configuration

The following `create-repository` example creates a repository configured to perform a vulnerability scan on image push in the default registry for an account.

```
aws ecr create-repository \  
  --repository-name sample-repo \  
  --image-scanning-configuration scanOnPush=true
```

Output:

```
{  
  "repository": {  
    "registryId": "123456789012",  
    "repositoryName": "sample-repo",  
    "repositoryArn": "arn:aws:ecr:us-west-2:123456789012:repository/sample-  
repo",  
    "imageScanningConfiguration": {  
      "scanOnPush": true  
    }  
  }  
}
```

For more information, see [Image Scanning](#) in the *Amazon ECR User Guide*.

- For API details, see [CreateRepository](#) in *AWS CLI Command Reference*.

delete-lifecycle-policy

The following code example shows how to use `delete-lifecycle-policy`.

AWS CLI

To delete the lifecycle policy for a repository

The following `delete-lifecycle-policy` example deletes the lifecycle policy for the `hello-world` repository.

```
aws ecr delete-lifecycle-policy \  
  --repository-name hello-world
```

Output:

```
{  
  "registryId": "012345678910",
```

```

    "repositoryName": "hello-world",
    "lifecyclePolicyText": "{\"rules\": [{\"rulePriority\": 1, \"description\": \"Remove
untagged images.\", \"selection\": {\"tagStatus\": \"untagged\", \"countType\":
\"sinceImagePushed\", \"countUnit\": \"days\", \"countNumber\": 10}, \"action\": {\"type
\": \"expire\"}}]}",
    "lastEvaluatedAt": 0.0
}

```

- For API details, see [DeleteLifecyclePolicy](#) in *AWS CLI Command Reference*.

delete-repository-policy

The following code example shows how to use `delete-repository-policy`.

AWS CLI

To delete the repository policy for a repository

The following `delete-repository-policy` example deletes the repository policy for the `cluster-autoscaler` repository.

```

aws ecr delete-repository-policy \
  --repository-name cluster-autoscaler

```

Output:

```

{
  "registryId": "012345678910",
  "repositoryName": "cluster-autoscaler",
  "policyText": "{\n  \"Version\" : \"2008-10-17\",\n  \"Statement\" : [ {\n
  \"Sid\" : \"allow public pull\",\n    \"Effect\" : \"Allow\",\n    \"Principal\" :
  \"*\",\n    \"Action\" : [ \"ecr:BatchCheckLayerAvailability\", \"ecr:BatchGetImage
  \", \"ecr:GetDownloadUrlForLayer\" ]\n  } ]\n}"
}

```

- For API details, see [DeleteRepositoryPolicy](#) in *AWS CLI Command Reference*.

delete-repository

The following code example shows how to use `delete-repository`.

AWS CLI

To delete a repository

The following `delete-repository` example command force deletes the specified repository in the default registry for an account. The `--force` flag is required if the repository contains images.

```
aws ecr delete-repository \  
  --repository-name ubuntu \  
  --force
```

Output:

```
{  
  "repository": {  
    "registryId": "123456789012",  
    "repositoryName": "ubuntu",  
    "repositoryArn": "arn:aws:ecr:us-west-2:123456789012:repository/ubuntu"  
  }  
}
```

For more information, see [Deleting a Repository](#) in the *Amazon ECR User Guide*.

- For API details, see [DeleteRepository](#) in *AWS CLI Command Reference*.

describe-image-scan-findings

The following code example shows how to use `describe-image-scan-findings`.

AWS CLI

To describe the scan findings for an image

The following `describe-image-scan-findings` example returns the image scan findings for an image using the image digest in the specified repository in the default registry for an account.

```
aws ecr describe-image-scan-findings \  
  --repository-name sample-repo \  
  --image-id  
  imageDigest=sha256:74b2c688c700ec95a93e478cdb959737c148df3fbf5ea706abe0318726e885e6
```

Output:

```
{
  "imageScanFindings": {
    "findings": [
      {
        "name": "CVE-2019-5188",
        "description": "A code execution vulnerability exists in the directory rehashing functionality of E2fsprogs e2fsck 1.45.4. A specially crafted ext4 directory can cause an out-of-bounds write on the stack, resulting in code execution. An attacker can corrupt a partition to trigger this vulnerability.",
        "uri": "http://people.ubuntu.com/~ubuntu-security/cve/CVE-2019-5188",
        "severity": "MEDIUM",
        "attributes": [
          {
            "key": "package_version",
            "value": "1.44.1-1ubuntu1.1"
          },
          {
            "key": "package_name",
            "value": "e2fsprogs"
          },
          {
            "key": "CVSS2_VECTOR",
            "value": "AV:L/AC:L/Au:N/C:P/I:P/A:P"
          },
          {
            "key": "CVSS2_SCORE",
            "value": "4.6"
          }
        ]
      }
    ],
    "imageScanCompletedAt": 1579839105.0,
    "vulnerabilitySourceUpdatedAt": 1579811117.0,
    "findingSeverityCounts": {
      "MEDIUM": 1
    }
  },
  "registryId": "123456789012",
  "repositoryName": "sample-repo",
  "imageId": {
    "imageDigest":
    "sha256:74b2c688c700ec95a93e478cdb959737c148df3fbf5ea706abe0318726e885e6"
  }
}
```

```
  },
  "imageScanStatus": {
    "status": "COMPLETE",
    "description": "The scan was completed successfully."
  }
}
```

For more information, see [Image Scanning](#) in the *Amazon ECR User Guide*.

- For API details, see [DescribeImageScanFindings](#) in *AWS CLI Command Reference*.

describe-images

The following code example shows how to use `describe-images`.

AWS CLI

To describe an image in a repository

The following `describe-images` example displays details about an image in the `cluster-autoscaler` repository with the tag `v1.13.6`.

```
aws ecr describe-images \
  --repository-name cluster-autoscaler \
  --image-ids imageTag=v1.13.6
```

Output:

```
{
  "imageDetails": [
    {
      "registryId": "012345678910",
      "repositoryName": "cluster-autoscaler",
      "imageDigest":
"sha256:4a1c6567c38904384ebc64e35b7eeddd8451110c299e3368d2210066487d97e5",
      "imageTags": [
        "v1.13.6"
      ],
      "imageSizeInBytes": 48318255,
      "imagePushedAt": 1565128275.0
    }
  ]
}
```

```
}
```

- For API details, see [DescribeImages](#) in *AWS CLI Command Reference*.

describe-repositories

The following code example shows how to use `describe-repositories`.

AWS CLI

To describe the repositories in a registry

This example describes the repositories in the default registry for an account.

Command:

```
aws ecr describe-repositories
```

Output:

```
{
  "repositories": [
    {
      "registryId": "012345678910",
      "repositoryName": "ubuntu",
      "repositoryArn": "arn:aws:ecr:us-west-2:012345678910:repository/ubuntu"
    },
    {
      "registryId": "012345678910",
      "repositoryName": "test",
      "repositoryArn": "arn:aws:ecr:us-west-2:012345678910:repository/test"
    }
  ]
}
```

- For API details, see [DescribeRepositories](#) in *AWS CLI Command Reference*.

get-authorization-token

The following code example shows how to use `get-authorization-token`.

AWS CLI

To get an authorization token for your default registry

The following `get-authorization-token` example command gets an authorization token for your default registry.

```
aws ecr get-authorization-token
```

Output:

```
{
  "authorizationData": [
    {
      "authorizationToken": "QVdT0kN...",
      "expiresAt": 1448875853.241,
      "proxyEndpoint": "https://123456789012.dkr.ecr.us-west-2.amazonaws.com"
    }
  ]
}
```

- For API details, see [GetAuthorizationToken](#) in *AWS CLI Command Reference*.

get-download-url-for-layer

The following code example shows how to use `get-download-url-for-layer`.

AWS CLI

To get the download URL of a layer

The following `get-download-url-for-layer` example displays the download URL of a layer with the digest

sha256:6171c7451a50945f8ddd72f7732cc04d7a0d1f48138a426b2e64387fdeb834ed
in the `cluster-autoscaler` repository.

```
aws ecr get-download-url-for-layer \
  --repository-name cluster-autoscaler \
  --layer-digest
sha256:6171c7451a50945f8ddd72f7732cc04d7a0d1f48138a426b2e64387fdeb834ed
```


Output:

```
{
  "downloadUrl": "https://prod-us-west-2-starport-layer-bucket.s3.us-
west-2.amazonaws.com/e501-012345678910-9cb60dc0-7284-5643-3987-
da6dac0465f0/04620aac-66a5-4167-8232-55ee7ef6d565?X-Amz-Algorithm=AWS4-HMAC-
SHA256&X-Amz-Date=20190814T220617Z&X-Amz-SignedHeaders=host&X-Amz-Expires=3600&X-
Amz-Credential=AKIA32P3D2JDNMVAJLGF%2F20190814%2Fus-west-2%2Fs3%2Faws4_request&X-
Amz-Signature=9161345894947a1672467a0da7a1550f2f7157318312fe4941b59976239c3337",
  "layerDigest":
  "sha256:6171c7451a50945f8ddd72f7732cc04d7a0d1f48138a426b2e64387fdeb834ed"
}
```

- For API details, see [GetDownloadUrlForLayer](#) in *AWS CLI Command Reference*.

get-lifecycle-policy-preview

The following code example shows how to use `get-lifecycle-policy-preview`.

AWS CLI**To retrieve details for a lifecycle policy preview**

The following `get-lifecycle-policy-preview` example retrieves the result of a lifecycle policy preview for the specified repository in the default registry for an account.

Command:

```
aws ecr get-lifecycle-policy-preview \
  --repository-name "project-a/amazon-ecs-sample"
```

Output:

```
{
  "registryId": "012345678910",
  "repositoryName": "project-a/amazon-ecs-sample",
  "lifecyclePolicyText": "{\n  \"rules\": [\n    {\n\n      \"rulePriority\": 1,\n      \"description\": \"Expire images older than 14\n      days\",\n      \"selection\": {\n        \"tagStatus\": \"untagged\",\n\n        \"countType\": \"sinceImagePushed\",\n        \"countUnit\n      \": \"days\",\n      \"countNumber\": 14\n    },\n  ],\n}
```

```

  \"action\": {\n
    ]\n}\n",
    "status": "COMPLETE",
    "previewResults": [],
    "summary": {\n
      "expiringImageTotalCount": 0
    }
  }
}

```

For more information, see [Lifecycle Policies](#) in the *Amazon ECR User Guide*.

- For API details, see [GetLifecyclePolicyPreview](#) in *AWS CLI Command Reference*.

get-lifecycle-policy

The following code example shows how to use `get-lifecycle-policy`.

AWS CLI

To retrieve a lifecycle policy

The following `get-lifecycle-policy` example displays details of the lifecycle policy for the specified repository in the default registry for the account.

```

aws ecr get-lifecycle-policy \
  --repository-name "project-a/amazon-ecs-sample"

```

Output:

```

{
  "registryId": "123456789012",
  "repositoryName": "project-a/amazon-ecs-sample",
  "lifecyclePolicyText": "{\n\"rules\":[\n{\n\"rulePriority\":1,\n\"description\":\n\"Expire images older than 14 days\",\n\"selection\":{\n\"tagStatus\":\n\"untagged\",\n\"countType\":\n\"sinceImagePushed\",\n\"countUnit\":\n\"days\",\n\"countNumber\":14},\n\"action\":{\n\"type\":\n\"expire\"}}]\n}",
  "lastEvaluatedAt": 1504295007.0
}

```

For more information, see [Lifecycle Policies](#) in the *Amazon ECR User Guide*.

- For API details, see [GetLifecyclePolicy](#) in *AWS CLI Command Reference*.

get-login-password

The following code example shows how to use `get-login-password`.

AWS CLI

To retrieve a password to authenticate to a registry

The following `get-login-password` displays a password that you can use with a container client of your choice to authenticate to any Amazon ECR registry that your IAM principal has access to.

```
aws ecr get-login-password
```

Output:

```
<password>
```

To use with the Docker CLI, pipe the output of the `get-login-password` command to the `docker login` command. When retrieving the password, ensure that you specify the same Region that your Amazon ECR registry exists in.

```
aws ecr get-login-password \
  --region <region> \
| docker login \
  --username AWS \
  --password-stdin <aws_account_id>.dkr.ecr.<region>.amazonaws.com
```

For more information, see [Registry Authentication](#) in the *Amazon ECR User Guide*.

- For API details, see [GetLoginPassword](#) in *AWS CLI Command Reference*.

get-login

The following code example shows how to use `get-login`.

AWS CLI

To retrieve a Docker login command to your default registry

This example prints a command that you can use to log in to your default Amazon ECR registry.

Command:

```
aws ecr get-login
```

Output:

```
docker login -u AWS -p <password> -e none https://  
<aws_account_id>.dkr.ecr.<region>.amazonaws.com
```

To log in to another account's registry

This example prints one or more commands that you can use to log in to Amazon ECR registries associated with other accounts.

Command:

```
aws ecr get-login --registry-ids 012345678910 023456789012
```

Output:

```
docker login -u <username> -p <token-1> -e none <endpoint-1>  
docker login -u <username> -p <token-2> -e none <endpoint-2>
```

- For API details, see [GetLogin](#) in *AWS CLI Command Reference*.

get-repository-policy

The following code example shows how to use `get-repository-policy`.

AWS CLI**To retrieve the repository policy for a repository**

The following `get-repository-policy` example displays details about the repository policy for the `cluster-autoscaler` repository.

```
aws ecr get-repository-policy \  
  --repository-name cluster-autoscaler
```

Output:

```
{
  "registryId": "012345678910",
  "repositoryName": "cluster-autoscaler",
  "policyText": "{\n  \"Version\" : \"2008-10-17\",\n  \"Statement\" : [ {\n    \"Sid\" : \"allow public pull\",\n    \"Effect\" : \"Allow\",\n    \"Principal\" :\n    \"*\",\n    \"Action\" : [ \"ecr:BatchCheckLayerAvailability\", \"ecr:BatchGetImage\",\n    \"ecr:GetDownloadUrlForLayer\" ]\n  } ]\n}"
```

- For API details, see [GetRepositoryPolicy](#) in *AWS CLI Command Reference*.

initiate-layer-upload

The following code example shows how to use `initiate-layer-upload`.

AWS CLI

To initiate an image layer upload

The following `initiate-layer-upload` example initiates an image layer upload to the `layer-test` repository.

```
aws ecr initiate-layer-upload \
  --repository-name layer-test
```

Output:

```
{
  "partSize": 10485760,
  "uploadId": "6cb64b8a-9378-0e33-2ab1-b780fab8a9e9"
}
```

- For API details, see [InitiateLayerUpload](#) in *AWS CLI Command Reference*.

list-images

The following code example shows how to use `list-images`.

AWS CLI

To list the images in a repository

The following `list-images` example displays a list of the images in the `cluster-autoscaler` repository.

```
aws ecr list-images \  
  --repository-name cluster-autoscaler
```

Output:

```
{  
  "imageIds": [  
    {  
      "imageDigest":  
"sha256:99c6fb4377e9a420a1eb3b410a951c9f464eff3b7dbc76c65e434e39b94b6570",  
      "imageTag": "v1.13.8"  
    },  
    {  
      "imageDigest":  
"sha256:99c6fb4377e9a420a1eb3b410a951c9f464eff3b7dbc76c65e434e39b94b6570",  
      "imageTag": "v1.13.7"  
    },  
    {  
      "imageDigest":  
"sha256:4a1c6567c38904384ebc64e35b7eeddd8451110c299e3368d2210066487d97e5",  
      "imageTag": "v1.13.6"  
    }  
  ]  
}
```

- For API details, see [ListImages](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags for repository

The following `list-tags-for-resource` example displays a list of the tags associated with the `hello-world` repository.

```
aws ecr list-tags-for-resource \  
  --repository-name hello-world
```

```
--resource-arn arn:aws:ecr:us-west-2:012345678910:repository/hello-world
```

Output:

```
{
  "tags": [
    {
      "Key": "Stage",
      "Value": "Integ"
    }
  ]
}
```

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

put-image-scanning-configuration

The following code example shows how to use `put-image-scanning-configuration`.

AWS CLI

To update the image scanning configuration for a repository

The following `put-image-scanning-configuration` example updates the image scanning configuration for the specified repository.

```
aws ecr put-image-scanning-configuration \
  --repository-name sample-repo \
  --image-scanning-configuration scanOnPush=true
```

Output:

```
{
  "registryId": "012345678910",
  "repositoryName": "sample-repo",
  "imageScanningConfiguration": {
    "scanOnPush": true
  }
}
```

For more information, see [Image Scanning](#) in the *Amazon ECR User Guide*.

- For API details, see [PutImageScanningConfiguration](#) in *AWS CLI Command Reference*.

put-image-tag-mutability

The following code example shows how to use `put-image-tag-mutability`.

AWS CLI

To update the image tag mutability setting for a repository

The following `put-image-tag-mutability` example configures the specified repository for tag immutability. This prevents all image tags within the repository from being overwritten.

```
aws ecr put-image-tag-mutability \  
  --repository-name hello-repository \  
  --image-tag-mutability IMMUTABLE
```

Output:

```
{  
  "registryId": "012345678910",  
  "repositoryName": "sample-repo",  
  "imageTagMutability": "IMMUTABLE"  
}
```

For more information, see [Image Tag Mutability](#) in the *Amazon ECR User Guide*.

- For API details, see [PutImageTagMutability](#) in *AWS CLI Command Reference*.

put-image

The following code example shows how to use `put-image`.

AWS CLI

To retag an image with its manifest

The following `put-image` example creates a new tag in the `hello-world` repository with an existing image manifest.

```
aws ecr put-image \  
  --repository-name hello-world
```



```
--repository-name hello-world \  
--image-tag 2019.08 \  
--image-manifest file://hello-world.manifest.json
```

Contents of hello-world.manifest.json:

```
{  
  "schemaVersion": 2,  
  "mediaType": "application/vnd.docker.distribution.manifest.v2+json",  
  "config": {  
    "mediaType": "application/vnd.docker.container.image.v1+json",  
    "size": 5695,  
    "digest":  
"sha256:cea5fe7701b7db3dd1c372f3cea6f43cdda444fcc488f530829145e426d8b980"  
  },  
  "layers": [  
    {  
      "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",  
      "size": 39096921,  
      "digest":  
"sha256:d8868e50ac4c7104d2200d42f432b661b2da8c1e417ccfae217e6a1e04bb9295"  
    },  
    {  
      "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",  
      "size": 57938,  
      "digest":  
"sha256:83251ac64627fc331584f6c498b3aba5badc01574e2c70b2499af3af16630eed"  
    },  
    {  
      "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",  
      "size": 423,  
      "digest":  
"sha256:589bba2f1b36ae56f0152c246e2541c5aa604b058febfcf2be32e9a304fec610"  
    },  
    {  
      "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",  
      "size": 680,  
      "digest":  
"sha256:d62ecaceda3964b735cdd2af613d6bb136a52c1da0838b2ff4b4dab4212bcb1c"  
    },  
    {  
      "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",  
      "size": 162,
```

```

        "digest":
"sha256:6d93b41cfc6bf0d2522b7cf61588de4cd045065b36c52bd3aec2ba0622b2b22b"
    },
    {
        "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
        "size": 28268840,
        "digest":
"sha256:6986b4d4c07932c680b3587f2eac8b0e013568c003cc23b04044628a5c5e599f"
    },
    {
        "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
        "size": 35369152,
        "digest":
"sha256:8c5ec60f10102dc8da0649d866c7c2f706e459d0bdc25c83ad2de86f4996c276"
    },
    {
        "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
        "size": 155,
        "digest":
"sha256:cde50b1c594539c5f67cbede9aef95c9ae321ccfb857f7b251b45b84198adc85"
    },
    {
        "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
        "size": 28737,
        "digest":
"sha256:2e102807ab72a73fc9abf53e8c50e421bdc337a0a8afcb242176edeec65977e4"
    },
    {
        "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
        "size": 190,
        "digest":
"sha256:fc379bbd5ed37808772bef016553a297356c59b8f134659e6ee4ecb563c2f5a7"
    },
    {
        "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
        "size": 28748,
        "digest":
"sha256:021db240dfccf5a1aff19507d17c0177e5888e518acf295b52204b1825e8b7ee"
    }
    ]
}

```

Output:

```

{
  "image": {
    "registryId": "130757420319",
    "repositoryName": "hello-world",
    "imageId": {
      "imageDigest":
"sha256:8ece96b74f87652876199d83bd107d0435a196133af383ac54cb82b6cc5283ae",
      "imageTag": "2019.08"
    },
    "imageManifest": "{\n  \"schemaVersion\": 2,\n  \"mediaType
\": \"application/vnd.docker.distribution.manifest.v2+json
\", \n  \"config\": {\n    \"mediaType\": \"application/
vnd.docker.container.image.v1+json\", \n    \"size\": 5695, \n    \"digest\":
\"sha256:cea5fe7701b7db3dd1c372f3cea6f43cdda444fcc488f530829145e426d8b980\" \n
  }, \n  \"layers\": [\n    {\n      \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\", \n      \"size\": 39096921, \n      \"digest
\": \"sha256:d8868e50ac4c7104d2200d42f432b661b2da8c1e417ccfae217e6a1e04bb9295\" \n
    }, \n    {\n      \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\", \n      \"size\": 57938, \n      \"digest
\": \"sha256:83251ac64627fc331584f6c498b3aba5badc01574e2c70b2499af3af16630eed
\" \n    }, \n    {\n      \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\", \n      \"size\": 423, \n      \"digest\":
\"sha256:589bba2f1b36ae56f0152c246e2541c5aa604b058febfcf2be32e9a304fec610\" \n
    }, \n    {\n      \"mediaType\": \"application/vnd.docker.image.rootfs.diff.tar.gzip\",
\n      \"size\": 680, \n      \"digest\":
\"sha256:d62ecaceda3964b735cdd2af613d6bb136a52c1da0838b2ff4b4dab4212bcb1c
\" \n    }, \n    {\n      \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\", \n      \"size\": 162, \n      \"digest
\": \"sha256:6d93b41cfc6bf0d2522b7cf61588de4cd045065b36c52bd3aec2ba0622b2b22b
\" \n    }, \n    {\n      \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\", \n      \"size\": 28268840, \n      \"digest
\": \"sha256:6986b4d4c07932c680b3587f2eac8b0e013568c003cc23b04044628a5c5e599f
\" \n    }, \n    {\n      \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\", \n      \"size\": 35369152, \n      \"digest
\": \"sha256:8c5ec60f10102dc8da0649d866c7c2f706e459d0bdc25c83ad2de86f4996c276\" \n
    }, \n    {\n      \"mediaType\": \"application/
vnd.docker.image.rootfs.diff.tar.gzip\", \n      \"size\": 155, \n      \"digest\":
\"sha256:cde50b1c594539c5f67cbede9aef95c9ae321ccfb857f7b251b45b84198adc85\" \n
    }, \n    {\n      \"mediaType\": \"application/vnd.docker.image.rootfs.diff.tar.gzip\",
\n      \"size\": 28737, \n      \"digest\":
\"sha256:2e102807ab72a73fc9abf53e8c50e421bdc337a0a8afcb242176edeec65977e4\" \n
    }, \n    {\n      \"mediaType\": \"application/vnd.docker.image.rootfs.diff.tar.gzip\",
\n      \"size\": 190, \n      \"digest\":

```

```

  \"sha256:fc379bbd5ed37808772bef016553a297356c59b8f134659e6ee4ecb563c2f5a7\"\\n    },
\\n    {\\n      \"mediaType\": \"application/vnd.docker.image.rootfs.diff.tar.gzip\",
\\n      \"size\": 28748,\\n      \"digest\":
  \"sha256:021db240dfccf5a1aff19507d17c0177e5888e518acf295b52204b1825e8b7ee\"\\n
    }\\n  ]\\n}\\n\"
  }
}

```

- For API details, see [PutImage](#) in *AWS CLI Command Reference*.

put-lifecycle-policy

The following code example shows how to use `put-lifecycle-policy`.

AWS CLI

To create a lifecycle policy

The following `put-lifecycle-policy` example creates a lifecycle policy for the specified repository in the default registry for an account.

```

aws ecr put-lifecycle-policy \
  --repository-name "project-a/amazon-ecs-sample" \
  --lifecycle-policy-text "file://policy.json"

```

Contents of `policy.json`:

```

{
  "rules": [
    {
      "rulePriority": 1,
      "description": "Expire images older than 14 days",
      "selection": {
        "tagStatus": "untagged",
        "countType": "sinceImagePushed",
        "countUnit": "days",
        "countNumber": 14
      },
      "action": {
        "type": "expire"
      }
    }
  ]
}

```

```
]
}
```

Output:

```
{
  "registryId": "<aws_account_id>",
  "repositoryName": "project-a/amazon-ecs-sample",
  "lifecyclePolicyText": "{\"rules\": [{\"rulePriority\": 1, \"description\": \"Expire images older than 14 days\", \"selection\": {\"tagStatus\": \"untagged\", \"countType\": \"sinceImagePushed\", \"countUnit\": \"days\", \"countNumber\": 14}, \"action\": {\"type\": \"expire\"}}]}"
}
```

For more information, see [Lifecycle Policies](#) in the *Amazon ECR User Guide*.

- For API details, see [PutLifecyclePolicy](#) in *AWS CLI Command Reference*.

set-repository-policy

The following code example shows how to use `set-repository-policy`.

AWS CLI**To set the repository policy for a repository**

The following `set-repository-policy` example attaches a repository policy contained in a file to the `cluster-autoscaler` repository.

```
aws ecr set-repository-policy \
  --repository-name cluster-autoscaler \
  --policy-text file://my-policy.json
```

Contents of `my-policy.json`:

```
{
  "Version" : "2008-10-17",
  "Statement" : [
    {
      "Sid" : "allow public pull",
      "Effect" : "Allow",
      "Principal" : "*",
```

```

        "Action" : [
            "ecr:BatchCheckLayerAvailability",
            "ecr:BatchGetImage",
            "ecr:GetDownloadUrlForLayer"
        ]
    }
]
}

```

Output:

```

{
  "registryId": "012345678910",
  "repositoryName": "cluster-autoscaler",
  "policyText": "{\n  \"Version\" : \"2008-10-17\",\n  \"Statement\" : [ {\n\n    \"Sid\" : \"allow public pull\",\n    \"Effect\" : \"Allow\",\n    \"Principal\" :\n    \"*\",\n    \"Action\" : [ \"ecr:BatchCheckLayerAvailability\", \"ecr:BatchGetImage\n    \", \"ecr:GetDownloadUrlForLayer\" ]\n  } ]\n}"
}

```

- For API details, see [SetRepositoryPolicy](#) in *AWS CLI Command Reference*.

start-image-scan

The following code example shows how to use `start-image-scan`.

AWS CLI**To start an image vulnerability scan**

The following `start-image-scan` example starts an image scan for and specified by the image digest in the specified repository.

```

aws ecr start-image-scan \
  --repository-name sample-repo \
  --image-id
imageDigest=sha256:74b2c688c700ec95a93e478cdb959737c148df3fbf5ea706abe0318726e885e6

```

Output:

```

{

```

```
"registryId": "012345678910",
"repositoryName": "sample-repo",
"imageId": {
  "imageDigest":
"sha256:74b2c688c700ec95a93e478cdb959737c148df3fbf5ea706abe0318726e885e6"
},
"imageScanStatus": {
  "status": "IN_PROGRESS"
}
}
```

For more information, see [Image Scanning](#) in the *Amazon ECR User Guide*.

- For API details, see [StartImageScan](#) in *AWS CLI Command Reference*.

start-lifecycle-policy-preview

The following code example shows how to use start-lifecycle-policy-preview.

AWS CLI

To create a lifecycle policy preview

The following start-lifecycle-policy-preview example creates a lifecycle policy preview defined by a JSON file for the specified repository.

```
aws ecr start-lifecycle-policy-preview \
  --repository-name "project-a/amazon-ecs-sample" \
  --lifecycle-policy-text "file://policy.json"
```

Contents of policy.json:

```
{
  "rules": [
    {
      "rulePriority": 1,
      "description": "Expire images older than 14 days",
      "selection": {
        "tagStatus": "untagged",
        "countType": "sinceImagePushed",
        "countUnit": "days",
        "countNumber": 14
      }
    },
  ],
}
```

```

        "action": {
            "type": "expire"
        }
    ]
}

```

Output:

```

{
  "registryId": "012345678910",
  "repositoryName": "project-a/amazon-ecs-sample",
  "lifecyclePolicyText": "{\n  \"rules\": [\n    {\n      \"rulePriority\": 1,\n      \"description\": \"Expire images older than 14 days\",\n      \"selection\": {\n        \"tagStatus\": \"untagged\",\n        \"countType\": \"sinceImagePushed\",\n        \"countUnit\": \"days\",\n        \"countNumber\": 14\n      },\n      \"action\": {\n        \"type\": \"expire\"\n      }\n    }\n  ]\n}\n",
  "status": "IN_PROGRESS"
}

```

- For API details, see [StartLifecyclePolicyPreview](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To tag a repository

The following tag-resource example sets a tag with key Stage and value Integ on the hello-world repository.

```

aws ecr tag-resource \
  --resource-arn arn:aws:ecr:us-west-2:012345678910:repository/hello-world \
  --tags Key=Stage,Value=Integ

```

This command produces no output.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To untag a repository

The following `untag-resource` example removes the tag with the key `Stage` from the `hello-world` repository.

```
aws ecr untag-resource \  
  --resource-arn arn:aws:ecr:us-west-2:012345678910:repository/hello-world \  
  --tag-keys Stage
```

This command produces no output.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

upload-layer-part

The following code example shows how to use `upload-layer-part`.

AWS CLI

To upload a layer part

This following `upload-layer-part` uploads an image layer part to the `layer-test` repository.

```
aws ecr upload-layer-part \  
  --repository-name layer-test \  
  --upload-id 6cb64b8a-9378-0e33-2ab1-b780fab8a9e9 \  
  --part-first-byte 0 \  
  --part-last-byte 8323314 \  
  --layer-part-blob file:///var/lib/docker/image/overlay2/layerdb/sha256/  
ff986b10a018b48074e6d3a68b39aad8ccc002cdad912d4148c0f92b3729323e/layer.b64
```

Output:

```
{  
  "uploadId": "6cb64b8a-9378-0e33-2ab1-b780fab8a9e9",
```

```
"registryId": "012345678910",  
"lastByteReceived": 8323314,  
"repositoryName": "layer-test"  
}
```

- For API details, see [UploadLayerPart](#) in *AWS CLI Command Reference*.

Amazon ECS examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon ECS.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-capacity-provider

The following code example shows how to use `create-capacity-provider`.

AWS CLI

To create a capacity provider

The following `create-capacity-provider` example creates a capacity provider that uses an Auto Scaling group named `MyASG`, has managed scaling and managed termination protection enabled. This configuration is used for Amazon ECS cluster auto scaling.

```
aws ecs create-capacity-provider \
```

```
--name "MyCapacityProvider" \
--auto-scaling-group-provider "autoScalingGroupArn=arn:aws:autoscaling:us-
east-1:123456789012:autoScalingGroup:57ffcb94-11f0-4d6d-
bf60-3bac5EXAMPLE:autoScalingGroupName/
MyASG,managedScaling={status=ENABLED,targetCapacity=100},managedTerminationProtection=ENABLED"
```

Output:

```
{
  "capacityProvider": {
    "capacityProviderArn": "arn:aws:ecs:us-east-1:123456789012:capacity-provider/
MyCapacityProvider",
    "name": "MyCapacityProvider",
    "status": "ACTIVE",
    "autoScalingGroupProvider": {
      "autoScalingGroupArn": "arn:aws:autoscaling:us-
east-1:123456789012:autoScalingGroup:57ffcb94-11f0-4d6d-
bf60-3bac5EXAMPLE:autoScalingGroupName/MyASG",
      "managedScaling": {
        "status": "ENABLED",
        "targetCapacity": 100,
        "minimumScalingStepSize": 1,
        "maximumScalingStepSize": 10000,
        "instanceWarmupPeriod": 300
      },
      "managedTerminationProtection": "ENABLED"
    },
    "tags": []
  }
}
```

For more information, see [Amazon ECS cluster auto scaling](#) in the *Amazon ECS Developer Guide*.

- For API details, see [CreateCapacityProvider](#) in *AWS CLI Command Reference*.

create-cluster

The following code example shows how to use `create-cluster`.

AWS CLI

Example 1: To create a new cluster

The following `create-cluster` example creates a cluster.

```
aws ecs create-cluster \  
  --cluster-name MyCluster
```

Output:

```
{  
  "cluster": {  
    "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",  
    "clusterName": "MyCluster",  
    "status": "ACTIVE",  
    "registeredContainerInstancesCount": 0,  
    "pendingTasksCount": 0,  
    "runningTasksCount": 0,  
    "activeServicesCount": 0,  
    "statistics": [],  
    "tags": []  
  }  
}
```

For more information, see [Creating a Cluster](#) in the *Amazon ECS Developer Guide*.

Example 2: To create a new cluster using capacity providers

The following `create-cluster` example creates a cluster and associates two existing capacity providers with it. The `create-capacity-provider` command is used to create a capacity provider. Specifying a default capacity provider strategy is optional, but recommended. In this example, we create a cluster named `MyCluster` and associate the `MyCapacityProvider1` and `MyCapacityProvider2` capacity providers with it. A default capacity provider strategy is specified that spreads the tasks evenly across both capacity providers.

```
aws ecs create-cluster --cluster-name MyCluster --capacity-providers  
MyCapacityProvider1 MyCapacityProvider2 --default-capacity-  
provider-strategy capacityProvider=MyCapacityProvider1,weight=1  
capacityProvider=MyCapacityProvider2,weight=1
```

Output:

```
{  
  "cluster": {  
    "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",  
    "clusterName": "MyCluster",
```

```
"status": "PROVISIONING",
"registeredContainerInstancesCount": 0,
"pendingTasksCount": 0,
"runningTasksCount": 0,
"activeServicesCount": 0,
"statistics": [],
"settings": [
  {
    "name": "containerInsights",
    "value": "enabled"
  }
],
"capacityProviders": [
  "MyCapacityProvider1",
  "MyCapacityProvider2"
],
"defaultCapacityProviderStrategy": [
  {
    "capacityProvider": "MyCapacityProvider1",
    "weight": 1,
    "base": 0
  },
  {
    "capacityProvider": "MyCapacityProvider2",
    "weight": 1,
    "base": 0
  }
],
"attachments": [
  {
    "id": "0fb0c8f4-6edd-4de1-9b09-17e470ee1918",
    "type": "asp",
    "status": "PRECREATED",
    "details": [
      {
        "name": "capacityProviderName",
        "value": "MyCapacityProvider1"
      },
      {
        "name": "scalingPlanName",
        "value": "ECManagedAutoScalingPlan-a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
      }
    ]
  }
]
```

```

    },
    {
      "id": "ae592060-2382-4663-9476-b015c685593c",
      "type": "asp",
      "status": "PRECREATED",
      "details": [
        {
          "name": "capacityProviderName",
          "value": "MyCapacityProvider2"
        },
        {
          "name": "scalingPlanName",
          "value": "ECManagedAutoScalingPlan-a1b2c3d4-5678-90ab-cdef-
EXAMPLE22222"
        }
      ]
    }
  ],
  "attachmentsStatus": "UPDATE_IN_PROGRESS"
}

```

For more information, see [Cluster capacity providers](#) in the *Amazon ECS Developer Guide*.

Example 3: To create a new cluster with multiple tags

The following `create-cluster` example creates a cluster with multiple tags. For more information about adding tags using shorthand syntax, see [Using Shorthand Syntax with the AWS Command Line Interface](#) in the *AWS CLI User Guide*.

```

aws ecs create-cluster \
  --cluster-name MyCluster \
  --tags key=key1,value=value1 key=key2,value=value2 key=key3,value=value3

```

Output:

```

{
  "cluster": {
    "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",
    "clusterName": "MyCluster",
    "status": "ACTIVE",
    "registeredContainerInstancesCount": 0,
    "pendingTasksCount": 0,

```

```
    "runningTasksCount": 0,
    "activeServicesCount": 0,
    "statistics": [],
    "tags": [
      {
        "key": "key1",
        "value": "value1"
      },
      {
        "key": "key2",
        "value": "value2"
      },
      {
        "key": "key3",
        "value": "value3"
      }
    ]
  }
}
```

For more information, see [Creating a Cluster](#) in the *Amazon ECS Developer Guide*.

- For API details, see [CreateCluster](#) in *AWS CLI Command Reference*.

create-service

The following code example shows how to use `create-service`.

AWS CLI

Example 1: To create a service with a Fargate task

The following `create-service` example shows how to create a service using a Fargate task.

```
aws ecs create-service \
  --cluster MyCluster \
  --service-name MyService \
  --task-definition sample-fargate:1 \
  --desired-count 2 \
  --launch-type FARGATE \
  --platform-version LATEST \
  --network-configuration
  "awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321],assignPublicIp
  \
```

```
--tags key=key1,value=value1 key=key2,value=value2 key=key3,value=value3
```

Output:

```
{
  "service": {
    "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/MyCluster/MyService",
    "serviceName": "MyService",
    "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",
    "loadBalancers": [],
    "serviceRegistries": [],
    "status": "ACTIVE",
    "desiredCount": 2,
    "runningCount": 0,
    "pendingCount": 0,
    "launchType": "FARGATE",
    "platformVersion": "LATEST",
    "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/sample-fargate:1",
    "deploymentConfiguration": {
      "maximumPercent": 200,
      "minimumHealthyPercent": 100
    },
    "deployments": [
      {
        "id": "ecs-svc/1234567890123456789",
        "status": "PRIMARY",
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/sample-fargate:1",
        "desiredCount": 2,
        "pendingCount": 0,
        "runningCount": 0,
        "createdAt": 1557119253.821,
        "updatedAt": 1557119253.821,
        "launchType": "FARGATE",
        "platformVersion": "1.3.0",
        "networkConfiguration": {
          "awsvpcConfiguration": {
            "subnets": [
              "subnet-12344321"
            ],
            "securityGroups": [
```



```
        "sg-12344321"
      ],
      "assignPublicIp": "ENABLED"
    }
  }
},
"roleArn": "arn:aws:iam::123456789012:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
"events": [],
"createdAt": 1557119253.821,
"placementConstraints": [],
"placementStrategy": [],
"networkConfiguration": {
  "awsvpcConfiguration": {
    "subnets": [
      "subnet-12344321"
    ],
    "securityGroups": [
      "sg-12344321"
    ],
    "assignPublicIp": "ENABLED"
  }
},
"schedulingStrategy": "REPLICA",
"tags": [
  {
    "key": "key1",
    "value": "value1"
  },
  {
    "key": "key2",
    "value": "value2"
  },
  {
    "key": "key3",
    "value": "value3"
  }
],
"enableECSManagedTags": false,
"propagateTags": "NONE"
}
```

Example 2: To create a service using the EC2 launch type

The following `create-service` example shows how to create a service called `ecs-simple-service` with a task that uses the EC2 launch type. The service uses the `sleep360` task definition and it maintains 1 instantiation of the task.

```
aws ecs create-service \  
  --cluster MyCluster \  
  --service-name ecs-simple-service \  
  --task-definition sleep360:2 \  
  --desired-count 1
```

Output:

```
{  
  "service": {  
    "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/MyCluster/ecs-  
simple-service",  
    "serviceName": "ecs-simple-service",  
    "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",  
    "loadBalancers": [],  
    "serviceRegistries": [],  
    "status": "ACTIVE",  
    "desiredCount": 1,  
    "runningCount": 0,  
    "pendingCount": 0,  
    "launchType": "EC2",  
    "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/  
sleep360:2",  
    "deploymentConfiguration": {  
      "maximumPercent": 200,  
      "minimumHealthyPercent": 100  
    },  
    "deployments": [  
      {  
        "id": "ecs-svc/1234567890123456789",  
        "status": "PRIMARY",  
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-  
definition/sleep360:2",  
        "desiredCount": 1,  
        "pendingCount": 0,  
        "runningCount": 0,  
        "createdAt": 1557206498.798,  
        "desiredStatus": "PRIMARY",  
        "desiredStatusReason": "The service is in the ACTIVE state and the  
deployment is the primary deployment.",  
        "pendingStatus": "PENDING",  
        "pendingStatusReason": "The service is in the ACTIVE state and the  
deployment is the primary deployment.",  
        "runningStatus": "RUNNING",  
        "runningStatusReason": "The service is in the ACTIVE state and the  
deployment is the primary deployment."
```

```

        "updatedAt": 1557206498.798,
        "launchType": "EC2"
    }
],
"events": [],
"createdAt": 1557206498.798,
"placementConstraints": [],
"placementStrategy": [],
"schedulingStrategy": "REPLICA",
"enableECSTags": false,
"propagateTags": "NONE"
}
}

```

Example 3: To create a service that uses an external deployment controller

The following `create-service` example creates a service that uses an external deployment controller.

```

aws ecs create-service \
  --cluster MyCluster \
  --service-name MyService \
  --deployment-controller type=EXTERNAL \
  --desired-count 1

```

Output:

```

{
  "service": {
    "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/MyCluster/MyService",
    "serviceName": "MyService",
    "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",
    "loadBalancers": [],
    "serviceRegistries": [],
    "status": "ACTIVE",
    "desiredCount": 1,
    "runningCount": 0,
    "pendingCount": 0,
    "launchType": "EC2",
    "deploymentConfiguration": {
      "maximumPercent": 200,
      "minimumHealthyPercent": 100
    }
  }
}

```

```

    },
    "taskSets": [],
    "deployments": [],
    "roleArn": "arn:aws:iam::123456789012:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
    "events": [],
    "createdAt": 1557128207.101,
    "placementConstraints": [],
    "placementStrategy": [],
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
      "type": "EXTERNAL"
    },
    "enableECSManagedTags": false,
    "propagateTags": "NONE"
  }
}

```

Example 4: To create a new service behind a load balancer

The following `create-service` example shows how to create a service that is behind a load balancer. You must have a load balancer configured in the same Region as your container instance. This example uses the `--cli-input-json` option and a JSON input file called `ecs-simple-service-elb.json` with the following content:

```

{
  "serviceName": "ecs-simple-service-elb",
  "taskDefinition": "ecs-demo",
  "loadBalancers": [
    {
      "loadBalancerName": "EC2Contai-EcsElast-123456789012",
      "containerName": "simple-demo",
      "containerPort": 80
    }
  ],
  "desiredCount": 10,
  "role": "ecsServiceRole"
}

```

Command:

```
aws ecs create-service \
```

```
--cluster MyCluster \  
--service-name ecs-simple-service-elb \  
--cli-input-json file://ecs-simple-service-elb.json
```

Output:

```
{  
  "service": {  
    "status": "ACTIVE",  
    "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/ecs-  
demo:1",  
    "pendingCount": 0,  
    "loadBalancers": [  
      {  
        "containerName": "ecs-demo",  
        "containerPort": 80,  
        "loadBalancerName": "EC2Contai-EcsElast-123456789012"  
      }  
    ],  
    "roleArn": "arn:aws:iam::123456789012:role/ecsServiceRole",  
    "desiredCount": 10,  
    "serviceName": "ecs-simple-service-elb",  
    "clusterArn": "arn:aws:ecs:<us-west-2:123456789012:cluster/MyCluster",  
    "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/ecs-simple-  
service-elb",  
    "deployments": [  
      {  
        "status": "PRIMARY",  
        "pendingCount": 0,  
        "createdAt": 1428100239.123,  
        "desiredCount": 10,  
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-  
definition/ecs-demo:1",  
        "updatedAt": 1428100239.123,  
        "id": "ecs-svc/1234567890123456789",  
        "runningCount": 0  
      }  
    ],  
    "events": [],  
    "runningCount": 0  
  }  
}
```

For more information, see [Creating a Service](#) in the *Amazon ECS Developer Guide*.

- For API details, see [CreateService](#) in *AWS CLI Command Reference*.

create-task-set

The following code example shows how to use create-task-set.

AWS CLI

To create a task set

The following create-task-set example creates a task set in a service that uses an external deployment controller.

```
aws ecs create-task-set \  
  --cluster MyCluster \  
  --service MyService \  
  --task-definition MyTaskDefinition:2 \  
  --network-configuration  
  "awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321]}"
```

Output:

```
{  
  "taskSet": {  
    "id": "ecs-svc/1234567890123456789",  
    "taskSetArn": "arn:aws:ecs:us-west-2:123456789012:task-set/MyCluster/  
MyService/ecs-svc/1234567890123456789",  
    "status": "ACTIVE",  
    "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/  
MyTaskDefinition:2",  
    "computedDesiredCount": 0,  
    "pendingCount": 0,  
    "runningCount": 0,  
    "createdAt": 1557128360.711,  
    "updatedAt": 1557128360.711,  
    "launchType": "EC2",  
    "networkConfiguration": {  
      "awsvpcConfiguration": {  
        "subnets": [  
          "subnet-12344321"  
        ],  
      },  
    },  
  },  
}
```

```

        "securityGroups": [
            "sg-12344321"
        ],
        "assignPublicIp": "DISABLED"
    }
},
"loadBalancers": [],
"serviceRegistries": [],
"scale": {
    "value": 0.0,
    "unit": "PERCENT"
},
"stabilityStatus": "STABILIZING",
"stabilityStatusAt": 1557128360.711
}
}

```

- For API details, see [CreateTaskSet](#) in *AWS CLI Command Reference*.

delete-account-setting

The following code example shows how to use `delete-account-setting`.

AWS CLI

To delete the account settings for a specific IAM user or IAM role

The following example `delete-account-setting` deletes the account settings for the specific IAM user or IAM role.

```

aws ecs delete-account-setting \
  --name serviceLongArnFormat \
  --principal-arn arn:aws:iam::123456789012:user/MyUser

```

Output:

```

{
  "setting": {
    "name": "serviceLongArnFormat",
    "value": "enabled",
    "principalArn": "arn:aws:iam::123456789012:user/MyUser"
  }
}

```

```
}
```

For more information, see [Amazon Resource Names \(ARNs\) and IDs](#) in the *Amazon ECS Developer Guide*.

- For API details, see [DeleteAccountSetting](#) in *AWS CLI Command Reference*.

delete-attributes

The following code example shows how to use `delete-attributes`.

AWS CLI

To delete one or more custom attributes from an Amazon ECS resource

The following `delete-attributes` deletes an attribute with the name `stack` from a container instance.

```
aws ecs delete-attributes \  
  --attributes name=stack,targetId=arn:aws:ecs:us-west-2:130757420319:container-  
instance/1c3be8ed-df30-47b4-8f1e-6e68ebd01f34
```

Output:

```
{  
  "attributes": [  
    {  
      "name": "stack",  
      "targetId": "arn:aws:ecs:us-west-2:130757420319:container-  
instance/1c3be8ed-df30-47b4-8f1e-6e68ebd01f34",  
      "value": "production"  
    }  
  ]  
}
```

- For API details, see [DeleteAttributes](#) in *AWS CLI Command Reference*.

delete-capacity-provider

The following code example shows how to use `delete-capacity-provider`.

AWS CLI

Example 1: To delete a capacity provider using the Amazon Resource Name (ARN)

The following `delete-capacity-provider` example deletes a capacity provider by specifying the Amazon Resource Name (ARN) of the capacity provider. The ARN as well as the status of the capacity provider deletion can be retrieved using the `describe-capacity-providers` command.

```
aws ecs delete-capacity-provider \  
  --capacity-provider arn:aws:ecs:us-west-2:123456789012:capacity-provider/  
  ExampleCapacityProvider
```

Output:

```
{  
  "capacityProvider": {  
    "capacityProviderArn": "arn:aws:ecs:us-west-2:123456789012:capacity-  
provider/ExampleCapacityProvider",  
    "name": "ExampleCapacityProvider",  
    "status": "ACTIVE",  
    "autoScalingGroupProvider": {  
      "autoScalingGroupArn": "arn:aws:autoscaling:us-  
west-2:123456789012:autoScalingGroup:a1b2c3d4-5678-90ab-cdef-  
EXAMPLE11111:autoScalingGroupName/MyAutoScalingGroup",  
      "managedScaling": {  
        "status": "ENABLED",  
        "targetCapacity": 100,  
        "minimumScalingStepSize": 1,  
        "maximumScalingStepSize": 10000  
      },  
      "managedTerminationProtection": "DISABLED"  
    },  
    "updateStatus": "DELETE_IN_PROGRESS",  
    "tags": []  
  }  
}
```

For more information, see [Cluster capacity providers](#) in the *Amazon ECS Developer Guide*.

Example 2: To delete a capacity provider using the name

The following `delete-capacity-provider` example deletes a capacity provider by specifying the short name of the capacity provider. The short name as well as the status of the capacity provider deletion can be retrieved using the `describe-capacity-providers` command.

```
aws ecs delete-capacity-provider \  
  --capacity-provider ExampleCapacityProvider
```

Output:

```
{  
  "capacityProvider": {  
    "capacityProviderArn": "arn:aws:ecs:us-west-2:123456789012:capacity-  
provider/ExampleCapacityProvider",  
    "name": "ExampleCapacityProvider",  
    "status": "ACTIVE",  
    "autoScalingGroupProvider": {  
      "autoScalingGroupArn": "arn:aws:autoscaling:us-  
west-2:123456789012:autoScalingGroup:a1b2c3d4-5678-90ab-cdef-  
EXAMPLE11111:autoScalingGroupName/MyAutoScalingGroup",  
      "managedScaling": {  
        "status": "ENABLED",  
        "targetCapacity": 100,  
        "minimumScalingStepSize": 1,  
        "maximumScalingStepSize": 10000  
      },  
      "managedTerminationProtection": "DISABLED"  
    },  
    "updateStatus": "DELETE_IN_PROGRESS",  
    "tags": []  
  }  
}
```

For more information, see [Cluster capacity providers](#) in the *Amazon ECS Developer Guide*.

- For API details, see [DeleteCapacityProvider](#) in *AWS CLI Command Reference*.

delete-cluster

The following code example shows how to use `delete-cluster`.

AWS CLI

To delete an empty cluster

The following `delete-cluster` example deletes the specified empty cluster.

```
aws ecs delete-cluster --cluster MyCluster
```

Output:

```
{
  "cluster": {
    "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",
    "status": "INACTIVE",
    "clusterName": "MyCluster",
    "registeredContainerInstancesCount": 0,
    "pendingTasksCount": 0,
    "runningTasksCount": 0,
    "activeServicesCount": 0
    "statistics": [],
    "tags": []
  }
}
```

For more information, see [Deleting a Cluster](#) in the *Amazon ECS Developer Guide*.

- For API details, see [DeleteCluster](#) in *AWS CLI Command Reference*.

delete-service

The following code example shows how to use `delete-service`.

AWS CLI

To delete a service

The following `ecs delete-service` example deletes the specified service from a cluster. You can include the `--force` parameter to delete a service even if it has not been scaled to zero tasks.

```
aws ecs delete-service --cluster MyCluster --service MyService1 --force
```

For more information, see [Deleting a Service](#) in the *Amazon ECS Developer Guide*.

- For API details, see [DeleteService](#) in *AWS CLI Command Reference*.

delete-task-definitions

The following code example shows how to use `delete-task-definitions`.

AWS CLI

To delete a task definition

The following `delete-task-definitions` example deletes an INACTIVE task definition.

```
aws ecs delete-task-definitions \  
  --task-definition curltest:1
```

Output:

```
{  
  "taskDefinitions": [  
    {  
      "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/  
curltest:1",  
      "containerDefinitions": [  
        {  
          "name": "ctest",  
          "image": "mreferre/eksutils",  
          "cpu": 0,  
          "portMappings": [],  
          "essential": true,  
          "entryPoint": [  
            "sh",  
            "-c"  
          ],  
          "command": [  
            "curl ${ECS_CONTAINER_METADATA_URI_V4}/task"  
          ],  
          "environment": [],  
          "mountPoints": [],  
          "volumesFrom": [],  
          "logConfiguration": {  
            "logDriver": "awslogs",
```

```

        "options": {
            "awslogs-create-group": "true",
            "awslogs-group": "/ecs/curltest",
            "awslogs-region": "us-east-1",
            "awslogs-stream-prefix": "ecs"
        }
    },
    ],
    "family": "curltest",
    "taskRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
    "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
    "networkMode": "awsvpc",
    "revision": 1,
    "volumes": [],
    "status": "DELETE_IN_PROGRESS",
    "compatibilities": [
        "EC2",
        "FARGATE"
    ],
    "requiresCompatibilities": [
        "FARGATE"
    ],
    "cpu": "256",
    "memory": "512",
    "registeredAt": "2021-09-10T12:56:24.704000+00:00",
    "deregisteredAt": "2023-03-14T15:20:59.419000+00:00",
    "registeredBy": "arn:aws:sts::123456789012:assumed-role/Admin/jdoe"
    }
    ],
    "failures": []
}

```

For more information, see [Amazon ECS Task Definitions](#) in the *Amazon ECS Developer Guide*.

- For API details, see [DeleteTaskDefinitions](#) in *AWS CLI Command Reference*.

delete-task-set

The following code example shows how to use `delete-task-set`.

AWS CLI

To delete a task set

The following `delete-task-set` example shows how to delete a task set. You can include the `--force` parameter to delete a task set even if it has not been scaled to zero.

```
aws ecs delete-task-set \  
  --cluster MyCluster \  
  --service MyService \  
  --task-set arn:aws:ecs:us-west-2:123456789012:task-set/MyCluster/MyService/ecs-  
svc/1234567890123456789 \  
  --force
```

Output:

```
{  
  "taskSet": {  
    "id": "ecs-svc/1234567890123456789",  
    "taskSetArn": "arn:aws:ecs:us-west-2:123456789012:task-set/MyCluster/  
MyService/ecs-svc/1234567890123456789",  
    "status": "DRAINING",  
    "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/  
sample-fargate:2",  
    "computedDesiredCount": 0,  
    "pendingCount": 0,  
    "runningCount": 0,  
    "createdAt": 1557130260.276,  
    "updatedAt": 1557130290.707,  
    "launchType": "EC2",  
    "networkConfiguration": {  
      "awsvpcConfiguration": {  
        "subnets": [  
          "subnet-12345678"  
        ],  
        "securityGroups": [  
          "sg-12345678"  
        ],  
        "assignPublicIp": "DISABLED"  
      }  
    },  
    "loadBalancers": [],  
    "serviceRegistries": [],  
    "scale": {  
      "value": 0.0,  
      "unit": "PERCENT"  
    }  
  },  
}
```

```

    "stabilityStatus": "STABILIZING",
    "stabilityStatusAt": 1557130290.707
  }
}

```

- For API details, see [DeleteTaskSet](#) in *AWS CLI Command Reference*.

deregister-container-instance

The following code example shows how to use `deregister-container-instance`.

AWS CLI

To deregister a container instance from a cluster

The following `deregister-container-instance` example deregisters a container instance from the specified cluster. If there are still tasks running in the container instance, you must either stop those tasks before deregistering, or use the `--force` option.

```

aws ecs deregister-container-instance \
  --cluster arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster \
  --container-instance arn:aws:ecs:us-west-2:123456789012:container-instance/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \
  --force

```

Output:

```

{
  "containerInstance": {
    "remainingResources": [
      {
        "integerValue": 1024,
        "doubleValue": 0.0,
        "type": "INTEGER",
        "longValue": 0,
        "name": "CPU"
      },
      {
        "integerValue": 985,
        "doubleValue": 0.0,
        "type": "INTEGER",
        "longValue": 0,

```

```
        "name": "MEMORY"
      },
      {
        "type": "STRINGSET",
        "integerValue": 0,
        "name": "PORTS",
        "stringSetValue": [
          "22",
          "2376",
          "2375",
          "51678",
          "51679"
        ],
        "longValue": 0,
        "doubleValue": 0.0
      },
      {
        "type": "STRINGSET",
        "integerValue": 0,
        "name": "PORTS_UDP",
        "stringSetValue": [],
        "longValue": 0,
        "doubleValue": 0.0
      }
    ],
    "agentConnected": true,
    "attributes": [
      {
        "name": "ecs.capability.secrets.asm.environment-variables"
      },
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.syslog"
      },
      {
        "value": "ami-01a82c3fce2c3ba58",
        "name": "ecs.ami-id"
      },
      {
        "name": "ecs.capability.secrets.asm.bootstrap.log-driver"
      },
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.none"
      }
    ]
  }
```



```
    "name": "ecs.capability.ecr-endpoint"
  },
  {
    "name": "com.amazonaws.ecs.capability.logging-driver.json-file"
  },
  {
    "value": "vpc-1234567890123467",
    "name": "ecs.vpc-id"
  },
  {
    "name": "ecs.capability.execution-role-awslogs"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.17"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.18"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.19"
  },
  {
    "name": "ecs.capability.docker-plugin.local"
  },
  {
    "name": "ecs.capability.task-eni"
  },
  {
    "name": "ecs.capability.task-cpu-mem-limit"
  },
  {
    "name": "ecs.capability.secrets.ssm.bootstrap.log-driver"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.30"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.31"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.32"
  },
  {
    "name": "ecs.capability.execution-role-ecr-pull"
```

```
},
{
  "name": "ecs.capability.container-health-check"
},
{
  "value": "subnet-1234567890123467",
  "name": "ecs.subnet-id"
},
{
  "value": "us-west-2a",
  "name": "ecs.availability-zone"
},
{
  "value": "t2.micro",
  "name": "ecs.instance-type"
},
{
  "name": "com.amazonaws.ecs.capability.task-iam-role-network-host"
},
{
  "name": "ecs.capability.aws-appmesh"
},
{
  "name": "com.amazonaws.ecs.capability.logging-driver.awslogs"
},
{
  "name": "com.amazonaws.ecs.capability.docker-remote-api.1.24"
},
{
  "name": "com.amazonaws.ecs.capability.docker-remote-api.1.25"
},
{
  "name": "com.amazonaws.ecs.capability.docker-remote-api.1.26"
},
{
  "name": "com.amazonaws.ecs.capability.docker-remote-api.1.27"
},
{
  "name": "com.amazonaws.ecs.capability.privileged-container"
},
{
  "name": "ecs.capability.container-ordering"
},
{
```

```
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.28"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.29"
  },
  {
    "value": "x86_64",
    "name": "ecs.cpu-architecture"
  },
  {
    "value": "93f43776-2018.10.0",
    "name": "ecs.capability.cni-plugin-version"
  },
  {
    "name": "ecs.capability.secrets.ssm.environment-variables"
  },
  {
    "name": "ecs.capability.pid-ipc-namespace-sharing"
  },
  {
    "name": "com.amazonaws.ecs.capability.ecr-auth"
  },
  {
    "value": "linux",
    "name": "ecs.os-type"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.20"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.21"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.22"
  },
  {
    "name": "ecs.capability.task-eia"
  },
  {
    "name": "ecs.capability.private-registry-
authentication.secretsmanager"
  },
  {
    "name": "com.amazonaws.ecs.capability.task-iam-role"
```

```
    },
    {
      "name": "com.amazonaws.ecs.capability.docker-remote-api.1.23"
    }
  ],
  "pendingTasksCount": 0,
  "tags": [],
  "containerInstanceArn": "arn:aws:ecs:us-west-2:123456789012:container-
instance/a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
  "registeredResources": [
    {
      "integerValue": 1024,
      "doubleValue": 0.0,
      "type": "INTEGER",
      "longValue": 0,
      "name": "CPU"
    },
    {
      "integerValue": 985,
      "doubleValue": 0.0,
      "type": "INTEGER",
      "longValue": 0,
      "name": "MEMORY"
    },
    {
      "type": "STRINGSET",
      "integerValue": 0,
      "name": "PORTS",
      "stringSetValue": [
        "22",
        "2376",
        "2375",
        "51678",
        "51679"
      ],
      "longValue": 0,
      "doubleValue": 0.0
    },
    {
      "type": "STRINGSET",
      "integerValue": 0,
      "name": "PORTS_UDP",
      "stringSetValue": [],
      "longValue": 0,
```

```
        "doubleValue": 0.0
      }
    ],
    "status": "INACTIVE",
    "registeredAt": 1557768075.681,
    "version": 4,
    "versionInfo": {
      "agentVersion": "1.27.0",
      "agentHash": "aabe65ee",
      "dockerVersion": "DockerVersion: 18.06.1-ce"
    },
    "attachments": [],
    "runningTasksCount": 0,
    "ec2InstanceId": "i-12345678901234678"
  }
}
```

For more information, see [Deregister a Container Instance](#) in the *ECS Developer Guide*.

- For API details, see [DeregisterContainerInstance](#) in *AWS CLI Command Reference*.

deregister-task-definition

The following code example shows how to use `deregister-task-definition`.

AWS CLI

To deregister a task definition

The following `deregister-task-definition` example deregisters the first revision of the `curler` task definition in your default region.

```
aws ecs deregister-task-definition --task-definition curler:1
```

Note that in the resulting output, the task definition status shows `INACTIVE`:

```
{
  "taskDefinition": {
    "status": "INACTIVE",
    "family": "curler",
    "volumes": [],
    "taskDefinitionArn": "arn:aws:ecs:us-west-2:123456789012:task-definition/
curler:1",
```

```
    "containerDefinitions": [
      {
        "environment": [],
        "name": "curler",
        "mountPoints": [],
        "image": "curl:latest",
        "cpu": 100,
        "portMappings": [],
        "entryPoint": [],
        "memory": 256,
        "command": [
          "curl -v http://example.com/"
        ],
        "essential": true,
        "volumesFrom": []
      }
    ],
    "revision": 1
  }
}
```

For more information, see [Amazon ECS Task Definitions](#) in the *Amazon ECS Developer Guide*.

- For API details, see [DeregisterTaskDefinition](#) in *AWS CLI Command Reference*.

describe-capacity-providers

The following code example shows how to use describe-capacity-providers.

AWS CLI

Example 1: To describe all capacity providers

The following describe-capacity-providers example retrieves details about all capacity providers.

```
aws ecs describe-capacity-providers
```

Output:

```
{
  "capacityProviders": [
```

```

    {
      "capacityProviderArn": "arn:aws:ecs:us-west-2:123456789012:capacity-
provider/MyCapacityProvider",
      "name": "MyCapacityProvider",
      "status": "ACTIVE",
      "autoScalingGroupProvider": {
        "autoScalingGroupArn": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111:autoScalingGroupName/MyAutoScalingGroup",
        "managedScaling": {
          "status": "ENABLED",
          "targetCapacity": 100,
          "minimumScalingStepSize": 1,
          "maximumScalingStepSize": 1000
        },
        "managedTerminationProtection": "ENABLED"
      },
      "tags": []
    },
    {
      "capacityProviderArn": "arn:aws:ecs:us-west-2:123456789012:capacity-
provider/FARGATE",
      "name": "FARGATE",
      "status": "ACTIVE",
      "tags": []
    },
    {
      "capacityProviderArn": "arn:aws:ecs:us-west-2:123456789012:capacity-
provider/FARGATE_SPOT",
      "name": "FARGATE_SPOT",
      "status": "ACTIVE",
      "tags": []
    }
  ]
}

```

For more information, see [Cluster capacity providers](#) in the *Amazon ECS Developer Guide*.

Example 2: To describe a specific capacity providers

The following `describe-capacity-providers` example retrieves details about a specific capacity provider. Using the `--include TAGS` parameter will add the tags associated with the capacity provider to the output.

```
aws ecs describe-capacity-providers \  
  --capacity-providers MyCapacityProvider \  
  --include TAGS
```

Output:

```
{  
  "capacityProviders": [  
    {  
      "capacityProviderArn": "arn:aws:ecs:us-west-2:123456789012:capacity-  
provider/MyCapacityProvider",  
      "name": "MyCapacityProvider",  
      "status": "ACTIVE",  
      "autoScalingGroupProvider": {  
        "autoScalingGroupArn": "arn:aws:autoscaling:us-  
west-2:123456789012:autoScalingGroup:a1b2c3d4-5678-90ab-cdef-  
EXAMPLE11111:autoScalingGroupName/MyAutoScalingGroup",  
        "managedScaling": {  
          "status": "ENABLED",  
          "targetCapacity": 100,  
          "minimumScalingStepSize": 1,  
          "maximumScalingStepSize": 1000  
        },  
        "managedTerminationProtection": "ENABLED"  
      },  
      "tags": [  
        {  
          "key": "environment",  
          "value": "production"  
        }  
      ]  
    }  
  ]  
}
```

For more information, see [Cluster capacity providers](#) in the *Amazon ECS Developer Guide*.

- For API details, see [DescribeCapacityProviders](#) in *AWS CLI Command Reference*.

describe-clusters

The following code example shows how to use `describe-clusters`.

AWS CLI

Example 1: To describe a cluster

The following `describe-clusters` example retrieves details about the specified cluster.

```
aws ecs describe-clusters \  
  --cluster default
```

Output:

```
{  
  "clusters": [  
    {  
      "status": "ACTIVE",  
      "clusterName": "default",  
      "registeredContainerInstancesCount": 0,  
      "pendingTasksCount": 0,  
      "runningTasksCount": 0,  
      "activeServicesCount": 1,  
      "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/default"  
    }  
  ],  
  "failures": []  
}
```

For more information, see [Amazon ECS Clusters](#) in the *Amazon ECS Developer Guide*.

Example 2: To describe a cluster with the attachment option

The following `describe-clusters` example specifies the `ATTACHMENTS` option. It retrieves details about the specified cluster and a list of resources attached to the cluster in the form of attachments. When using a capacity provider with a cluster, the resources, either `AutoScaling` plans or scaling policies, will be represented as `asp` or `as_policy` `ATTACHMENTS`.

```
aws ecs describe-clusters \  
  --include ATTACHMENTS \  
  --clusters sampleCluster
```

Output:

```
{
```

```
"clusters": [
  {
    "clusterArn": "arn:aws:ecs:af-south-1:123456789222:cluster/
sampleCluster",
    "clusterName": "sampleCluster",
    "status": "ACTIVE",
    "registeredContainerInstancesCount": 0,
    "runningTasksCount": 0,
    "pendingTasksCount": 0,
    "activeServicesCount": 0,
    "statistics": [],
    "tags": [],
    "settings": [],
    "capacityProviders": [
      "sampleCapacityProvider"
    ],
    "defaultCapacityProviderStrategy": [],
    "attachments": [
      {
        "id": "a1b2c3d4-5678-901b-cdef-EXAMPLE22222",
        "type": "as_policy",
        "status": "CREATED",
        "details": [
          {
            "name": "capacityProviderName",
            "value": "sampleCapacityProvider"
          },
          {
            "name": "scalingPolicyName",
            "value": "ECSManagedAutoScalingPolicy-3048e262-
fe39-4eaf-826d-6f975d303188"
          }
        ]
      }
    ],
    "attachmentsStatus": "UPDATE_COMPLETE"
  }
],
"failures": []
}
```

For more information, see [Amazon ECS Clusters](#) in the *Amazon ECS Developer Guide*.

- For API details, see [DescribeClusters](#) in *AWS CLI Command Reference*.

describe-container-instances

The following code example shows how to use `describe-container-instances`.

AWS CLI

To describe container instance

The following `describe-container-instances` example retrieves details for a container instance in the update cluster, using the container instance UUID as an identifier.

```
aws ecs describe-container-instances \
  --cluster update \
  --container-instances a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
```

Output:

```
{
  "failures": [],
  "containerInstances": [
    {
      "status": "ACTIVE",
      "registeredResources": [
        {
          "integerValue": 2048,
          "longValue": 0,
          "type": "INTEGER",
          "name": "CPU",
          "doubleValue": 0.0
        },
        {
          "integerValue": 3955,
          "longValue": 0,
          "type": "INTEGER",
          "name": "MEMORY",
          "doubleValue": 0.0
        },
        {
          "name": "PORTS",
          "longValue": 0,
          "doubleValue": 0.0,
          "stringSetValue": [
            "22",
```

```
        "2376",
        "2375",
        "51678"
    ],
    "type": "STRINGSET",
    "integerValue": 0
}
],
"ec2InstanceId": "i-A1B2C3D4",
"agentConnected": true,
"containerInstanceArn": "arn:aws:ecs:us-west-2:123456789012:container-
instance/a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
"pendingTasksCount": 0,
"remainingResources": [
    {
        "integerValue": 2048,
        "longValue": 0,
        "type": "INTEGER",
        "name": "CPU",
        "doubleValue": 0.0
    },
    {
        "integerValue": 3955,
        "longValue": 0,
        "type": "INTEGER",
        "name": "MEMORY",
        "doubleValue": 0.0
    },
    {
        "name": "PORTS",
        "longValue": 0,
        "doubleValue": 0.0,
        "stringSetValue": [
            "22",
            "2376",
            "2375",
            "51678"
        ],
        "type": "STRINGSET",
        "integerValue": 0
    }
],
"runningTasksCount": 0,
"versionInfo": {
```

```

        "agentVersion": "1.0.0",
        "agentHash": "4023248",
        "dockerVersion": "DockerVersion: 1.5.0"
    }
}
]
}

```

For more information, see [Amazon ECS Container Instances](#) in the *Amazon ECS Developer Guide*.

- For API details, see [DescribeContainerInstances](#) in *AWS CLI Command Reference*.

describe-services

The following code example shows how to use `describe-services`.

AWS CLI

To describe a service

The following `describe-services` example retrieves details for the `my-http-service` service in the default cluster.

```
aws ecs describe-services --services my-http-service
```

Output:

```

{
  "services": [
    {
      "status": "ACTIVE",
      "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/
amazon-ecs-sample:1",
      "pendingCount": 0,
      "loadBalancers": [],
      "desiredCount": 10,
      "createdAt": 1466801808.595,
      "serviceName": "my-http-service",
      "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/default",
      "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/my-http-
service",
      "deployments": [

```

```

        {
            "status": "PRIMARY",
            "pendingCount": 0,
            "createdAt": 1466801808.595,
            "desiredCount": 10,
            "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-
definition/amazon-ecs-sample:1",
            "updatedAt": 1428326312.703,
            "id": "ecs-svc/1234567890123456789",
            "runningCount": 10
        }
    ],
    "events": [
        {
            "message": "(service my-http-service) has reached a steady
state.",
            "id": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
            "createdAt": 1466801812.435
        }
    ],
    "runningCount": 10
}
],
"failures": []
}

```

For more information, see [Services](#) in the *Amazon ECS Developer Guide*.

- For API details, see [DescribeServices](#) in *AWS CLI Command Reference*.

describe-task-definition

The following code example shows how to use `describe-task-definition`.

AWS CLI

To describe a task definition

The following `describe-task-definition` example retrieves the details of a task definition.

```
aws ecs describe-task-definition \
  --task-definition hello_world:8
```

Output:

```
{
  "tasks": [
    {
      "attachments": [
        {
          "id": "17f3dff6-a9e9-4d83-99a9-7eb5193c2634",
          "type": "ElasticNetworkInterface",
          "status": "ATTACHED",
          "details": [
            {
              "name": "subnetId",
              "value": "subnet-0d0eab1bb38d5ca64"
            },
            {
              "name": "networkInterfaceId",
              "value": "eni-0d542ffb4a12aa6d9"
            },
            {
              "name": "macAddress",
              "value": "0e:6d:18:f6:2d:29"
            },
            {
              "name": "privateDnsName",
              "value": "ip-10-0-1-170.ec2.internal"
            },
            {
              "name": "privateIPv4Address",
              "value": "10.0.1.170"
            }
          ]
        }
      ],
      "attributes": [
        {
          "name": "ecs.cpu-architecture",
          "value": "x86_64"
        }
      ],
      "availabilityZone": "us-east-1b",
      "clusterArn": "arn:aws:ecs:us-east-1:053534965804:cluster/fargate-
cluster",
      "connectivity": "CONNECTED",
```

```
    "connectivityAt": "2023-11-28T11:10:52.907000-05:00",
    "containers": [
      {
        "containerArn": "arn:aws:ecs:us-east-1:053534965804:container/
fargate-cluster/
c524291ae4154100b601a543108b193a/772c4784-92ae-414e-8df2-03d3358e39fa",
        "taskArn": "arn:aws:ecs:us-east-1:053534965804:task/fargate-
cluster/c524291ae4154100b601a543108b193a",
        "name": "web",
        "image": "nginx",
        "imageDigest":
"sha256:10d1f5b58f74683ad34eb29287e07dab1e90f10af243f151bb50aa5dbb4d62ee",
        "runtimeId": "c524291ae4154100b601a543108b193a-265927825",
        "lastStatus": "RUNNING",
        "networkBindings": [],
        "networkInterfaces": [
          {
            "attachmentId": "17f3dff6-a9e9-4d83-99a9-7eb5193c2634",
            "privateIpv4Address": "10.0.1.170"
          }
        ],
        "healthStatus": "HEALTHY",
        "cpu": "99",
        "memory": "100"
      },
      {
        "containerArn": "arn:aws:ecs:us-east-1:053534965804:container/
fargate-cluster/c524291ae4154100b601a543108b193a/c051a779-40d2-48ca-
ad5e-6ec875ceb610",
        "taskArn": "arn:aws:ecs:us-east-1:053534965804:task/fargate-
cluster/c524291ae4154100b601a543108b193a",
        "name": "aws-guardduty-agent-FvWGoDU",
        "imageDigest":
"sha256:359b8b014e5076c625daa1056090e522631587a7afa3b2e055edda6bd1141017",
        "runtimeId": "c524291ae4154100b601a543108b193a-505093495",
        "lastStatus": "RUNNING",
        "networkBindings": [],
        "networkInterfaces": [
          {
            "attachmentId": "17f3dff6-a9e9-4d83-99a9-7eb5193c2634",
            "privateIpv4Address": "10.0.1.170"
          }
        ],
        "healthStatus": "UNKNOWN"
      }
    ]
  }
}
```



```

    }
  ],
  "cpu": "256",
  "createdAt": "2023-11-28T11:10:49.299000-05:00",
  "desiredStatus": "RUNNING",
  "enableExecuteCommand": false,
  "group": "family:webserver",
  "healthStatus": "HEALTHY",
  "lastStatus": "RUNNING",
  "launchType": "FARGATE",
  "memory": "512"
  "platformVersion": "1.4.0",
  "platformFamily": "Linux",
  "pullStartedAt": "2023-11-28T11:10:59.773000-05:00",
  "pullStoppedAt": "2023-11-28T11:11:12.624000-05:00",
  "startedAt": "2023-11-28T11:11:20.316000-05:00",
  "tags": [],
  "taskArn": "arn:aws:ecs:us-east-1:053534965804:task/fargate-cluster/
c524291ae4154100b601a543108b193a",
  "taskDefinitionArn": "arn:aws:ecs:us-east-1:053534965804:task-
definition/webserver:5",
  "version": 4,
  "ephemeralStorage": {
    "sizeInGiB": 20
  }
}
],
"failures": []
}

```

For more information, see [Amazon ECS Task Definitions](#) in the *Amazon ECS Developer Guide*.

- For API details, see [DescribeTaskDefinition](#) in *AWS CLI Command Reference*.

describe-task-sets

The following code example shows how to use `describe-task-sets`.

AWS CLI

To describe a task set

The following `describe-task-sets` example describes a task set in a service that uses an external deployer.

```
aws ecs describe-task-sets \  
  --cluster MyCluster \  
  --service MyService \  
  --task-sets arn:aws:ecs:us-west-2:123456789012:task-set/MyCluster/MyService/ecs-  
svc/1234567890123456789
```

Output:

```
{  
  "taskSets": [  
    {  
      "id": "ecs-svc/1234567890123456789",  
      "taskSetArn": "arn:aws:ecs:us-west-2:123456789012:task-set/MyCluster/  
MyService/ecs-svc/1234567890123456789",  
      "status": "ACTIVE",  
      "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/  
sample-fargate:2",  
      "computedDesiredCount": 0,  
      "pendingCount": 0,  
      "runningCount": 0,  
      "createdAt": 1557207715.195,  
      "updatedAt": 1557207740.014,  
      "launchType": "EC2",  
      "networkConfiguration": {  
        "awsvpcConfiguration": {  
          "subnets": [  
            "subnet-12344321"  
          ],  
          "securityGroups": [  
            "sg-1234431"  
          ],  
          "assignPublicIp": "DISABLED"  
        }  
      },  
      "loadBalancers": [],  
      "serviceRegistries": [],  
      "scale": {  
        "value": 0.0,  
        "unit": "PERCENT"  
      },  
    },  
  ],  
}
```

```

        "stabilityStatus": "STEADY_STATE",
        "stabilityStatusAt": 1557207740.014
      }
    ],
    "failures": []
  }

```

- For API details, see [DescribeTaskSets](#) in *AWS CLI Command Reference*.

describe-tasks

The following code example shows how to use describe-tasks.

AWS CLI

Example 1: To describe a single task tasks

The following describe-tasks example retrieves the details of a task in a cluster. You can specify the task by using either the ID or full ARN of the task. This example uses the full ARN of the task.

```

aws ecs describe-tasks \
  --cluster MyCluster \
  --tasks arn:aws:ecs:us-east-1:123456789012:task/
MyCluster/4d590253bb114126b7afa7b58EXAMPLE

```

Output:

```

{
  "tasks": [
    {
      "attachments": [],
      "attributes": [
        {
          "name": "ecs.cpu-architecture",
          "value": "x86_64"
        }
      ],
      "availabilityZone": "us-east-1b",
      "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/MyCluster",
      "connectivity": "CONNECTED",
      "connectivityAt": "2021-08-11T12:21:26.681000-04:00",

```

```
    "containerInstanceArn": "arn:aws:ecs:us-east-1:123456789012:container-
instance/test/025c7e2c5e054a6790a29fc1fEXAMPLE",
    "containers": [
      {
        "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/
MyCluster/4d590253bb114126b7afa7b58eea9221/a992d1cc-ea46-474a-b6e8-24688EXAMPLE",
        "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/
MyCluster/4d590253bb114126b7afa7b58EXAMPLE",
        "name": "simple-app",
        "image": "httpd:2.4",
        "runtimeId":
"91251eed27db90006ad67b1a08187290869f216557717dd5c39b37c94EXAMPLE",
        "lastStatus": "RUNNING",
        "networkBindings": [
          {
            "bindIP": "0.0.0.0",
            "containerPort": 80,
            "hostPort": 80,
            "protocol": "tcp"
          }
        ],
        "networkInterfaces": [],
        "healthStatus": "UNKNOWN",
        "cpu": "10",
        "memory": "300"
      }
    ],
    "cpu": "10",
    "createdAt": "2021-08-11T12:21:26.681000-04:00",
    "desiredStatus": "RUNNING",
    "enableExecuteCommand": false,
    "group": "service:testupdate",
    "healthStatus": "UNKNOWN",
    "lastStatus": "RUNNING",
    "launchType": "EC2",
    "memory": "300",
    "overrides": {
      "containerOverrides": [
        {
          "name": "simple-app"
        }
      ],
      "inferenceAcceleratorOverrides": []
    }
  },
```

```

    "pullStartedAt": "2021-08-11T12:21:28.234000-04:00",
    "pullStoppedAt": "2021-08-11T12:21:33.793000-04:00",
    "startedAt": "2021-08-11T12:21:34.945000-04:00",
    "startedBy": "ecs-svc/968695068243EXAMPLE",
    "tags": [],
    "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/
MyCluster/4d590253bb114126b7afa7b58eea9221",
    "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-
definition/console-sample-app-static2:1",
    "version": 2
  }
],
"failures": []
}

```

For more information, see [Amazon ECS Task Definitions](#) in the *Amazon ECS Developer Guide*.

Example 2: To describe multiple tasks

The following `describe-tasks` example retrieves the details of multiple tasks in a cluster. You can specify the task by using either the ID or full ARN of the task. This example uses the full IDs of the tasks.

```

aws ecs describe-tasks \
  --cluster MyCluster \
  --tasks "74de0355a10a4f979ac495c14EXAMPLE" "d789e94343414c25b9f6bd59eEXAMPLE"

```

Output:

```

{
  "tasks": [
    {
      "attachments": [
        {
          "id": "d9e7735a-16aa-4128-bc7a-b2d51EXAMPLE",
          "type": "ElasticNetworkInterface",
          "status": "ATTACHED",
          "details": [
            {
              "name": "subnetId",
              "value": "subnet-0d0eab1bb3EXAMPLE"
            },
            {

```

```
        "name": "networkInterfaceId",
        "value": "eni-0fa40520aeEXAMPLE"
    },
    {
        "name": "macAddress",
        "value": "0e:89:76:28:07:b3"
    },
    {
        "name": "privateDnsName",
        "value": "ip-10-0-1-184.ec2.internal"
    },
    {
        "name": "privateIPv4Address",
        "value": "10.0.1.184"
    }
    ]
}
],
"attributes": [
    {
        "name": "ecs.cpu-architecture",
        "value": "x86_64"
    }
],
"availabilityZone": "us-east-1b",
"clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/MyCluster",
"connectivity": "CONNECTED",
"connectivityAt": "2021-12-20T12:13:37.875000-05:00",
"containers": [
    {
        "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/
MyCluster/74de0355a10a4f979ac495c14EXAMPLE/aad3ba00-83b3-4dac-84d4-11f8cEXAMPLE",
        "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/
MyCluster/74de0355a10a4f979ac495c14EXAMPLE",
        "name": "web",
        "image": "nginx",
        "runtimeId": "74de0355a10a4f979ac495c14EXAMPLE-265927825",
        "lastStatus": "RUNNING",
        "networkBindings": [],
        "networkInterfaces": [
            {
                "attachmentId": "d9e7735a-16aa-4128-bc7a-b2d51EXAMPLE",
                "privateIpv4Address": "10.0.1.184"
            }
        ]
    }
]
```

```
    ],
    "healthStatus": "UNKNOWN",
    "cpu": "99",
    "memory": "100"
  }
],
"cpu": "256",
"createdAt": "2021-12-20T12:13:20.226000-05:00",
"desiredStatus": "RUNNING",
"enableExecuteCommand": false,
"group": "service:tdsevicetag",
"healthStatus": "UNKNOWN",
"lastStatus": "RUNNING",
"launchType": "FARGATE",
"memory": "512",
"overrides": {
  "containerOverrides": [
    {
      "name": "web"
    }
  ],
  "inferenceAcceleratorOverrides": []
},
"platformVersion": "1.4.0",
"platformFamily": "Linux",
"pullStartedAt": "2021-12-20T12:13:42.665000-05:00",
"pullStoppedAt": "2021-12-20T12:13:46.543000-05:00",
"startedAt": "2021-12-20T12:13:48.086000-05:00",
"startedBy": "ecs-svc/988401040018EXAMPLE",
"tags": [],
"taskArn": "arn:aws:ecs:us-east-1:123456789012:task/MyCluster/74de0355a10a4f979ac495c14EXAMPLE",
"taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/webserver:2",
"version": 3,
"ephemeralStorage": {
  "sizeInGiB": 20
}
},
{
  "attachments": [
    {
      "id": "214eb5a9-45cd-4bf8-87bc-57fefEXAMPLE",
      "type": "ElasticNetworkInterface",
```

```
    "status": "ATTACHED",
    "details": [
      {
        "name": "subnetId",
        "value": "subnet-0d0eab1bb3EXAMPLE"
      },
      {
        "name": "networkInterfaceId",
        "value": "eni-064c7766daEXAMPLE"
      },
      {
        "name": "macAddress",
        "value": "0e:76:83:01:17:a9"
      },
      {
        "name": "privateDnsName",
        "value": "ip-10-0-1-41.ec2.internal"
      },
      {
        "name": "privateIPv4Address",
        "value": "10.0.1.41"
      }
    ]
  },
  "attributes": [
    {
      "name": "ecs.cpu-architecture",
      "value": "x86_64"
    }
  ],
  "availabilityZone": "us-east-1b",
  "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/MyCluster",
  "connectivity": "CONNECTED",
  "connectivityAt": "2021-12-20T12:13:35.243000-05:00",
  "containers": [
    {
      "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/MyCluster/d789e94343414c25b9f6bd59eEXAMPLE/9afef792-609b-43a5-bb6a-3efdbEXAMPLE",
      "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/MyCluster/d789e94343414c25b9f6bd59eEXAMPLE",
      "name": "web",
      "image": "nginx",
      "runtimeId": "d789e94343414c25b9f6bd59eEXAMPLE-265927825",
```



```
        "lastStatus": "RUNNING",
        "networkBindings": [],
        "networkInterfaces": [
            {
                "attachmentId": "214eb5a9-45cd-4bf8-87bc-57fefEXAMPLE",
                "privateIpv4Address": "10.0.1.41"
            }
        ],
        "healthStatus": "UNKNOWN",
        "cpu": "99",
        "memory": "100"
    }
],
"cpu": "256",
"createdAt": "2021-12-20T12:13:20.226000-05:00",
"desiredStatus": "RUNNING",
"enableExecuteCommand": false,
"group": "service:tdsevicetag",
"healthStatus": "UNKNOWN",
"lastStatus": "RUNNING",
"launchType": "FARGATE",
"memory": "512",
"overrides": {
    "containerOverrides": [
        {
            "name": "web"
        }
    ],
    "inferenceAcceleratorOverrides": []
},
"platformVersion": "1.4.0",
"platformFamily": "Linux",
"pullStartedAt": "2021-12-20T12:13:44.611000-05:00",
"pullStoppedAt": "2021-12-20T12:13:48.251000-05:00",
"startedAt": "2021-12-20T12:13:49.326000-05:00",
"startedBy": "ecs-svc/988401040018EXAMPLE",
"tags": [],
"taskArn": "arn:aws:ecs:us-east-1:123456789012:task/MyCluster/d789e94343414c25b9f6bd59eEXAMPLE",
"taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/webserver:2",
"version": 3,
"ephemeralStorage": {
    "sizeInGiB": 20
}
```

```
    }  
  }  
],  
"failures": []  
}
```

For more information, see [Amazon ECS Task Definitions](#) in the *Amazon ECS Developer Guide*.

- For API details, see [DescribeTasks](#) in *AWS CLI Command Reference*.

execute-command

The following code example shows how to use `execute-command`.

AWS CLI

To run an interactive `/bin/sh` command

The following `execute-command` example runs an interactive `/bin/sh` command against a container named `MyContainer` for a task with an id of `arn:aws:ecs:us-east-1:123456789012:task/MyCluster/d789e94343414c25b9f6bd59eEXAMPLE`.

```
aws ecs execute-command \  
  --cluster MyCluster \  
  --task arn:aws:ecs:us-east-1:123456789012:task/MyCluster/  
d789e94343414c25b9f6bd59eEXAMPLE \  
  --container MyContainer \  
  --interactive \  
  --command "/bin/sh"
```

This command produces no output.

For more information, see [Using Amazon ECS Exec for debugging](#) in the *Amazon ECS Developer Guide*.

- For API details, see [ExecuteCommand](#) in *AWS CLI Command Reference*.

list-account-settings

The following code example shows how to use `list-account-settings`.

AWS CLI

Example 1: To view the account settings for an account

The following `list-account-settings` example displays the effective account settings for an account.

```
aws ecs list-account-settings --effective-settings
```

Output:

```
{
  "settings": [
    {
      "name": "containerInstanceLongArnFormat",
      "value": "enabled",
      "principalArn": "arn:aws:iam::123456789012:root"
    },
    {
      "name": "serviceLongArnFormat",
      "value": "enabled",
      "principalArn": "arn:aws:iam::123456789012:root"
    },
    {
      "name": "taskLongArnFormat",
      "value": "enabled",
      "principalArn": "arn:aws:iam::123456789012:root"
    }
  ]
}
```

Example 2: To view the account settings for a specific IAM user or IAM role

The following `list-account-settings` example displays the account settings for the specified IAM user or IAM role.

```
aws ecs list-account-settings --principal-arn arn:aws:iam::123456789012:user/MyUser
```

Output:

```
{
  "settings": [
```

```
{
  "name": "serviceLongArnFormat",
  "value": "enabled",
  "principalArn": "arn:aws:iam::123456789012:user/MyUser"
}
]
```

For more information, see [Amazon Resource Names \(ARNs\) and IDs](#) in the *Amazon ECS Developer Guide*.

- For API details, see [ListAccountSettings](#) in *AWS CLI Command Reference*.

list-attributes

The following code example shows how to use `list-attributes`.

AWS CLI

To list the container instances that contain a specific attribute

The following example lists the attributes for container instances that have the `stack=production` attribute in the default cluster.

```
aws ecs list-attributes \
  --target-type container-instance \
  --attribute-name stack \
  --attribute-value production \
  --cluster default
```

Output:

```
{
  "attributes": [
    {
      "name": "stack",
      "targetId": "arn:aws:ecs:us-west-2:130757420319:container-
instance/1c3be8ed-df30-47b4-8f1e-6e68ebd01f34",
      "value": "production"
    }
  ]
}
```

For more information, see [Amazon ECS Container Agent Configuration](#) in the *Amazon ECS Developer Guide*.

- For API details, see [ListAttributes](#) in *AWS CLI Command Reference*.

list-clusters

The following code example shows how to use `list-clusters`.

AWS CLI

To list your available clusters

The following `list-clusters` example lists all of the available clusters.

```
aws ecs list-clusters
```

Output:

```
{
  "clusterArns": [
    "arn:aws:ecs:us-west-2:123456789012:cluster/MyECSCluster1",
    "arn:aws:ecs:us-west-2:123456789012:cluster/AnotherECSCluster"
  ]
}
```

For more information, see [Amazon ECS Clusters](#) in the *Amazon ECS Developer Guide*.

- For API details, see [ListClusters](#) in *AWS CLI Command Reference*.

list-container-instances

The following code example shows how to use `list-container-instances`.

AWS CLI

To list the container instances in a cluster

The following `list-container-instances` example lists all of the available container instances in a cluster.

```
aws ecs list-container-instances --cluster MyCluster
```

Output:

```
{
  "containerInstanceArns": [
    "arn:aws:ecs:us-west-2:123456789012:container-instance/MyCluster/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "arn:aws:ecs:us-west-2:123456789012:container-instance/MyCluster/
a1b2c3d4-5678-90ab-cdef-22222EXAMPLE"
  ]
}
```

For more information, see [Amazon ECS Container Instances](#) in the *Amazon ECS Developer Guide*.

- For API details, see [ListContainerInstances](#) in *AWS CLI Command Reference*.

list-services-by-namespace

The following code example shows how to use `list-services-by-namespace`.

AWS CLI**To list the services in a namespace**

The following `list-services-by-namespace` example lists all of the services configured for the specified namespace in your default Region.

```
aws ecs list-services-by-namespace \
  --namespace service-connect
```

Output:

```
{
  "serviceArns": [
    "arn:aws:ecs:us-west-2:123456789012:service/MyCluster/MyService",
    "arn:aws:ecs:us-west-2:123456789012:service/tutorial/service-connect-nginx-
service"
  ]
}
```

For more information, see [Service Connect](#) in the *Amazon ECS Developer Guide*.

- For API details, see [ListServicesByNamespace](#) in *AWS CLI Command Reference*.

list-services

The following code example shows how to use `list-services`.

AWS CLI

To list the services in a cluster

The following `list-services` example shows how to list the services running in a cluster.

```
aws ecs list-services --cluster MyCluster
```

Output:

```
{
  "serviceArns": [
    "arn:aws:ecs:us-west-2:123456789012:service/MyCluster/MyService"
  ]
}
```

For more information, see [Services](#) in the *Amazon ECS Developer Guide*.

- For API details, see [ListServices](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags for a resource

The following `list-tags-for-resource` example lists the tags for a specific cluster.

```
aws ecs list-tags-for-resource \
  --resource-arn arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster
```

Output:

```
{
  "tags": [
    {
```

```
    "key": "key1",
    "value": "value1"
  },
  {
    "key": "key2",
    "value": "value2"
  },
  {
    "key": "key3",
    "value": "value3"
  }
]
}
```

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

list-task-definition-families

The following code example shows how to use `list-task-definition-families`.

AWS CLI

Example 1: To list the registered task definition families

The following `list-task-definition-families` example lists all of the registered task definition families.

```
aws ecs list-task-definition-families
```

Output:

```
{
  "families": [
    "node-js-app",
    "web-timer",
    "hpcc",
    "hpcc-c4-8xlarge"
  ]
}
```

Example 2: To filter the registered task definition families

The following `list-task-definition-families` example lists the task definition revisions that start with "hpcc".

```
aws ecs list-task-definition-families --family-prefix hpcc
```

Output:

```
{
  "families": [
    "hpcc",
    "hpcc-c4-8xlarge"
  ]
}
```

For more information, see [Task Definition Parameters](#) in the *Amazon ECS Developer Guide*.

- For API details, see [ListTaskDefinitionFamilies](#) in *AWS CLI Command Reference*.

list-task-definitions

The following code example shows how to use `list-task-definitions`.

AWS CLI

Example 1: To list the registered task definitions

The following `list-task-definitions` example lists all of the registered task definitions.

```
aws ecs list-task-definitions
```

Output:

```
{
  "taskDefinitionArns": [
    "arn:aws:ecs:us-west-2:123456789012:task-definition/sleep300:2",
    "arn:aws:ecs:us-west-2:123456789012:task-definition/sleep360:1",
    "arn:aws:ecs:us-west-2:123456789012:task-definition/wordpress:3",
    "arn:aws:ecs:us-west-2:123456789012:task-definition/wordpress:4",
    "arn:aws:ecs:us-west-2:123456789012:task-definition/wordpress:5",
    "arn:aws:ecs:us-west-2:123456789012:task-definition/wordpress:6"
  ]
}
```

```
}
```

Example 2: To list the registered task definitions in a family

The following `list-task-definitions` example lists the task definition revisions of a specified family.

```
aws ecs list-task-definitions --family-prefix wordpress
```

Output:

```
{
  "taskDefinitionArns": [
    "arn:aws:ecs:us-west-2:123456789012:task-definition/wordpress:3",
    "arn:aws:ecs:us-west-2:123456789012:task-definition/wordpress:4",
    "arn:aws:ecs:us-west-2:123456789012:task-definition/wordpress:5",
    "arn:aws:ecs:us-west-2:123456789012:task-definition/wordpress:6"
  ]
}
```

For more information, see [Amazon ECS Task Definitions](#) in the *Amazon ECS Developer Guide*.

- For API details, see [ListTaskDefinitions](#) in *AWS CLI Command Reference*.

list-tasks

The following code example shows how to use `list-tasks`.

AWS CLI

Example 1: To list the tasks in a cluster

The following `list-tasks` example lists all of the tasks in a cluster.

```
aws ecs list-tasks --cluster default
```

Output:

```
{
  "taskArns": [
```

```
    "arn:aws:ecs:us-west-2:123456789012:task/a1b2c3d4-5678-90ab-
cdef-11111EXAMPLE",
    "arn:aws:ecs:us-west-2:123456789012:task/a1b2c3d4-5678-90ab-
cdef-22222EXAMPLE"
  ]
}
```

Example 2: To list the tasks on a particular container instance

The following `list-tasks` example lists the tasks on a container instance, using the container instance UUID as a filter.

```
aws ecs list-tasks --cluster default --container-instance a1b2c3d4-5678-90ab-
cdef-33333EXAMPLE
```

Output:

```
{
  "taskArns": [
    "arn:aws:ecs:us-west-2:123456789012:task/a1b2c3d4-5678-90ab-
cdef-44444EXAMPLE"
  ]
}
```

For more information, see [Amazon ECS Task Definitions](#) in the *Amazon ECS Developer Guide*.

- For API details, see [ListTasks](#) in *AWS CLI Command Reference*.

put-account-setting-default

The following code example shows how to use `put-account-setting-default`.

AWS CLI

To modify the default account settings

The following `put-account-setting-default` example modifies the default account setting for all IAM users or roles on your account. These changes apply to the entire AWS account unless an IAM user or role explicitly overrides these settings for themselves.

```
aws ecs put-account-setting-default --name serviceLongArnFormat --value enabled
```

Output:

```
{
  "setting": {
    "name": "serviceLongArnFormat",
    "value": "enabled",
    "principalArn": "arn:aws:iam::123456789012:root"
  }
}
```

For more information, see [Amazon Resource Names \(ARNs\) and IDs](#) in the *Amazon ECS Developer Guide*.

- For API details, see [PutAccountSettingDefault](#) in *AWS CLI Command Reference*.

put-account-setting

The following code example shows how to use `put-account-setting`.

AWS CLI**To modify the account setting for your IAM user account**

The following `put-account-setting` example enables the `serviceLongArnFormat` account setting for your IAM user account.

```
aws ecs put-account-setting --name serviceLongArnFormat --value enabled
```

Output:

```
{
  "setting": {
    "name": "serviceLongArnFormat",
    "value": "enabled",
    "principalArn": "arn:aws:iam::130757420319:user/your_username"
  }
}
```

For more information, see [Modifying Account Settings](#) in the *Amazon ECS Developer Guide*.

- For API details, see [PutAccountSetting](#) in *AWS CLI Command Reference*.

put-account-settings

The following code example shows how to use `put-account-settings`.

AWS CLI

To modify the account settings for an IAM user or IAM role

The following `put-account-setting` example modifies the account settings for the specified IAM user or IAM role.

```
aws ecs put-account-setting \  
  --name serviceLongArnFormat \  
  --value enabled \  
  --principal-arn arn:aws:iam::123456789012:user/MyUser
```

Output:

```
{  
  "setting": {  
    "name": "serviceLongArnFormat",  
    "value": "enabled",  
    "principalArn": "arn:aws:iam::123456789012:user/MyUser"  
  }  
}
```

- For API details, see [PutAccountSettings](#) in *AWS CLI Command Reference*.

put-attributes

The following code example shows how to use `put-attributes`.

AWS CLI

To create an attribute and associate it with an Amazon ECS resource

The following `put-attributes` applies an attribute with the name `stack` and the value `production` to a container instance.

```
aws ecs put-attributes \  
  --attributes name=stack,value=production,targetId=arn:aws:ecs:us-  
west-2:130757420319:container-instance/1c3be8ed-df30-47b4-8f1e-6e68ebd01f34
```

Output:

```
{
  "attributes": [
    {
      "name": "stack",
      "targetId": "arn:aws:ecs:us-west-2:130757420319:container-
instance/1c3be8ed-df30-47b4-8f1e-6e68ebd01f34",
      "value": "production"
    }
  ]
}
```

- For API details, see [PutAttributes](#) in *AWS CLI Command Reference*.

put-cluster-capacity-providers

The following code example shows how to use `put-cluster-capacity-providers`.

AWS CLI**Example 1: To add an existing capacity provider to a cluster**

The following `put-cluster-capacity-providers` example adds an existing capacity provider to a cluster. The `create-capacity-provider` command is used to create a capacity provider. The `describe-clusters` command is used to describe the current capacity providers and the default capacity provider strategy associated with a cluster. When adding a new capacity provider to a cluster, you must specify all existing capacity providers in addition to the new capacity provider you want to associate with the cluster. You must also specify the default capacity provider strategy to associate with the cluster. In this example, the `MyCluster` cluster has the `MyCapacityProvider1` capacity provider associated with it and you want to add the `MyCapacityProvider2` capacity provider and include it in the default capacity provider strategy so tasks are spread evenly across both capacity providers.

```
aws ecs put-cluster-capacity-providers \
  --cluster MyCluster \
  --capacity-providers MyCapacityProvider1 MyCapacityProvider2 \
  --default-capacity-provider-strategy
capacityProvider=MyCapacityProvider1,weight=1
capacityProvider=MyCapacityProvider2,weight=1
```

Output:

```
{
  "cluster": {
    "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",
    "clusterName": "MyCluster",
    "status": "ACTIVE",
    "registeredContainerInstancesCount": 0,
    "runningTasksCount": 0,
    "pendingTasksCount": 0,
    "activeServicesCount": 0,
    "statistics": [],
    "tags": [],
    "settings": [
      {
        "name": "containerInsights",
        "value": "enabled"
      }
    ],
    "capacityProviders": [
      "MyCapacityProvider1",
      "MyCapacityProvider2"
    ],
    "defaultCapacityProviderStrategy": [
      {
        "capacityProvider": "MyCapacityProvider1",
        "weight": 1,
        "base": 0
      },
      {
        "capacityProvider": "MyCapacityProvider2",
        "weight": 1,
        "base": 0
      }
    ],
    "attachments": [
      {
        "id": "0fb0c8f4-6edd-4de1-9b09-17e470ee1918",
        "type": "as_policy",
        "status": "ACTIVE",
        "details": [
          {
            "name": "capacityProviderName",
            "value": "MyCapacityProvider1"
          }
        ]
      }
    ]
  }
}
```

```

        },
        {
            "name": "scalingPolicyName",
            "value": "ECSManagedAutoScalingPolicy-a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111"
        }
    ]
},
{
    "id": "ae592060-2382-4663-9476-b015c685593c",
    "type": "as_policy",
    "status": "ACTIVE",
    "details": [
        {
            "name": "capacityProviderName",
            "value": "MyCapacityProvider2"
        },
        {
            "name": "scalingPolicyName",
            "value": "ECSManagedAutoScalingPolicy-a1b2c3d4-5678-90ab-
cdef-EXAMPLE22222"
        }
    ]
}
],
"attachmentsStatus": "UPDATE_IN_PROGRESS"
}
}

```

For more information, see [Cluster capacity providers](#) in the *Amazon ECS Developer Guide*.

Example 2: To remove a capacity provider from a cluster

The following `put-cluster-capacity-providers` example removes a capacity provider from a cluster. The `describe-clusters` command is used to describe the current capacity providers associated with a cluster. When removing a capacity provider from a cluster, you must specify the capacity providers you want to remain associated with the cluster as well as the default capacity provider strategy to associate with the cluster. In this example, the cluster has the `MyCapacityProvider1` and `MyCapacityProvider2` capacity providers associated with it and you want to remove the `MyCapacityProvider2` capacity provider, so you specify only `MyCapacityProvider1` in the command along with the updated default capacity provider strategy.


```
aws ecs put-cluster-capacity-providers \  
  --cluster MyCluster \  
  --capacity-providers MyCapacityProvider1 \  
  --default-capacity-provider-strategy  
capacityProvider=MyCapacityProvider1,weight=1,base=0
```

Output:

```
{  
  "cluster": {  
    "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",  
    "clusterName": "MyCluster",  
    "status": "ACTIVE",  
    "registeredContainerInstancesCount": 0,  
    "runningTasksCount": 0,  
    "pendingTasksCount": 0,  
    "activeServicesCount": 0,  
    "statistics": [],  
    "tags": [],  
    "settings": [  
      {  
        "name": "containerInsights",  
        "value": "enabled"  
      }  
    ],  
    "capacityProviders": [  
      "MyCapacityProvider1"  
    ],  
    "defaultCapacityProviderStrategy": [  
      "capacityProvider": "MyCapacityProvider1",  
      "weight": 1,  
      "base": 0  
    ],  
    "attachments": [  
      {  
        "id": "0fb0c8f4-6edd-4de1-9b09-17e470ee1918",  
        "type": "as_policy",  
        "status": "ACTIVE",  
        "details": [  
          {  
            "name": "capacityProviderName",  
            "value": "MyCapacityProvider1"  
          }  
        ],  
      }  
    ],  
  }  
}
```

```

        {
            "name": "scalingPolicyName",
            "value": "ECManagedAutoScalingPolicy-a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111"
        }
    ]
},
{
    "id": "ae592060-2382-4663-9476-b015c685593c",
    "type": "as_policy",
    "status": "DELETING",
    "details": [
        {
            "name": "capacityProviderName",
            "value": "MyCapacityProvider2"
        },
        {
            "name": "scalingPolicyName",
            "value": "ECManagedAutoScalingPolicy-a1b2c3d4-5678-90ab-
cdef-EXAMPLE22222"
        }
    ]
}
],
"attachmentsStatus": "UPDATE_IN_PROGRESS"
}
}

```

For more information, see [Cluster capacity providers](#) in the *Amazon ECS Developer Guide*.

Example 3: To remove all capacity providers from a cluster

The following `put-cluster-capacity-providers` example removes all existing capacity providers from the cluster.

```

aws ecs put-cluster-capacity-providers \
  --cluster MyCluster \
  --capacity-providers [] \
  --default-capacity-provider-strategy []

```

Output:

```
{
```

```
"cluster": {
  "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",
  "clusterName": "MyCluster",
  "status": "ACTIVE",
  "registeredContainerInstancesCount": 0,
  "runningTasksCount": 0,
  "pendingTasksCount": 0,
  "activeServicesCount": 0,
  "statistics": [],
  "tags": [],
  "settings": [
    {
      "name": "containerInsights",
      "value": "enabled"
    }
  ],
  "capacityProviders": [],
  "defaultCapacityProviderStrategy": [],
  "attachments": [
    {
      "id": "0fb0c8f4-6edd-4de1-9b09-17e470ee1918",
      "type": "as_policy",
      "status": "DELETING",
      "details": [
        {
          "name": "capacityProviderName",
          "value": "MyCapacityProvider1"
        },
        {
          "name": "scalingPolicyName",
          "value": "ECManagedAutoScalingPolicy-a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111"
        }
      ]
    },
    {
      "id": "ae592060-2382-4663-9476-b015c685593c",
      "type": "as_policy",
      "status": "DELETING",
      "details": [
        {
          "name": "capacityProviderName",
          "value": "MyCapacityProvider2"
        }
      ]
    }
  ]
}
```

```

        {
            "name": "scalingPolicyName",
            "value": "ECSManagedAutoScalingPolicy-a1b2c3d4-5678-90ab-
cdef-EXAMPLE22222"
        }
    ]
}
],
"attachmentsStatus": "UPDATE_IN_PROGRESS"
}
}

```

For more information, see [Cluster capacity providers](#) in the *Amazon ECS Developer Guide*.

- For API details, see [PutClusterCapacityProviders](#) in *AWS CLI Command Reference*.

register-task-definition

The following code example shows how to use `register-task-definition`.

AWS CLI

Example 1: To register a task definition with a JSON file

The following `register-task-definition` example registers a task definition to the specified family. The container definitions are saved in JSON format at the specified file location.

```
aws ecs register-task-definition \
  --cli-input-json file://<path_to_json_file>/sleep360.json
```

Contents of `sleep360.json`:

```

{
  "containerDefinitions": [
    {
      "name": "sleep",
      "image": "busybox",
      "cpu": 10,
      "command": [
        "sleep",
        "360"
      ],
    }
  ],
}

```

```
        "memory": 10,
        "essential": true
    }
],
"family": "sleep360"
}
```

Output:

```
{
  "taskDefinition": {
    "status": "ACTIVE",
    "family": "sleep360",
    "placementConstraints": [],
    "compatibilities": [
      "EXTERNAL",
      "EC2"
    ],
    "volumes": [],
    "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/sleep360:1",
    "containerDefinitions": [
      {
        "environment": [],
        "name": "sleep",
        "mountPoints": [],
        "image": "busybox",
        "cpu": 10,
        "portMappings": [],
        "command": [
          "sleep",
          "360"
        ],
        "memory": 10,
        "essential": true,
        "volumesFrom": []
      }
    ],
    "revision": 1
  }
}
```

For more information, see [Example task definitions](#) in the *Amazon ECS Developer Guide*.

Example 2: To register a task definition with a JSON string parameter

The following `register-task-definition` example registers a task definition using container definitions provided as a JSON string parameter with escaped double quotes.

```
aws ecs register-task-definition \  
  --family sleep360 \  
  --container-definitions "[{\"name\":\"sleep\",\"image\":\"busybox\",\"cpu\":10,\  
  \"/>"command\":[\"sleep\",\"360\"],\"memory\":10,\"essential\":true}]"
```

The output is identical to the previous example.

For more information, see [Creating a Task Definition](#) in the *Amazon ECS Developer Guide*.

- For API details, see [RegisterTaskDefinition](#) in *AWS CLI Command Reference*.

run-task

The following code example shows how to use `run-task`.

AWS CLI

To run a task on your default cluster

The following `run-task` example runs a task on the default cluster and uses a client token.

```
aws ecs run-task \  
  --cluster default \  
  --task-definition sleep360:1 \  
  --client-token 550e8400-e29b-41d4-a716-446655440000
```

Output:

```
{  
  "tasks": [  
    {  
      "attachments": [],  
      "attributes": [  
        {  
          "name": "ecs.cpu-architecture",  
          "value": "x86_64"  
        }  
      ],  
    }  
  ],  
}
```

```
    "availabilityZone": "us-east-1b",
    "capacityProviderName": "example-capacity-provider",
    "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/default",
    "containerInstanceArn": "arn:aws:ecs:us-east-1:123456789012:container-
instance/default/bc4d2ec611d04bb7bb97e83ceEXAMPLE",
    "containers": [
      {
        "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/
default/d6f51cc5bbc94a47969c92035e9f66f8/75853d2d-711e-458a-8362-0f0aEXAMPLE",
        "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/default/
d6f51cc5bbc94a47969c9203EXAMPLE",
        "name": "sleep",
        "image": "busybox",
        "lastStatus": "PENDING",
        "networkInterfaces": [],
        "cpu": "10",
        "memory": "10"
      }
    ],
    "cpu": "10",
    "createdAt": "2023-11-21T16:59:34.403000-05:00",
    "desiredStatus": "RUNNING",
    "enableExecuteCommand": false,
    "group": "family:sleep360",
    "lastStatus": "PENDING",
    "launchType": "EC2",
    "memory": "10",
    "overrides": {
      "containerOverrides": [
        {
          "name": "sleep"
        }
      ],
      "inferenceAcceleratorOverrides": []
    },
    "tags": [],
    "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/default/
d6f51cc5bbc94a47969c9203EXAMPLE",
    "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-
definition/sleep360:1",
    "version": 1
  }
],
"failures": []
```

```
}
```

For more information, see [Running Tasks](#) in the *Amazon ECS Developer Guide*.

- For API details, see [RunTask](#) in *AWS CLI Command Reference*.

start-task

The following code example shows how to use `start-task`.

AWS CLI

To start a new task

The following `start-task` starts a task using the latest revision of the `sleep360` task definition on the specified container instance in the default cluster.

```
aws ecs start-task \  
  --task-definition sleep360 \  
  --container-instances 765936fadbdd46b5991a4bd70c2a43d4
```

Output:

```
{  
  "tasks": [  
    {  
      "taskArn": "arn:aws:ecs:us-west-2:130757420319:task/  
default/666fdccc2e2d4b6894dd422f4eeee8f8",  
      "clusterArn": "arn:aws:ecs:us-west-2:130757420319:cluster/default",  
      "taskDefinitionArn": "arn:aws:ecs:us-west-2:130757420319:task-  
definition/sleep360:3",  
      "containerInstanceArn": "arn:aws:ecs:us-west-2:130757420319:container-  
instance/default/765936fadbdd46b5991a4bd70c2a43d4",  
      "overrides": {  
        "containerOverrides": [  
          {  
            "name": "sleep"  
          }  
        ]  
      },  
      "lastStatus": "PENDING",  
      "desiredStatus": "RUNNING",  
      "cpu": "128",
```



```
    "memory": "128",
    "containers": [
      {
        "containerArn": "arn:aws:ecs:us-
west-2:130757420319:container/75f11ed4-8a3d-4f26-a33b-ad1db9e02d41",
        "taskArn": "arn:aws:ecs:us-west-2:130757420319:task/
default/666fdccc2e2d4b6894dd422f4eeee8f8",
        "name": "sleep",
        "lastStatus": "PENDING",
        "networkInterfaces": [],
        "cpu": "10",
        "memory": "10"
      }
    ],
    "version": 1,
    "createdAt": 1563421494.186,
    "group": "family:sleep360",
    "launchType": "EC2",
    "attachments": [],
    "tags": []
  }
],
"failures": []
}
```

- For API details, see [StartTask](#) in *AWS CLI Command Reference*.

stop-task

The following code example shows how to use stop-task.

AWS CLI

To stop a task

The following stop-task stops the specified task from running in the default cluster.

```
aws ecs stop-task \  
  --task 666fdccc2e2d4b6894dd422f4eeee8f8
```

Output:

```
{
```

```
"task": {
  "taskArn": "arn:aws:ecs:us-west-2:130757420319:task/
default/666fdccc2e2d4b6894dd422f4eeee8f8",
  "clusterArn": "arn:aws:ecs:us-west-2:130757420319:cluster/default",
  "taskDefinitionArn": "arn:aws:ecs:us-west-2:130757420319:task-definition/
sleep360:3",
  "containerInstanceArn": "arn:aws:ecs:us-west-2:130757420319:container-
instance/default/765936fadbdd46b5991a4bd70c2a43d4",
  "overrides": {
    "containerOverrides": []
  },
  "lastStatus": "STOPPED",
  "desiredStatus": "STOPPED",
  "cpu": "128",
  "memory": "128",
  "containers": [],
  "version": 2,
  "stoppedReason": "Taskfailedtostart",
  "stopCode": "TaskFailedToStart",
  "connectivity": "CONNECTED",
  "connectivityAt": 1563421494.186,
  "pullStartedAt": 1563421494.252,
  "pullStoppedAt": 1563421496.252,
  "executionStoppedAt": 1563421497,
  "createdAt": 1563421494.186,
  "stoppingAt": 1563421497.252,
  "stoppedAt": 1563421497.252,
  "group": "family:sleep360",
  "launchType": "EC2",
  "attachments": [],
  "tags": []
}
}
```

- For API details, see [StopTask](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To tag a resource

The following `tag-resource` example adds a single tag to the specified resource.

```
aws ecs tag-resource \  
  --resource-arn arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster \  
  --tags key=key1,value=value1
```

This command produces no output.

To add multiple tags to a resource

The following `tag-resource` example adds multiple tags to the specified resource.

```
aws ecs tag-resource \  
  --resource-arn arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster \  
  --tags key=key1,value=value1 key=key2,value=value2 key=key3,value=value3
```

This command produces no output.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag from a resource

The following `untag-resource` example removes the listed tags from the specified resource.

```
aws ecs untag-resource \  
  --resource-arn arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster \  
  --tag-keys key1,key2
```

This command produces no output.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-cluster-settings

The following code example shows how to use `update-cluster-settings`.

AWS CLI

To modify the settings for your cluster

The following `update-cluster-settings` example enables CloudWatch Container Insights for the default cluster.

```
aws ecs update-cluster-settings \  
  --cluster default \  
  --settings name=containerInsights,value=enabled
```

Output:

```
{  
  "cluster": {  
    "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster",  
    "clusterName": "default",  
    "status": "ACTIVE",  
    "registeredContainerInstancesCount": 0,  
    "runningTasksCount": 0,  
    "pendingTasksCount": 0,  
    "activeServicesCount": 0,  
    "statistics": [],  
    "tags": [],  
    "settings": [  
      {  
        "name": "containerInsights",  
        "value": "enabled"  
      }  
    ]  
  }  
}
```

For more information, see [Modifying Account Settings](#) in the *Amazon ECS Developer Guide*.

- For API details, see [UpdateClusterSettings](#) in *AWS CLI Command Reference*.

update-container-agent

The following code example shows how to use `update-container-agent`.

AWS CLI

To update the container agent on an Amazon ECS container instance

The following `update-container-agent` example updates the container agent on the specified container instance in the default cluster.

```
aws ecs update-container-agent --cluster default --container-instance
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
```

Output:

```
{
  "containerInstance": {
    "status": "ACTIVE",
    ...
    "agentUpdateStatus": "PENDING",
    "versionInfo": {
      "agentVersion": "1.0.0",
      "agentHash": "4023248",
      "dockerVersion": "DockerVersion: 1.5.0"
    }
  }
}
```

For more information, see [Updating the Amazon ECS Container Agent](#) in the *Amazon ECS Developer Guide*.

- For API details, see [UpdateContainerAgent](#) in *AWS CLI Command Reference*.

update-container-instances-state

The following code example shows how to use `update-container-instances-state`.

AWS CLI

To update the state of a container instance

The following `update-container-instances-state` updates the state of the specified container instance to DRAINING which will remove it from the cluster is it registered to.

```
aws ecs update-container-instances-state \
```

```
--container-instances 765936fadbdd46b5991a4bd70c2a43d4 \  
--status DRAINING
```

Output:

```
{  
  "containerInstances": [  
    {  
      "containerInstanceArn": "arn:aws:ecs:us-west-2:130757420319:container-  
instance/default/765936fadbdd46b5991a4bd70c2a43d4",  
      "ec2InstanceId": "i-013d87ffbb4d513bf",  
      "version": 4390,  
      "versionInfo": {  
        "agentVersion": "1.29.0",  
        "agentHash": "a190a73f",  
        "dockerVersion": "DockerVersion:18.06.1-ce"  
      },  
      "remainingResources": [  
        {  
          "name": "CPU",  
          "type": "INTEGER",  
          "doubleValue": 0,  
          "longValue": 0,  
          "integerValue": 1536  
        },  
        {  
          "name": "MEMORY",  
          "type": "INTEGER",  
          "doubleValue": 0,  
          "longValue": 0,  
          "integerValue": 2681  
        },  
        {  
          "name": "PORTS",  
          "type": "STRINGSET",  
          "doubleValue": 0,  
          "longValue": 0,  
          "integerValue": 0,  
          "stringSetValue": [  
            "22",  
            "2376",  
            "2375",  
            "51678",  
          ]  
        }  
      ]  
    }  
  ]  
}
```

```
        "51679"
      ]
    },
    {
      "name": "PORTS_UDP",
      "type": "STRINGSET",
      "doubleValue": 0,
      "longValue": 0,
      "integerValue": 0,
      "stringSetValue": []
    }
  ],
  "registeredResources": [
    {
      "name": "CPU",
      "type": "INTEGER",
      "doubleValue": 0,
      "longValue": 0,
      "integerValue": 2048
    },
    {
      "name": "MEMORY",
      "type": "INTEGER",
      "doubleValue": 0,
      "longValue": 0,
      "integerValue": 3705
    },
    {
      "name": "PORTS",
      "type": "STRINGSET",
      "doubleValue": 0,
      "longValue": 0,
      "integerValue": 0,
      "stringSetValue": [
        "22",
        "2376",
        "2375",
        "51678",
        "51679"
      ]
    }
  ],
  {
    "name": "PORTS_UDP",
    "type": "STRINGSET",
```

```
        "doubleValue": 0,
        "longValue": 0,
        "integerValue": 0,
        "stringSetValue": []
    }
],
"status": "DRAINING",
"agentConnected": true,
"runningTasksCount": 2,
"pendingTasksCount": 0,
"attributes": [
    {
        "name": "ecs.capability.secrets.asm.environment-variables"
    },
    {
        "name": "ecs.capability.branch-cni-plugin-version",
        "value": "e0703516-"
    },
    {
        "name": "ecs.ami-id",
        "value": "ami-00e0090ac21971297"
    },
    {
        "name": "ecs.capability.secrets.asm.bootstrap.log-driver"
    },
    {
        "name": "com.amazonaws.ecs.capability.logging-driver.none"
    },
    {
        "name": "ecs.capability.ecr-endpoint"
    },
    {
        "name": "ecs.capability.docker-plugin.local"
    },
    {
        "name": "ecs.capability.task-cpu-mem-limit"
    },
    {
        "name": "ecs.capability.secrets.ssm.bootstrap.log-driver"
    },
    {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.30"
    }
]
```



```
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.31"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.32"
  },
  {
    "name": "ecs.availability-zone",
    "value": "us-west-2c"
  },
  {
    "name": "ecs.capability.aws-appmesh"
  },
  {
    "name": "com.amazonaws.ecs.capability.logging-driver.awslogs"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.24"
  },
  {
    "name": "ecs.capability.task-eni-trunking"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.25"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.26"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.27"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.28"
  },
  {
    "name": "com.amazonaws.ecs.capability.privileged-container"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.29"
  },
  {
    "name": "ecs.cpu-architecture",
    "value": "x86_64"
  },
  {
```

```
    "name": "com.amazonaws.ecs.capability.ecr-auth"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.20"
  },
  {
    "name": "ecs.os-type",
    "value": "linux"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.21"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.22"
  },
  {
    "name": "ecs.capability.task-eia"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.23"
  },
  {
    "name": "ecs.capability.private-registry-
authentication.secretsmanager"
  },
  {
    "name": "com.amazonaws.ecs.capability.logging-driver.syslog"
  },
  {
    "name": "com.amazonaws.ecs.capability.logging-driver.json-file"
  },
  {
    "name": "ecs.capability.execution-role-awslogs"
  },
  {
    "name": "ecs.vpc-id",
    "value": "vpc-1234"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.17"
  },
  {
    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.18"
  },
  },
```

```
host"
    {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.19"
    },
    {
        "name": "ecs.capability.task-eni"
    },
    {
        "name": "ecs.capability.execution-role-ecr-pull"
    },
    {
        "name": "ecs.capability.container-health-check"
    },
    {
        "name": "ecs.subnet-id",
        "value": "subnet-1234"
    },
    {
        "name": "ecs.instance-type",
        "value": "c5.large"
    },
    {
        "name": "com.amazonaws.ecs.capability.task-iam-role-network-
    },
    {
        "name": "ecs.capability.container-ordering"
    },
    {
        "name": "ecs.capability.cni-plugin-version",
        "value": "91ccef8-2019.06.0"
    },
    {
        "name": "ecs.capability.pid-ipc-namespace-sharing"
    },
    {
        "name": "ecs.capability.secrets.ssm.environment-variables"
    },
    {
        "name": "com.amazonaws.ecs.capability.task-iam-role"
    }
],
"registeredAt": 1560788724.507,
"attachments": [],
"tags": []
```

```

    }
  ],
  "failures": []
}

```

- For API details, see [UpdateContainerInstancesState](#) in *AWS CLI Command Reference*.

update-service-primary-task-set

The following code example shows how to use `update-service-primary-task-set`.

AWS CLI

To update the primary task set for a service

The following `update-service-primary-task-set` example updates the primary task set for the specified service.

```

aws ecs update-service-primary-task-set \
  --cluster MyCluster \
  --service MyService \
  --primary-task-set arn:aws:ecs:us-west-2:123456789012:task-set/MyCluster/
MyService/ecs-svc/1234567890123456789

```

Output:

```

{
  "taskSet": {
    "id": "ecs-svc/1234567890123456789",
    "taskSetArn": "arn:aws:ecs:us-west-2:123456789012:task-set/MyCluster/
MyService/ecs-svc/1234567890123456789",
    "status": "PRIMARY",
    "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/
sample-fargate:2",
    "computedDesiredCount": 1,
    "pendingCount": 0,
    "runningCount": 0,
    "createdAt": 1557128360.711,
    "updatedAt": 1557129412.653,
    "launchType": "EC2",
    "networkConfiguration": {
      "awsvpcConfiguration": {

```

```
        "subnets": [
            "subnet-12344321"
        ],
        "securityGroups": [
            "sg-12344312"
        ],
        "assignPublicIp": "DISABLED"
    }
},
"loadBalancers": [],
"serviceRegistries": [],
"scale": {
    "value": 50.0,
    "unit": "PERCENT"
},
"stabilityStatus": "STABILIZING",
"stabilityStatusAt": 1557129279.914
}
}
```

- For API details, see [UpdateServicePrimaryTaskSet](#) in *AWS CLI Command Reference*.

update-service

The following code example shows how to use `update-service`.

AWS CLI

Example 1: To change the task definition used in a service

The following `update-service` example updates the `my-http-service` service to use the `amazon-ecs-sample` task definition.

```
aws ecs update-service --service my-http-service --task-definition amazon-ecs-sample
```

Example 2: To change the number of tasks in a service

The following `update-service` example updates the desired task count of the service `my-http-service` to 3.

```
aws ecs update-service --service my-http-service --desired-count 3
```

For more information, see [Updating a Service](#) in the *Amazon ECS Developer Guide*.

- For API details, see [UpdateService](#) in *AWS CLI Command Reference*.

update-task-set

The following code example shows how to use `update-task-set`.

AWS CLI

To update a task set

The following `update-task-set` example updates a task set to adjust the scale.

```
aws ecs update-task-set \  
  --cluster MyCluster \  
  --service MyService \  
  --task-set arn:aws:ecs:us-west-2:123456789012:task-set/MyCluster/MyService/ecs-  
svc/1234567890123456789 \  
  --scale value=50,unit=PERCENT
```

Output:

```
{  
  "taskSet": {  
    "id": "ecs-svc/1234567890123456789",  
    "taskSetArn": "arn:aws:ecs:us-west-2:123456789012:task-set/MyCluster/  
MyService/ecs-svc/1234567890123456789",  
    "status": "ACTIVE",  
    "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/  
sample-fargate:2",  
    "computedDesiredCount": 0,  
    "pendingCount": 0,  
    "runningCount": 0,  
    "createdAt": 1557128360.711,  
    "updatedAt": 1557129279.914,  
    "launchType": "EC2",  
    "networkConfiguration": {  
      "awsvpcConfiguration": {  
        "subnets": [  
          "subnet-12344321"  
        ],  
        "securityGroups": [  

```

```
        "sg-12344321"  
      ],  
      "assignPublicIp": "DISABLED"  
    }  
  },  
  "loadBalancers": [],  
  "serviceRegistries": [],  
  "scale": {  
    "value": 50.0,  
    "unit": "PERCENT"  
  },  
  "stabilityStatus": "STABILIZING",  
  "stabilityStatusAt": 1557129279.914  
}  
}
```

- For API details, see [UpdateTaskSet](#) in *AWS CLI Command Reference*.

Amazon EFS examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon EFS.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-file-system

The following code example shows how to use `create-file-system`.

AWS CLI

To create an encrypted file system

The following `create-file-system` example creates an encrypted file system using the default CMK. It also adds the tag `Name=my-file-system`.

```
aws efs create-file-system \  
  --performance-mode generalPurpose \  
  --throughput-mode bursting \  
  --encrypted \  
  --tags Key=Name,Value=my-file-system
```

Output:

```
{  
  "OwnerId": "123456789012",  
  "CreationToken": "console-d7f56c5f-e433-41ca-8307-9d9c0example",  
  "FileSystemId": "fs-c7a0456e",  
  "FileSystemArn": "arn:aws:elasticfilesystem:us-west-2:123456789012:file-system/  
fs-48499b4d",  
  "CreationTime": 1595286880.0,  
  "LifecycleState": "creating",  
  "Name": "my-file-system",  
  "NumberOfMountTargets": 0,  
  "SizeInBytes": {  
    "Value": 0,  
    "ValueInIA": 0,  
    "ValueInStandard": 0  
  },  
  "PerformanceMode": "generalPurpose",  
  "Encrypted": true,  
  "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/a59b3472-e62c-42e4-  
adcf-30d92example",  
  "ThroughputMode": "bursting",  
  "Tags": [  
    {  
      "Key": "Name",  
      "Value": "my-file-system"  
    }  
  ]  
}
```


For more information, see [Creating Amazon EFS file systems](#) in the *Amazon Elastic File System User Guide*.

- For API details, see [CreateFileSystem](#) in *AWS CLI Command Reference*.

create-mount-target

The following code example shows how to use `create-mount-target`.

AWS CLI

To create a mount target

The following `create-mount-target` example creates a mount target for the specified file system.

```
aws efs create-mount-target \  
  --file-system-id fs-c7a0456e \  
  --subnet-id subnet-02bf4c428bexample \  
  --security-groups sg-068f739363example
```

Output:

```
{  
  "OwnerId": "123456789012",  
  "MountTargetId": "fsmt-f9a14450",  
  "FileSystemId": "fs-c7a0456e",  
  "SubnetId": "subnet-02bf4c428bexample",  
  "LifecycleState": "creating",  
  "IpAddress": "10.0.1.24",  
  "NetworkInterfaceId": "eni-02d542216aexample",  
  "AvailabilityZoneId": "use2-az2",  
  "AvailabilityZoneName": "us-east-2b",  
  "VpcId": "vpc-0123456789abcdef0"  
}
```

For more information, see [Creating mount targets](#) in the *Amazon Elastic File System User Guide*.

- For API details, see [CreateMountTarget](#) in *AWS CLI Command Reference*.

delete-file-system

The following code example shows how to use `delete-file-system`.

AWS CLI

To delete a file system

The following `delete-file-system` example deletes the specified file system.

```
aws efs delete-file-system \  
  --file-system-id fs-c7a0456e
```

This command produces no output.

For more information, see [Deleting an Amazon EFS file system](#) in the *Amazon Elastic File System User Guide*.

- For API details, see [DeleteFileSystem](#) in *AWS CLI Command Reference*.

delete-mount-target

The following code example shows how to use `delete-mount-target`.

AWS CLI

To delete a mount target

The following `delete-mount-target` example deletes the specified mount target.

```
aws efs delete-mount-target \  
  --mount-target-id fsmt-f9a14450
```

This command produces no output.

For more information, see [Creating mount targets](#) in the *Amazon Elastic File System User Guide*.

- For API details, see [DeleteMountTarget](#) in *AWS CLI Command Reference*.

describe-file-systems

The following code example shows how to use `describe-file-systems`.

AWS CLI

To describe a file system

The following `describe-file-systems` example describes the specified file system.

```
aws efs describe-file-systems \  
  --file-system-id fs-c7a0456e
```

Output:

```
{  
  "FileSystems": [  
    {  
      "OwnerId": "123456789012",  
      "CreationToken": "console-d7f56c5f-e433-41ca-8307-9d9c0example",  
      "FileSystemId": "fs-c7a0456e",  
      "FileSystemArn": "arn:aws:elasticfilesystem:us-west-2:123456789012:file-  
system/fs-48499b4d",  
      "CreationTime": 1595286880.0,  
      "LifecycleState": "available",  
      "Name": "my-file-system",  
      "NumberOfMountTargets": 3,  
      "SizeInBytes": {  
        "Value": 6144,  
        "Timestamp": 1600991437.0,  
        "ValueInIA": 0,  
        "ValueInStandard": 6144  
      },  
      "PerformanceMode": "generalPurpose",  
      "Encrypted": true,  
      "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/a59b3472-e62c-42e4-  
adcf-30d92example",  
      "ThroughputMode": "bursting",  
      "Tags": [  
        {  
          "Key": "Name",  
          "Value": "my-file-system"  
        }  
      ]  
    }  
  ]  
}
```

For more information, see [Managing Amazon EFS file systems](#) in the *Amazon Elastic File System User Guide*.

- For API details, see [DescribeFileSystems](#) in *AWS CLI Command Reference*.

describe-mount-targets

The following code example shows how to use `describe-mount-targets`.

AWS CLI

To describe a mount target

The following `describe-mount-targets` example describes the specified mount target.

```
aws efs describe-mount-targets \  
  --mount-target-id fsmt-f9a14450
```

Output:

```
{  
  "MountTargets": [  
    {  
      "OwnerId": "123456789012",  
      "MountTargetId": "fsmt-f9a14450",  
      "FileSystemId": "fs-c7a0456e",  
      "SubnetId": "subnet-02bf4c428bexample",  
      "LifeCycleState": "creating",  
      "IpAddress": "10.0.1.24",  
      "NetworkInterfaceId": "eni-02d542216aexample",  
      "AvailabilityZoneId": "use2-az2",  
      "AvailabilityZoneName": "us-east-2b",  
      "VpcId": "vpc-0123456789abcdef0"  
    }  
  ]  
}
```

For more information, see [Creating mount targets](#) in the *Amazon Elastic File System User Guide*.

- For API details, see [DescribeMountTargets](#) in *AWS CLI Command Reference*.

describe-tags

The following code example shows how to use `describe-tags`.

AWS CLI

To describe the tags for a file system

The following `describe-tags` example describes the tags for the specified file system.

```
aws efs describe-tags \  
  --file-system-id fs-c7a0456e
```

Output:

```
{  
  "Tags": [  
    {  
      "Key": "Name",  
      "Value": "my-file-system"  
    },  
    {  
      "Key": "Department",  
      "Value": "Business Intelligence"  
    }  
  ]  
}
```

For more information, see [Managing file system tags](#) in the *Amazon Elastic File System User Guide*.

- For API details, see [DescribeTags](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To retrieve the tags for a resource

The following `list-tags-for-resource` example retrieves the tags associated with the specified file system.

```
aws efs list-tags-for-resource \  
  --resource-id fs-c7a0456e
```

Output:

```
{  
  "Tags": [  
    {  
      "Key": "Name",  
      "Value": "my-file-system"  
    },  
    {  
      "Key": "Department",  
      "Value": "Business Intelligence"  
    }  
  ]  
}
```

For more information, see [Managing file system tags](#) in the *Amazon Elastic File System User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To tag a resource

The following tag-resource example adds the tag Department=Business Intelligence to the specified file system.

```
aws efs tag-resource \  
  --resource-id fs-c7a0456e \  
  --tags Key=Department,Value="Business Intelligence"
```

This command produces no output.

For more information, see [Managing file system tags](#) in the *Amazon Elastic File System User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag from a resource

The following `untag-resource` example removes the tag with the `Department` tag key from the specified file system.

```
aws efs untag-resource \  
  --resource-id fs-c7a0456e \  
  --tag-keys Department
```

This command produces no output.

For more information, see [Managing file system tags](#) in the *Amazon Elastic File System User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

Amazon EKS examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon EKS.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-encryption-config

The following code example shows how to use `associate-encryption-config`.

AWS CLI

To associates an encryption configuration to an existing cluster

The following `associate-encryption-config` example enable's encryption on an existing EKS clusters that do not already have encryption enabled.

```
aws eks associate-encryption-config \
  --cluster-name my-eks-cluster \
  --encryption-config '[{"resources":["secrets"],"provider":
{"keyArn":"arn:aws:kms:region-code:account:key/key"}}]'
```

Output:

```
{
  "update": {
    "id": "3141b835-8103-423a-8e68-12c2521ffa4d",
    "status": "InProgress",
    "type": "AssociateEncryptionConfig",
    "params": [
      {
        "type": "EncryptionConfig",
        "value": "[{"resources":["secrets"],"provider":{"keyArn":
\\\"arn:aws:kms:region-code:account:key/key\\\"}]}"
      }
    ],
    "createdAt": "2024-03-14T11:01:26.297000-04:00",
    "errors": []
  }
}
```

For more information, see [Enabling secret encryption on an existing cluster](#) in the *Amazon EKS User Guide*.

- For API details, see [AssociateEncryptionConfig](#) in *AWS CLI Command Reference*.

associate-identity-provider-config

The following code example shows how to use `associate-identity-provider-config`.

AWS CLI

Associate identity provider to your Amazon EKS Cluster

The following `associate-identity-provider-config` example associates an identity provider to your Amazon EKS Cluster.

```
aws eks associate-identity-provider-config \
  --cluster-name my-eks-cluster \
  --oidc 'identityProviderConfigName=my-identity-provider,issuerUrl=https://
oidc.eks.us-east-2.amazonaws.com/
id/38D6A4619A0A69E342B113ED7F1A7652,clientId=kubernetes,usernameClaim=email,usernamePrefix=
username-prefix,groupsClaim=my-claim,groupsPrefix=my-groups-
prefix,requiredClaims={Claim1=value1,Claim2=value2}' \
  --tags env=dev
```

Output:

```
{
  "update": {
    "id": "8c6c1bef-61fe-42ac-a242-89412387b8e7",
    "status": "InProgress",
    "type": "AssociateIdentityProviderConfig",
    "params": [
      {
        "type": "IdentityProviderConfig",
        "value": "[{\"type\":\"oidc\",\"name\":\"my-identity-provider\"}]"
      }
    ],
    "createdAt": "2024-04-11T13:46:49.648000-04:00",
    "errors": []
  },
  "tags": {
    "env": "dev"
  }
}
```

For more information, see [Authenticate users for your cluster from an OpenID Connect identity provider - Associate an OIDC identity provider](#) in the *Amazon EKS User Guide*.

- For API details, see [AssociateIdentityProviderConfig](#) in *AWS CLI Command Reference*.

create-addon

The following code example shows how to use create-addon.

AWS CLI

Example 1: To create an Amazon EKS add-on with default compatible version for the respective EKS cluster version

The following create-addon example command creates an Amazon EKS add-on with default compatible version for the respective EKS cluster version.

```
aws eks create-addon \  
  --cluster-name my-eks-cluster \  
  --addon-name my-eks-addon \  
  --service-account-role-arn arn:aws:iam::111122223333:role/role-name
```

Output:

```
{  
  "addon": {  
    "addonName": "my-eks-addon",  
    "clusterName": "my-eks-cluster",  
    "status": "CREATING",  
    "addonVersion": "v1.15.1-eksbuild.1",  
    "health": {  
      "issues": []  
    },  
    "addonArn": "arn:aws:eks:us-east-2:111122223333:addon/my-eks-cluster/my-eks-addon/1ec71ee1-b9c2-8915-4e17-e8be0a55a149",  
    "createdAt": "2024-03-14T12:20:03.264000-04:00",  
    "modifiedAt": "2024-03-14T12:20:03.283000-04:00",  
    "serviceAccountRoleArn": "arn:aws:iam::111122223333:role/role-name",  
    "tags": {}  
  }  
}
```

For more information, see [Managing Amazon EKS add-ons - Creating an add-on](#) in the *Amazon EKS User Guide*.

Example 2: To create an Amazon EKS add-on with specific add-on version

The following `create-addon` example command creates an Amazon EKS add-on with specific add-on version.

```
aws eks create-addon \  
  --cluster-name my-eks-cluster \  
  --addon-name my-eks-addon \  
  --service-account-role-arn arn:aws:iam::111122223333:role/role-name \  
  --addon-version v1.16.4-eksbuild.2
```

Output:

```
{  
  "addon": {  
    "addonName": "my-eks-addon",  
    "clusterName": "my-eks-cluster",  
    "status": "CREATING",  
    "addonVersion": "v1.16.4-eksbuild.2",  
    "health": {  
      "issues": []  
    },  
    "addonArn": "arn:aws:eks:us-east-2:111122223333:addon/my-eks-cluster/my-eks-addon/34c71ee6-7738-6c8b-c6bd-3921a176b5ff",  
    "createdAt": "2024-03-14T12:30:24.507000-04:00",  
    "modifiedAt": "2024-03-14T12:30:24.521000-04:00",  
    "serviceAccountRoleArn": "arn:aws:iam::111122223333:role/role-name",  
    "tags": {}  
  }  
}
```

For more information, see [Managing Amazon EKS add-ons - Creating an add-on](#) in the *Amazon EKS User Guide*.

Example 3: To create an Amazon EKS add-on with custom configuration values and resolve conflicts details

The following `create-addon` example command creates an Amazon EKS add-on with custom configuration values and resolves conflicts details.

```
aws eks create-addon \
  --cluster-name my-eks-cluster \
  --addon-name my-eks-addon \
  --service-account-role-arn arn:aws:iam::111122223333:role/role-name \
  --addon-version v1.16.4-eksbuild.2 \
  --configuration-values '{"resources":{"limits":{"cpu":"100m"}}}' \
  --resolve-conflicts OVERWRITE
```

Output:

```
{
  "addon": {
    "addonName": "my-eks-addon",
    "clusterName": "my-eks-cluster",
    "status": "CREATING",
    "addonVersion": "v1.16.4-eksbuild.2",
    "health": {
      "issues": []
    },
    "addonArn": "arn:aws:eks:us-east-2:111122223333:addon/my-eks-cluster/my-eks-addon/a6c71ee9-0304-9237-1be8-25af1b0f1ffb",
    "createdAt": "2024-03-14T12:35:58.313000-04:00",
    "modifiedAt": "2024-03-14T12:35:58.327000-04:00",
    "serviceAccountRoleArn": "arn:aws:iam::111122223333:role/role-name",
    "tags": {},
    "configurationValues": "{\"resources\":{\"limits\":{\"cpu\":\"100m\"}}}"
  }
}
```

For more information, see [Managing Amazon EKS add-ons - Creating an add-on](#) in the *Amazon EKS User Guide*.

Example 4: To create an Amazon EKS add-on with custom JSON configuration values file

The following `create-addon` example command creates an Amazon EKS add-on with custom configuration values and resolve conflicts details.

```
aws eks create-addon \
  --cluster-name my-eks-cluster \
  --addon-name my-eks-addon \
  --service-account-role-arn arn:aws:iam::111122223333:role/role-name \
  --addon-version v1.16.4-eksbuild.2 \
```

```
--configuration-values 'file://configuration-values.json' \
--resolve-conflicts OVERWRITE \
--tags '{"eks-addon-key-1": "value-1" , "eks-addon-key-2": "value-2"}'
```

Contents of configuration-values.json:

```
{
  "resources": {
    "limits": {
      "cpu": "150m"
    }
  },
  "env": {
    "AWS_VPC_K8S_CNI_LOGLEVEL": "ERROR"
  }
}
```

Output:

```
{
  "addon": {
    "addonName": "my-eks-addon",
    "clusterName": "my-eks-cluster",
    "status": "CREATING",
    "addonVersion": "v1.16.4-eksbuild.2",
    "health": {
      "issues": []
    },
    "addonArn": "arn:aws:eks:us-east-2:111122223333:addon/my-eks-cluster/my-eks-addon/d8c71ef8-fbd8-07d0-fb32-6a7be19eeced",
    "createdAt": "2024-03-14T13:10:51.763000-04:00",
    "modifiedAt": "2024-03-14T13:10:51.777000-04:00",
    "serviceAccountRoleArn": "arn:aws:iam::111122223333:role/role-name",
    "tags": {
      "eks-addon-key-1": "value-1",
      "eks-addon-key-2": "value-2"
    },
    "configurationValues": "{\n  \"resources\": {\n    \"limits\": {\n      \"cpu\": \"150m\"\n    }\n  },\n  \"env\": {\n    \"AWS_VPC_K8S_CNI_LOGLEVEL\": \"ERROR\"\n  }\n}"
  }
}
```

For more information, see [Managing Amazon EKS add-ons - Creating an add-on](#) in the *Amazon EKS User Guide*.

Example 5: To create an Amazon EKS add-on with custom YAML configuration values file

The following `create-addon` example command creates an Amazon EKS add-on with custom configuration values and resolve conflicts details.

```
aws eks create-addon \  
  --cluster-name my-eks-cluster \  
  --addon-name my-eks-addon \  
  --service-account-role-arn arn:aws:iam::111122223333:role/role-name \  
  --addon-version v1.16.4-eksbuild.2 \  
  --configuration-values 'file://configuration-values.yaml' \  
  --resolve-conflicts OVERWRITE \  
  --tags '{"eks-addon-key-1": "value-1" , "eks-addon-key-2": "value-2"}'
```

Contents of `configuration-values.yaml`:

```
resources:  
  limits:  
    cpu: '100m'  
env:  
  AWS_VPC_K8S_CNI_LOGLEVEL: 'DEBUG'
```

Output:

```
{  
  "addon": {  
    "addonName": "my-eks-addon",  
    "clusterName": "my-eks-cluster",  
    "status": "CREATING",  
    "addonVersion": "v1.16.4-eksbuild.2",  
    "health": {  
      "issues": []  
    },  
    "addonArn": "arn:aws:eks:us-east-2:111122223333:addon/my-eks-cluster/my-eks-addon/d4c71efb-3909-6f36-a548-402cd4b5d59e",  
    "createdAt": "2024-03-14T13:15:45.220000-04:00",  
    "modifiedAt": "2024-03-14T13:15:45.237000-04:00",  
    "serviceAccountRoleArn": "arn:aws:iam::111122223333:role/role-name",  
    "tags": {  
      "eks-addon-key-3": "value-3",  
    }  
  }  
}
```

```
        "eks-addon-key-4": "value-4"
    },
    "configurationValues": "resources:\n    limits:\n        cpu: '100m'\n    nenv:\n    AWS_VPC_K8S_CNI_LOGLEVEL: 'INFO'"
}
}
```

For more information, see [Managing Amazon EKS add-ons - Creating an add-on](#) in the *Amazon EKS User Guide*.

- For API details, see [CreateAddon](#) in *AWS CLI Command Reference*.

create-cluster

The following code example shows how to use `create-cluster`.

AWS CLI

To create a new cluster

This example command creates a cluster named `prod` in your default region.

Command:

```
aws eks create-cluster --name prod \
--role-arn arn:aws:iam::012345678910:role/eks-service-role-
AWSServiceRoleForAmazonEKS-J7ONKE3BQ4PI \
--resources-vpc-config subnetIds=subnet-6782e71e,subnet-
e7e761ac,securityGroupIds=sg-6979fe18
```

Output:

```
{
  "cluster": {
    "name": "prod",
    "arn": "arn:aws:eks:us-west-2:012345678910:cluster/prod",
    "createdAt": 1527808069.147,
    "version": "1.10",
    "roleArn": "arn:aws:iam::012345678910:role/eks-service-role-
AWSServiceRoleForAmazonEKS-J7ONKE3BQ4PI",
    "resourcesVpcConfig": {
      "subnetIds": [
        "subnet-6782e71e",
```

```

        "subnet-e7e761ac"
      ],
      "securityGroupIds": [
        "sg-6979fe18"
      ],
      "vpcId": "vpc-950809ec"
    },
    "status": "CREATING",
    "certificateAuthority": {}
  }
}

```

To create a new cluster with private endpoint access and logging enabled

This example command creates a cluster named `example` in your default region with public endpoint access disabled, private endpoint access enabled, and all logging types enabled.

Command:

```

aws eks create-cluster --name example --kubernetes-version 1.12 \
--role-arn arn:aws:iam::012345678910:role/example-cluster-ServiceRole-1XWBQWYSFRE2Q \
--resources-vpc-config
  subnetIds=subnet-0a188dccd2f9a632f,subnet-09290d93da4278664,subnet-0f21dd86e0e91134a,subnet-
 \
--logging '{"clusterLogging":[{"types":
["api","audit","authenticator","controllerManager","scheduler"],"enabled":true}]}'

```

Output:

```

{
  "cluster": {
    "name": "example",
    "arn": "arn:aws:eks:us-west-2:012345678910:cluster/example",
    "createdAt": 1565804921.901,
    "version": "1.12",
    "roleArn": "arn:aws:iam::012345678910:role/example-cluster-
ServiceRole-1XWBQWYSFRE2Q",
    "resourcesVpcConfig": {
      "subnetIds": [
        "subnet-0a188dccd2f9a632f",
        "subnet-09290d93da4278664",
        "subnet-0f21dd86e0e91134a",

```



```
        "subnet-0173dead68481a583",
        "subnet-051f70a57ed6fcab6",
        "subnet-01322339c5c7de9b4"
    ],
    "securityGroupIds": [
        "sg-0c5b580845a031c10"
    ],
    "vpcId": "vpc-0f622c01f68d4afec",
    "endpointPublicAccess": false,
    "endpointPrivateAccess": true
},
"logging": {
    "clusterLogging": [
        {
            "types": [
                "api",
                "audit",
                "authenticator",
                "controllerManager",
                "scheduler"
            ],
            "enabled": true
        }
    ]
},
"status": "CREATING",
"certificateAuthority": {},
"platformVersion": "eks.3"
}
}
```

- For API details, see [CreateCluster](#) in *AWS CLI Command Reference*.

create-fargate-profile

The following code example shows how to use `create-fargate-profile`.

AWS CLI

Example 1: Create EKS Fargate Profile for a selector with a namespace

The following `create-fargate-profile` example creates an EKS Fargate Profile for a selector with a namespace.

```
aws eks create-fargate-profile \  
  --cluster-name my-eks-cluster \  
  --pod-execution-role-arn arn:aws:iam::111122223333:role/role-name \  
  --fargate-profile-name my-fargate-profile \  
  --selectors '[{"namespace": "default"}]'
```

Output:

```
{  
  "fargateProfile": {  
    "fargateProfileName": "my-fargate-profile",  
    "fargateProfileArn": "arn:aws:eks:us-east-2:111122223333:fargateprofile/my-  
eks-cluster/my-fargate-profile/a2c72bca-318e-abe8-8ed1-27c6d4892e9e",  
    "clusterName": "my-eks-cluster",  
    "createdAt": "2024-03-19T12:38:47.368000-04:00",  
    "podExecutionRoleArn": "arn:aws:iam::111122223333:role/role-name",  
    "subnets": [  
      "subnet-09d912bb63ef21b9a",  
      "subnet-04ad87f71c6e5ab4d",  
      "subnet-0e2907431c9988b72"  
    ],  
    "selectors": [  
      {  
        "namespace": "default"  
      }  
    ],  
    "status": "CREATING",  
    "tags": {}  
  }  
}
```

For more information, see [AWS Fargate profile - Creating a Fargate profile](#) in the *Amazon EKS User Guide*.

Example 2: Create EKS Fargate Profile for a selector with a namespace and labels

The following `create-fargate-profile` example creates an EKS Fargate Profile for a selector with a namespace and labels.

```
aws eks create-fargate-profile \  
  --cluster-name my-eks-cluster \  
  --pod-execution-role-arn arn:aws:iam::111122223333:role/role-name \  
  --fargate-profile-name my-fargate-profile \  
  --selectors '[{"namespace": "default", "labels": {"key": "value"}}]'
```

```
--fargate-profile-name my-fargate-profile \  
--selectors '[{"namespace": "default", "labels": {"labelname1":  
"labelvalue1"}}]'
```

Output:

```
{  
  "fargateProfile": {  
    "fargateProfileName": "my-fargate-profile",  
    "fargateProfileArn": "arn:aws:eks:us-east-2:111122223333:fargateprofile/my-  
eks-cluster/my-fargate-profile/88c72bc7-e8a4-fa34-44e4-2f1397224bb3",  
    "clusterName": "my-eks-cluster",  
    "createdAt": "2024-03-19T12:33:48.125000-04:00",  
    "podExecutionRoleArn": "arn:aws:iam::111122223333:role/role-name",  
    "subnets": [  
      "subnet-09d912bb63ef21b9a",  
      "subnet-04ad87f71c6e5ab4d",  
      "subnet-0e2907431c9988b72"  
    ],  
    "selectors": [  
      {  
        "namespace": "default",  
        "labels": {  
          "labelname1": "labelvalue1"  
        }  
      }  
    ],  
    "status": "CREATING",  
    "tags": {}  
  }  
}
```

For more information, see [AWS Fargate profile - Creating a Fargate profile](#) in the *Amazon EKS User Guide*.

Example 3: Create EKS Fargate Profile for a selector with a namespace and labels, along with IDs of subnets to launch a Pod into.

The following `create-fargate-profile` example create EKS Fargate Profile for a selector with a namespace and labels, along with IDs of subnets to launch a Pod into.

```
aws eks create-fargate-profile \  
--cluster-name my-eks-cluster \  

```

```
--pod-execution-role-arn arn:aws:iam::111122223333:role/role-name \
--fargate-profile-name my-fargate-profile \
--selectors '[{"namespace": "default", "labels": {"labelname1":
"labelvalue1"}}]' \
--subnets ["subnet-09d912bb63ef21b9a", "subnet-04ad87f71c6e5ab4d",
"subnet-0e2907431c9988b72"]'
```

Output:

```
{
  "fargateProfile": {
    "fargateProfileName": "my-fargate-profile",
    "fargateProfileArn": "arn:aws:eks:us-east-2:111122223333:fargateprofile/my-
eks-cluster/my-fargate-profile/e8c72bc8-e87b-5eb6-57cb-ed4fe57577e3",
    "clusterName": "my-eks-cluster",
    "createdAt": "2024-03-19T12:35:58.640000-04:00",
    "podExecutionRoleArn": "arn:aws:iam::111122223333:role/role-name",
    "subnets": [
      "subnet-09d912bb63ef21b9a",
      "subnet-04ad87f71c6e5ab4d",
      "subnet-0e2907431c9988b72"
    ],
    "selectors": [
      {
        "namespace": "default",
        "labels": {
          "labelname1": "labelvalue1"
        }
      }
    ],
    "status": "CREATING",
    "tags": {}
  }
}
```

For more information, see [AWS Fargate profile - Creating a Fargate profile](#) in the *Amazon EKS User Guide*.

Example 4: Create EKS Fargate Profile for a selector with multiple namespace and labels, along with IDs of subnets to launch a Pod into

The following `create-fargate-profile` example creates an EKS Fargate Profile for a selector with multiple namespace and labels, along with IDs of subnets to launch a Pod into.

```
aws eks create-fargate-profile \
  --cluster-name my-eks-cluster \
  --pod-execution-role-arn arn:aws:iam::111122223333:role/role-name \
  --fargate-profile-name my-fargate-profile \
  --selectors '[{"namespace": "default1", "labels": {"labelname1": "labelvalue1",
"labelname2": "labelvalue2"}}, {"namespace": "default2", "labels": {"labelname1":
"labelvalue1", "labelname2": "labelvalue2"}}]' \
  --subnets ["subnet-09d912bb63ef21b9a", "subnet-04ad87f71c6e5ab4d",
"subnet-0e2907431c9988b72"] \
  --tags '{"eks-fargate-profile-key-1": "value-1" , "eks-fargate-profile-key-2":
"value-2"}'
```

Output:

```
{
  "fargateProfile": {
    "fargateProfileName": "my-fargate-profile",
    "fargateProfileArn": "arn:aws:eks:us-east-2:111122223333:fargateprofile/my-
eks-cluster/my-fargate-profile/4cc72bbf-b766-8ee6-8d29-e62748feb3cd",
    "clusterName": "my-eks-cluster",
    "createdAt": "2024-03-19T12:15:55.271000-04:00",
    "podExecutionRoleArn": "arn:aws:iam::111122223333:role/role-name",
    "subnets": [
      "subnet-09d912bb63ef21b9a",
      "subnet-04ad87f71c6e5ab4d",
      "subnet-0e2907431c9988b72"
    ],
    "selectors": [
      {
        "namespace": "default1",
        "labels": {
          "labelname2": "labelvalue2",
          "labelname1": "labelvalue1"
        }
      },
      {
        "namespace": "default2",
        "labels": {
          "labelname2": "labelvalue2",
          "labelname1": "labelvalue1"
        }
      }
    ]
  },
}
```

```

    "status": "CREATING",
    "tags": {
      "eks-fargate-profile-key-2": "value-2",
      "eks-fargate-profile-key-1": "value-1"
    }
  }
}

```

For more information, see [AWS Fargate profile - Creating a Fargate profile](#) in the *Amazon EKS User Guide*.

Example 5: Create EKS Fargate Profile with a wildcard selector for namespaces and labels, along with IDs of subnets to launch a Pod into

The following `create-fargate-profile` example creates an EKS Fargate Profile for a selector with multiple namespace and labels, along with IDs of subnets to launch a Pod into.

```

aws eks create-fargate-profile \
  --cluster-name my-eks-cluster \
  --pod-execution-role-arn arn:aws:iam::111122223333:role/role-name \
  --fargate-profile-name my-fargate-profile \
  --selectors '[{"namespace": "prod*", "labels": {"labelname*?": "*value1"}}, {"namespace": "*dev*", "labels": {"labelname*?": "*value*"}}]' \
  --subnets ["subnet-09d912bb63ef21b9a", "subnet-04ad87f71c6e5ab4d", "subnet-0e2907431c9988b72"] \
  --tags '{"eks-fargate-profile-key-1": "value-1" , "eks-fargate-profile-key-2": "value-2"}'

```

Output:

```

{
  "fargateProfile": {
    "fargateProfileName": "my-fargate-profile",
    "fargateProfileArn": "arn:aws:eks:us-east-2:111122223333:fargateprofile/my-eks-cluster/my-fargate-profile/e8c72bd6-5966-0bfe-b77b-1802893e5a6f",
    "clusterName": "my-eks-cluster",
    "createdAt": "2024-03-19T13:05:20.550000-04:00",
    "podExecutionRoleArn": "arn:aws:iam::111122223333:role/role-name",
    "subnets": [
      "subnet-09d912bb63ef21b9a",
      "subnet-04ad87f71c6e5ab4d",
      "subnet-0e2907431c9988b72"
    ]
  }
}

```

```
    ],
    "selectors": [
      {
        "namespace": "prod*",
        "labels": {
          "labelname*?": "*value1"
        }
      },
      {
        "namespace": "*dev*",
        "labels": {
          "labelname*?": "*value*"
        }
      }
    ],
    "status": "CREATING",
    "tags": {
      "eks-fargate-profile-key-2": "value-2",
      "eks-fargate-profile-key-1": "value-1"
    }
  }
}
```

For more information, see [AWS Fargate profile - Creating a Fargate profile](#) in the *Amazon EKS User Guide*.

- For API details, see [CreateFargateProfile](#) in *AWS CLI Command Reference*.

create-nodegroup

The following code example shows how to use create-nodegroup.

AWS CLI

Example 1: Creates a managed node group for an Amazon EKS cluster

The following create-nodegroup example creates a managed node group for an Amazon EKS cluster.

```
aws eks create-nodegroup \
  --cluster-name my-eks-cluster \
  --nodegroup-name my-eks-nodegroup \
  --node-role arn:aws:iam::111122223333:role/role-name \
```

```
--subnets "subnet-0e2907431c9988b72" "subnet-04ad87f71c6e5ab4d"
"subnet-09d912bb63ef21b9a" \
--scaling-config minSize=1,maxSize=3,desiredSize=1 \
--region us-east-2
```

Output:

```
{
  "nodegroup": {
    "nodegroupName": "my-eks-nodegroup",
    "nodegroupArn": "arn:aws:eks:us-east-2:111122223333:nodegroup/my-eks-
cluster/my-eks-nodegroup/bac7550f-b8b8-5fbb-4f3e-7502a931119e",
    "clusterName": "my-eks-cluster",
    "version": "1.26",
    "releaseVersion": "1.26.12-20240329",
    "createdAt": "2024-04-04T13:19:32.260000-04:00",
    "modifiedAt": "2024-04-04T13:19:32.260000-04:00",
    "status": "CREATING",
    "capacityType": "ON_DEMAND",
    "scalingConfig": {
      "minSize": 1,
      "maxSize": 3,
      "desiredSize": 1
    },
    "instanceTypes": [
      "t3.medium"
    ],
    "subnets": [
      "subnet-0e2907431c9988b72, subnet-04ad87f71c6e5ab4d,
subnet-09d912bb63ef21b9a"
    ],
    "amiType": "AL2_x86_64",
    "nodeRole": "arn:aws:iam::111122223333:role/role-name",
    "diskSize": 20,
    "health": {
      "issues": []
    },
    "updateConfig": {
      "maxUnavailable": 1
    },
    "tags": {}
  }
}
```


For more information, see [Creating a managed node group](#) in the *Amazon EKS User Guide*.

Example 2: Creates a managed node group for an Amazon EKS cluster with custom instance-types and disk-size

The following `create-nodegroup` example creates a managed node group for an Amazon EKS cluster with custom instance-types and disk-size.

```
aws eks create-nodegroup \  
  --cluster-name my-eks-cluster \  
  --nodegroup-name my-eks-nodegroup \  
  --node-role arn:aws:iam::111122223333:role/role-name \  
  --subnets "subnet-0e2907431c9988b72" "subnet-04ad87f71c6e5ab4d" \  
  "subnet-09d912bb63ef21b9a" \  
  --scaling-config minSize=1,maxSize=3,desiredSize=1 \  
  --capacity-type ON_DEMAND \  
  --instance-types 'm5.large' \  
  --disk-size 50 \  
  --region us-east-2
```

Output:

```
{  
  "nodegroup": {  
    "nodegroupName": "my-eks-nodegroup",  
    "nodegroupArn": "arn:aws:eks:us-east-2:111122223333:nodegroup/my-eks-  
cluster/my-eks-nodegroup/c0c7551b-e4f9-73d9-992c-a450fdb82322",  
    "clusterName": "my-eks-cluster",  
    "version": "1.26",  
    "releaseVersion": "1.26.12-20240329",  
    "createdAt": "2024-04-04T13:46:07.595000-04:00",  
    "modifiedAt": "2024-04-04T13:46:07.595000-04:00",  
    "status": "CREATING",  
    "capacityType": "ON_DEMAND",  
    "scalingConfig": {  
      "minSize": 1,  
      "maxSize": 3,  
      "desiredSize": 1  
    },  
    "instanceTypes": [  
      "m5.large"  
    ],  
    "subnets": [  

```

```

        "subnet-0e2907431c9988b72",
        "subnet-04ad87f71c6e5ab4d",
        "subnet-09d912bb63ef21b9a"
    ],
    "amiType": "AL2_x86_64",
    "nodeRole": "arn:aws:iam::111122223333:role/role-name",
    "diskSize": 50,
    "health": {
        "issues": []
    },
    "updateConfig": {
        "maxUnavailable": 1
    },
    "tags": {}
}
}

```

For more information, see [Creating a managed node group](#) in the *Amazon EKS User Guide*.

Example 3: Creates a managed node group for an Amazon EKS cluster with custom instance-types, disk-size, ami-type, capacity-type, update-config, labels, taints and tags.

The following `create-nodegroup` example creates a managed node group for an Amazon EKS cluster with custom instance-types, disk-size, ami-type, capacity-type, update-config, labels, taints and tags.

```

aws eks create-nodegroup \
  --cluster-name my-eks-cluster \
  --nodegroup-name my-eks-nodegroup \
  --node-role arn:aws:iam::111122223333:role/role-name \
  --subnets "subnet-0e2907431c9988b72" "subnet-04ad87f71c6e5ab4d"
"subnet-09d912bb63ef21b9a" \
  --scaling-config minSize=1,maxSize=5,desiredSize=4 \
  --instance-types 't3.large' \
  --disk-size 50 \
  --ami-type AL2_x86_64 \
  --capacity-type SPOT \
  --update-config maxUnavailable=2 \
  --labels '{"my-eks-nodegroup-label-1": "value-1" , "my-eks-nodegroup-label-2":
"value-2"}' \
  --taints '{"key": "taint-key-1" , "value": "taint-value-1", "effect":
"NO_EXECUTE"}' \

```

```
--tags '{"my-eks-nodegroup-key-1": "value-1" , "my-eks-nodegroup-key-2":  
"value-2"}'
```

Output:

```
{  
  "nodegroup": {  
    "nodegroupName": "my-eks-nodegroup",  
    "nodegroupArn": "arn:aws:eks:us-east-2:111122223333:nodegroup/my-eks-  
cluster/my-eks-nodegroup/88c75524-97af-0cb9-a9c5-7c0423ab5314",  
    "clusterName": "my-eks-cluster",  
    "version": "1.26",  
    "releaseVersion": "1.26.12-20240329",  
    "createdAt": "2024-04-04T14:05:07.940000-04:00",  
    "modifiedAt": "2024-04-04T14:05:07.940000-04:00",  
    "status": "CREATING",  
    "capacityType": "SPOT",  
    "scalingConfig": {  
      "minSize": 1,  
      "maxSize": 5,  
      "desiredSize": 4  
    },  
    "instanceTypes": [  
      "t3.large"  
    ],  
    "subnets": [  
      "subnet-0e2907431c9988b72",  
      "subnet-04ad87f71c6e5ab4d",  
      "subnet-09d912bb63ef21b9a"  
    ],  
    "amiType": "AL2_x86_64",  
    "nodeRole": "arn:aws:iam::111122223333:role/role-name",  
    "labels": {  
      "my-eks-nodegroup-label-2": "value-2",  
      "my-eks-nodegroup-label-1": "value-1"  
    },  
    "taints": [  
      {  
        "key": "taint-key-1",  
        "value": "taint-value-1",  
        "effect": "NO_EXECUTE"  
      }  
    ],  
  },  
}
```

```
    "diskSize": 50,
    "health": {
      "issues": []
    },
    "updateConfig": {
      "maxUnavailable": 2
    },
    "tags": {
      "my-eks-nodegroup-key-1": "value-1",
      "my-eks-nodegroup-key-2": "value-2"
    }
  }
}
```

For more information, see [Creating a managed node group](#) in the *Amazon EKS User Guide*.

- For API details, see [CreateNodegroup](#) in *AWS CLI Command Reference*.

delete-addon

The following code example shows how to use `delete-addon`.

AWS CLI

Example 1. To delete an Amazon EKS add-on but preserve the add-on software on the EKS Cluster

The following `delete-addon` example command deletes an Amazon EKS add-on but preserve the add-on software on the EKS Cluster.

```
aws eks delete-addon \
  --cluster-name my-eks-cluster \
  --addon-name my-eks-addon \
  --preserve
```

Output:

```
{
  "addon": {
    "addonName": "my-eks-addon",
    "clusterName": "my-eks-cluster",
    "status": "DELETING",
    "addonVersion": "v1.9.3-eksbuild.7",
```

```

    "health": {
      "issues": []
    },
    "addonArn": "arn:aws:eks:us-east-2:111122223333:addon/my-eks-cluster/my-eks-
addon/a8c71ed3-944e-898b-9167-c763856af4b8",
    "createdAt": "2024-03-14T11:49:09.009000-04:00",
    "modifiedAt": "2024-03-14T12:03:49.776000-04:00",
    "tags": {}
  }
}

```

For more information, see [Managing Amazon EKS add-ons - Deleting an add-on](#) in the *Amazon EKS*.

Example 2. To delete an Amazon EKS add-on and also delete the add-on software from the EKS Cluster

The following `delete-addon` example command deletes an Amazon EKS add-on and also delete the add-on software from the EKS Cluster.

```

aws eks delete-addon \
  --cluster-name my-eks-cluster \
  --addon-name my-eks-addon

```

Output:

```

{
  "addon": {
    "addonName": "my-eks-addon",
    "clusterName": "my-eks-cluster",
    "status": "DELETING",
    "addonVersion": "v1.15.1-eksbuild.1",
    "health": {
      "issues": []
    },
    "addonArn": "arn:aws:eks:us-east-2:111122223333:addon/my-eks-cluster/my-eks-
addon/bac71ed1-ec43-3bb6-88ea-f243cdb58954",
    "createdAt": "2024-03-14T11:45:31.983000-04:00",
    "modifiedAt": "2024-03-14T11:58:40.136000-04:00",
    "serviceAccountRoleArn": "arn:aws:iam::111122223333:role/role-name",
    "tags": {}
  }
}

```

```
}
```

For more information, see [Managing Amazon EKS add-ons - Deleting an add-on](#) in the *Amazon EKS*.

- For API details, see [DeleteAddon](#) in *AWS CLI Command Reference*.

delete-cluster

The following code example shows how to use `delete-cluster`.

AWS CLI

Delete an Amazon EKS cluster control plane

The following `delete-cluster` example deletes an Amazon EKS cluster control plane.

```
aws eks delete-cluster \  
  --name my-eks-cluster
```

Output:

```
{  
  "cluster": {  
    "name": "my-eks-cluster",  
    "arn": "arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster",  
    "createdAt": "2024-03-14T11:31:44.348000-04:00",  
    "version": "1.27",  
    "endpoint": "https://DALSJ343KE23J3RN45653DSKJTT647TYD.y14.us-  
east-2.eks.amazonaws.com",  
    "roleArn": "arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-cluster-  
ServiceRole-zMF6CBakwwbW",  
    "resourcesVpcConfig": {  
      "subnetIds": [  
        "subnet-0fb75d2d8401716e7",  
        "subnet-02184492f67a3d0f9",  
        "subnet-04098063527aab776",  
        "subnet-0e2907431c9988b72",  
        "subnet-04ad87f71c6e5ab4d",  
        "subnet-09d912bb63ef21b9a"  
      ],  
      "securityGroupIds": [  
        "sg-0c1327f6270afbb36"  
      ]  
    }  
  }  
}
```

```
    ],
    "clusterSecurityGroupId": "sg-01c84d09d70f39a7f",
    "vpcId": "vpc-0012b8e1cc0abb17d",
    "endpointPublicAccess": true,
    "endpointPrivateAccess": true,
    "publicAccessCidrs": [
        "0.0.0.0/0"
    ]
},
"kubernetesNetworkConfig": {
    "serviceIpv4Cidr": "10.100.0.0/16",
    "ipFamily": "ipv4"
},
"logging": {
    "clusterLogging": [
        {
            "types": [
                "api",
                "audit",
                "authenticator",
                "controllerManager",
                "scheduler"
            ],
            "enabled": true
        }
    ]
},
"identity": {
    "oidc": {
        "issuer": "https://oidc.eks.us-east-2.amazonaws.com/id/
DALSJ343KE23J3RN45653DSKJTT647TYD"
    }
},
"status": "DELETING",
"certificateAuthority": {
    "data": "XXX_CA_DATA_XXX"
},
"platformVersion": "eks.16",
"tags": {
    "aws:cloudformation:stack-name": "eksctl-my-eks-cluster-cluster",
    "alpha.eksctl.io/cluster-name": "my-eks-cluster",
    "karpenter.sh/discovery": "my-eks-cluster",
```

```

    "aws:cloudformation:stack-id": "arn:aws:cloudformation:us-
east-2:111122223333:stack/eksctl-my-eks-cluster-cluster/e752ea00-e217-11ee-
beae-0a9599c8c7ed",
    "auto-delete": "no",
    "eksctl.cluster.k8s.io/v1alpha1/cluster-name": "my-eks-cluster",
    "EKS-Cluster-Name": "my-eks-cluster",
    "alpha.eksctl.io/cluster-oidc-enabled": "true",
    "aws:cloudformation:logical-id": "ControlPlane",
    "alpha.eksctl.io/eksctl-version": "0.173.0-dev
+a7ee89342.2024-03-01T03:40:57Z",
    "Name": "eksctl-my-eks-cluster-cluster/ControlPlane"
  },
  "accessConfig": {
    "authenticationMode": "API_AND_CONFIG_MAP"
  }
}
}

```

For more information, see [Deleting an Amazon EKS cluster](#) in the *Amazon EKS User Guide*.

- For API details, see [DeleteCluster](#) in *AWS CLI Command Reference*.

delete-fargate-profile

The following code example shows how to use delete-fargate-profile.

AWS CLI

Example 1: Create EKS Fargate Profile for a selector with a namespace

The following delete-fargate-profile example creates an EKS Fargate Profile for a selector with a namespace.

```

aws eks delete-fargate-profile \
  --cluster-name my-eks-cluster \
  --fargate-profile-name my-fargate-profile

```

Output:

```

{
  "fargateProfile": {
    "fargateProfileName": "my-fargate-profile",

```



```

    "fargateProfileArn": "arn:aws:eks:us-east-2:111122223333:fargateprofile/my-
eks-cluster/my-fargate-profile/1ac72bb3-3fc6-2631-f1e1-98bff53bed62",
    "clusterName": "my-eks-cluster",
    "createdAt": "2024-03-19T11:48:39.975000-04:00",
    "podExecutionRoleArn": "arn:aws:iam::111122223333:role/role-name",
    "subnets": [
      "subnet-09d912bb63ef21b9a",
      "subnet-04ad87f71c6e5ab4d",
      "subnet-0e2907431c9988b72"
    ],
    "selectors": [
      {
        "namespace": "default",
        "labels": {
          "foo": "bar"
        }
      }
    ],
    "status": "DELETING",
    "tags": {}
  }
}

```

For more information, see [AWS Fargate profile - Deleting a Fargate](#) in the *Amazon EKS User Guide*.

- For API details, see [DeleteFargateProfile](#) in *AWS CLI Command Reference*.

delete-nodegroup

The following code example shows how to use `delete-nodegroup`.

AWS CLI

Example 1: Delete a managed node group for an Amazon EKS cluster

The following `delete-nodegroup` example deletes a managed node group for an Amazon EKS cluster.

```

aws eks delete-nodegroup \
  --cluster-name my-eks-cluster \
  --nodegroup-name my-eks-nodegroup

```

Output:

```
{
  "nodegroup": {
    "nodegroupName": "my-eks-nodegroup",
    "nodegroupArn": "arn:aws:eks:us-east-2:111122223333:nodegroup/my-eks-
cluster/my-eks-nodegroup/1ec75f5f-0e21-dcc0-b46e-f9c442685cd8",
    "clusterName": "my-eks-cluster",
    "version": "1.26",
    "releaseVersion": "1.26.12-20240329",
    "createdAt": "2024-04-08T13:25:15.033000-04:00",
    "modifiedAt": "2024-04-08T13:25:31.252000-04:00",
    "status": "DELETING",
    "capacityType": "SPOT",
    "scalingConfig": {
      "minSize": 1,
      "maxSize": 5,
      "desiredSize": 4
    },
    "instanceTypes": [
      "t3.large"
    ],
    "subnets": [
      "subnet-0e2907431c9988b72",
      "subnet-04ad87f71c6e5ab4d",
      "subnet-09d912bb63ef21b9a"
    ],
    "amiType": "AL2_x86_64",
    "nodeRole": "arn:aws:iam::111122223333:role/role-name",
    "labels": {
      "my-eks-nodegroup-label-2": "value-2",
      "my-eks-nodegroup-label-1": "value-1"
    },
    "taints": [
      {
        "key": "taint-key-1",
        "value": "taint-value-1",
        "effect": "NO_EXECUTE"
      }
    ],
    "diskSize": 50,
    "health": {
      "issues": []
    },
  },
}
```

```
    "updateConfig": {
      "maxUnavailable": 2
    },
    "tags": {
      "my-eks-nodegroup-key-1": "value-1",
      "my-eks-nodegroup-key-2": "value-2"
    }
  }
}
```

- For API details, see [DeleteNodegroup](#) in *AWS CLI Command Reference*.

deregister-cluster

The following code example shows how to use `deregister-cluster`.

AWS CLI

To deregisters a connected cluster to remove it from the Amazon EKS control plane

The following `deregister-cluster` example deregisters a connected cluster to remove it from the Amazon EKS control plane.

```
aws eks deregister-cluster \
  --name my-eks-anywhere-cluster
```

Output:

```
{
  "cluster": {
    "name": "my-eks-anywhere-cluster",
    "arn": "arn:aws:eks:us-east-2:111122223333:cluster/my-eks-anywhere-cluster",
    "createdAt": "2024-04-12T12:38:37.561000-04:00",
    "status": "DELETING",
    "tags": {},
    "connectorConfig": {
      "activationId": "dfb5ad28-13c3-4e26-8a19-5b2457638c74",
      "activationExpiry": "2024-04-15T12:38:37.082000-04:00",
      "provider": "EKS_ANYWHERE",
      "roleArn": "arn:aws:iam::111122223333:role/AmazonEKSCoordinatorAgentRole"
    }
  }
}
```

```
}

```

For more information, see [Deregistering a cluster](#) in the *Amazon EKS User Guide*.

- For API details, see [DeregisterCluster](#) in *AWS CLI Command Reference*.

describe-addon-versions

The following code example shows how to use `describe-addon-versions`.

AWS CLI

Example 1: List all the available addons for EKS Cluster

The following `describe-addon-versions` example list all the available AWS addons.

```
aws eks describe-addon-versions \
  --query 'sort_by(addons &owner)[].{publisher: publisher, owner: owner,
  addonName: addonName, type: type}' \
  --output table

```

Output:

```
-----
|                                     DescribeAddonVersions
|                                     |
+-----+-----+-----+-----+
|                                     |                                     |
|          addonName                 |          owner          |          publisher
|          |          type            |          |          |
+-----+-----+-----+-----+
| vpc-cni                             | aws                    | eks
|   | networking                      |                         |
| snapshot-controller                 | aws                    | eks
|   | storage                         |                         |
| kube-proxy                          | aws                    | eks
|   | networking                      |                         |
| eks-pod-identity-agent              | aws                    | eks
|   | security                       |                         |
| coredns                             | aws                    | eks
|   | networking                      |                         |

```

aws-mountpoint-s3-csi-driver	aws	s3
storage		
aws-guardduty-agent	aws	eks
security		
aws-efs-csi-driver	aws	eks
storage		
aws-ebs-csi-driver	aws	eks
storage		
amazon-cloudwatch-observability	aws	eks
observability		
adot	aws	eks
observability		
upwind-security_upwind-operator	aws-marketplace	Upwind Security
security		
upbound_universal-crossplane	aws-marketplace	upbound
infra-management		
tetrade-io_istio-distro	aws-marketplace	tetrade-io
policy-management		
teleport_teleport	aws-marketplace	teleport
policy-management		
stormforge_optimize-live	aws-marketplace	StormForge
cost-management		
splunk_splunk-otel-collector-chart	aws-marketplace	Splunk
monitoring		
solo-io_istio-distro	aws-marketplace	Solo.io
service-mesh		
rafay-systems_rafay-operator	aws-marketplace	rafay-systems
kubernetes-management		
new-relic_kubernetes-operator	aws-marketplace	New Relic
observability		
netapp_trident-operator	aws-marketplace	NetApp Inc.
storage		
leaksignal_leakagent	aws-marketplace	leaksignal
monitoring		
kubecost_kubecost	aws-marketplace	kubecost
cost-management		
kong_konnect-ri	aws-marketplace	kong
ingress-service-type		
kasten_k10	aws-marketplace	Kasten by Veeam
data-protection		
haproxy-technologies_kubernetes-ingress-ee	aws-marketplace	HAProxy
Technologies ingress-controller		
groundcover_agent	aws-marketplace	groundcover
monitoring		

```

| grafana-labs_kubernetes-monitoring | aws-marketplace | Grafana Labs
|   monitoring                       |                  |
| factorhouse_kpow                   | aws-marketplace | factorhouse
|   monitoring                       |                  |
| dynatrace_dynatrace-operator       | aws-marketplace | dynatrace
|   monitoring                       |                  |
| datree_engine-pro                  | aws-marketplace | datree
|   policy-management                |                  |
| datadog_operator                   | aws-marketplace | Datadog
|   monitoring                       |                  |
| cribl_cribledge                    | aws-marketplace | Cribl
|   observability                    |                  |
| calyptia_fluent-bit                | aws-marketplace | Calyptia Inc
|   observability                    |                  |
| accuknox_kubearmor                 | aws-marketplace | AccuKnox
|   security                         |                  |
+-----+-----+-----+-----+
+-----+-----+-----+-----+

```

For more information, see [Managing Amazon EKS add-ons - Creating an add-on](#) in the *Amazon EKS User Guide*.

Example 2: List all the available addons for specified Kubernetes version supported for EKS

The following describe-addon-versions example list all the available addons for specified Kubernetes version supported for EKS.

```

aws eks describe-addon-versions \
  --kubernetes-version=1.26 \
  --query 'sort_by(addons &owner)[].{publisher: publisher, owner: owner,
addonName: addonName, type: type}' \
  --output table

```

Output:

```

-----
|                                     DescribeAddonVersions
|                                     |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
|               addonName               | owner | publisher
|               type                   |       |

```

+-----+-----		+-----+-----	
vpc-cni	aws	eks	
networking			
snapshot-controller	aws	eks	
storage			
kube-proxy	aws	eks	
networking			
eks-pod-identity-agent	aws	eks	
security			
coredns	aws	eks	
networking			
aws-mountpoint-s3-csi-driver	aws	s3	
storage			
aws-guardduty-agent	aws	eks	
security			
aws-efs-csi-driver	aws	eks	
storage			
aws-ebs-csi-driver	aws	eks	
storage			
amazon-cloudwatch-observability	aws	eks	
observability			
adot	aws	eks	
observability			
upwind-security_upwind-operator	aws-marketplace	Upwind Security	
security			
tetrade-io_istio-distro	aws-marketplace	tetrade-io	
policy-management			
stormforge_optimize-live	aws-marketplace	StormForge	
cost-management			
splunk_splunk-otel-collector-chart	aws-marketplace	Splunk	
monitoring			
solo-io_istio-distro	aws-marketplace	Solo.io	
service-mesh			
rafay-systems_rafay-operator	aws-marketplace	rafay-systems	
kubernetes-management			
new-relic_kubernetes-operator	aws-marketplace	New Relic	
observability			
netapp_trident-operator	aws-marketplace	NetApp Inc.	
storage			
leaksignal_leakagent	aws-marketplace	leaksignal	
monitoring			
kubecost_kubecost	aws-marketplace	kubecost	
cost-management			

```

| kong_konnect-ri | aws-marketplace | kong
| ingress-service-type |
| haproxy-technologies_kubernetes-ingress-ee | aws-marketplace | HAProxy
Technologies | ingress-controller |
| groundcover_agent | aws-marketplace | groundcover
| monitoring |
| grafana-labs_kubernetes-monitoring | aws-marketplace | Grafana Labs
| monitoring |
| dynatrace_dynatrace-operator | aws-marketplace | dynatrace
| monitoring |
| datadog_operator | aws-marketplace | Datadog
| monitoring |
| cribl_cribledge | aws-marketplace | Cribl
| observability |
| calyptia_fluent-bit | aws-marketplace | Calyptia Inc
| observability |
| accuknox_kubearmor | aws-marketplace | AccuKnox
| security |
+-----+-----+
+-----+-----+

```

For more information, see [Managing Amazon EKS add-ons - Creating an add-on](#) in the *Amazon EKS User Guide*.

Example 3: List all the available vpc-cni addons version for specified Kubernetes version supported for EKS

The following describe-addon-versions example list all the available vpc-cni addons version for specified Kubernetes version supported for EKS.

```

aws eks describe-addon-versions \
  --kubernetes-version=1.26 \
  --addon-name=vpc-cni \
  --query='addons[].addonVersions[].addonVersion'

```

Output:

```

[
  "v1.18.0-eksbuild.1",
  "v1.17.1-eksbuild.1",
  "v1.16.4-eksbuild.2",
  "v1.16.3-eksbuild.2",
  "v1.16.2-eksbuild.1",

```



```
"v1.16.0-eksbuild.1",  
"v1.15.5-eksbuild.1",  
"v1.15.4-eksbuild.1",  
"v1.15.3-eksbuild.1",  
"v1.15.1-eksbuild.1",  
"v1.15.0-eksbuild.2",  
"v1.14.1-eksbuild.1",  
"v1.14.0-eksbuild.3",  
"v1.13.4-eksbuild.1",  
"v1.13.3-eksbuild.1",  
"v1.13.2-eksbuild.1",  
"v1.13.0-eksbuild.1",  
"v1.12.6-eksbuild.2",  
"v1.12.6-eksbuild.1",  
"v1.12.5-eksbuild.2",  
"v1.12.0-eksbuild.2"
```

```
]
```

For more information, see [Managing Amazon EKS add-ons - Creating an add-on](#) in the *Amazon EKS User Guide*.

- For API details, see [DescribeAddonVersions](#) in *AWS CLI Command Reference*.

describe-addon

The following code example shows how to use describe-addon.

AWS CLI

Describe actively running EKS addon in your Amazon EKS cluster

The following describe-addon example actively running EKS addon in your Amazon EKS cluster.

```
aws eks describe-addon \  
  --cluster-name my-eks-cluster \  
  --addon-name vpc-cni
```

Output:

```
{  
  "addon": {  
    "addonName": "vpc-cni",
```

```

    "clusterName": "my-eks-cluster",
    "status": "ACTIVE",
    "addonVersion": "v1.16.4-eksbuild.2",
    "health": {
      "issues": []
    },
    "addonArn": "arn:aws:eks:us-east-2:111122223333:addon/my-eks-cluster/vpc-
cni/0ec71efc-98dd-3203-60b0-4b939b2a5e5f",
    "createdAt": "2024-03-14T13:18:45.417000-04:00",
    "modifiedAt": "2024-03-14T13:18:49.557000-04:00",
    "serviceAccountRoleArn": "arn:aws:iam::111122223333:role/eksctl-my-eks-
cluster-addon-vpc-cni-Role1-Yfakrq0C1UTm",
    "tags": {
      "eks-addon-key-3": "value-3",
      "eks-addon-key-4": "value-4"
    },
    "configurationValues": "resources:\n    limits:\n        cpu: '100m'\nenv:\n
AWS_VPC_K8S_CNI_LOGLEVEL: 'DEBUG' "
  }
}

```

- For API details, see [DescribeAddon](#) in *AWS CLI Command Reference*.

describe-cluster

The following code example shows how to use `describe-cluster`.

AWS CLI

Describe actively running EKS addon in your Amazon EKS cluster

The following `describe-cluster` example actively running EKS addon in your Amazon EKS cluster.

```

aws eks describe-cluster \
  --cluster-name my-eks-cluster

```

Output:

```

{
  "cluster": {
    "name": "my-eks-cluster",

```

```
"arn": "arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster",
"createdAt": "2024-03-14T11:31:44.348000-04:00",
"version": "1.26",
"endpoint": "https://JSA79429HJDASKJDJ8223829MNDNASW.y14.us-
east-2.eks.amazonaws.com",
"roleArn": "arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-cluster-
ServiceRole-zMF6CBakwwbW",
"resourcesVpcConfig": {
  "subnetIds": [
    "subnet-0fb75d2d8401716e7",
    "subnet-02184492f67a3d0f9",
    "subnet-04098063527aab776",
    "subnet-0e2907431c9988b72",
    "subnet-04ad87f71c6e5ab4d",
    "subnet-09d912bb63ef21b9a"
  ],
  "securityGroupIds": [
    "sg-0c1327f6270afbb36"
  ],
  "clusterSecurityGroupId": "sg-01c84d09d70f39a7f",
  "vpcId": "vpc-0012b8e1cc0abb17d",
  "endpointPublicAccess": true,
  "endpointPrivateAccess": true,
  "publicAccessCidrs": [
    "22.19.18.2/32"
  ]
},
"kubernetesNetworkConfig": {
  "serviceIpv4Cidr": "10.100.0.0/16",
  "ipFamily": "ipv4"
},
"logging": {
  "clusterLogging": [
    {
      "types": [
        "api",
        "audit",
        "authenticator",
        "controllerManager",
        "scheduler"
      ],
      "enabled": true
    }
  ]
}
```

```

    },
    "identity": {
      "oidc": {
        "issuer": "https://oidc.eks.us-east-2.amazonaws.com/id/
JSA79429HJDASKJDJ8223829MNDNASW"
      }
    },
    "status": "ACTIVE",
    "certificateAuthority": {
      "data": "CA_DATA_STRING..."
    },
    "platformVersion": "eks.14",
    "tags": {
      "aws:cloudformation:stack-name": "eksctl-my-eks-cluster-cluster",
      "alpha.eksctl.io/cluster-name": "my-eks-cluster",
      "karpenter.sh/discovery": "my-eks-cluster",
      "aws:cloudformation:stack-id": "arn:aws:cloudformation:us-
east-2:111122223333:stack/eksctl-my-eks-cluster-cluster/e752ea00-e217-11ee-
beae-0a9599c8c7ed",
      "auto-delete": "no",
      "eksctl.cluster.k8s.io/v1alpha1/cluster-name": "my-eks-cluster",
      "EKS-Cluster-Name": "my-eks-cluster",
      "alpha.eksctl.io/cluster-oidc-enabled": "true",
      "aws:cloudformation:logical-id": "ControlPlane",
      "alpha.eksctl.io/eksctl-version": "0.173.0-dev
+a7ee89342.2024-03-01T03:40:57Z",
      "Name": "eksctl-my-eks-cluster-cluster/ControlPlane"
    },
    "health": {
      "issues": []
    },
    "accessConfig": {
      "authenticationMode": "API_AND_CONFIG_MAP"
    }
  }
}

```

- For API details, see [DescribeCluster](#) in *AWS CLI Command Reference*.

describe-fargate-profile

The following code example shows how to use `describe-fargate-profile`.

AWS CLI

Describe a Fargate profile

The following `describe-fargate-profile` example describes a Fargate profile.

```
aws eks describe-fargate-profile \  
  --cluster-name my-eks-cluster \  
  --fargate-profile-name my-fargate-profile
```

Output:

```
{  
  "fargateProfile": {  
    "fargateProfileName": "my-fargate-profile",  
    "fargateProfileArn": "arn:aws:eks:us-east-2:111122223333:fargateprofile/my-  
eks-cluster/my-fargate-profile/96c766ce-43d2-f9c9-954c-647334391198",  
    "clusterName": "my-eks-cluster",  
    "createdAt": "2024-04-11T10:42:52.486000-04:00",  
    "podExecutionRoleArn": "arn:aws:iam::111122223333:role/eksctl-my-eks-  
cluster-farga-FargatePodExecutionRole-1htfAaJdJUE0",  
    "subnets": [  
      "subnet-09d912bb63ef21b9a",  
      "subnet-04ad87f71c6e5ab4d",  
      "subnet-0e2907431c9988b72"  
    ],  
    "selectors": [  
      {  
        "namespace": "prod*",  
        "labels": {  
          "labelname*?": "*value1"  
        }  
      },  
      {  
        "namespace": "*dev*",  
        "labels": {  
          "labelname*?": "*value*"  
        }  
      }  
    ],  
    "status": "ACTIVE",  
    "tags": {  
      "eks-fargate-profile-key-2": "value-2",
```

```

    "eks-fargate-profile-key-1": "value-1"
  }
}
}

```

- For API details, see [DescribeFargateProfile](#) in *AWS CLI Command Reference*.

describe-identity-provider-config

The following code example shows how to use `describe-identity-provider-config`.

AWS CLI

Describe an identity provider configuration associated to your Amazon EKS Cluster

The following `describe-identity-provider-config` example describes an identity provider configuration associated to your Amazon EKS Cluster.

```

aws eks describe-identity-provider-config \
  --cluster-name my-eks-cluster \
  --identity-provider-config type=oidc,name=my-identity-provider

```

Output:

```

{
  "identityProviderConfig": {
    "oidc": {
      "identityProviderConfigName": "my-identity-provider",
      "identityProviderConfigArn": "arn:aws:eks:us-
east-2:111122223333:identityproviderconfig/my-eks-cluster/oidc/my-identity-
provider/8ac76722-78e4-cec1-ed76-d49eea058622",
      "clusterName": "my-eks-cluster",
      "issuerUrl": "https://oidc.eks.us-east-2.amazonaws.com/
id/38D6A4619A0A69E342B113ED7F1A7652",
      "clientId": "kubernetes",
      "usernameClaim": "email",
      "usernamePrefix": "my-username-prefix",
      "groupsClaim": "my-claim",
      "groupsPrefix": "my-groups-prefix",
      "requiredClaims": {
        "Claim1": "value1",

```

```
        "Claim2": "value2"
      },
      "tags": {
        "env": "dev"
      },
      "status": "ACTIVE"
    }
  }
}
```

For more information, see [Authenticate users for your cluster from an OpenID Connect identity provider](#) in the *Amazon EKS User Guide*.

- For API details, see [DescribeIdentityProviderConfig](#) in *AWS CLI Command Reference*.

describe-nodegroup

The following code example shows how to use `describe-nodegroup`.

AWS CLI

Describe a managed node group for an Amazon EKS cluster

The following `describe-nodegroup` example describes a managed node group for an Amazon EKS cluster.

```
aws eks describe-nodegroup \
  --cluster-name my-eks-cluster \
  --nodegroup-name my-eks-nodegroup
```

Output:

```
{
  "nodegroup": {
    "nodegroupName": "my-eks-nodegroup",
    "nodegroupArn": "arn:aws:eks:us-east-2:111122223333:nodegroup/my-eks-cluster/my-eks-nodegroup/a8c75f2f-df78-a72f-4063-4b69af3de5b1",
    "clusterName": "my-eks-cluster",
    "version": "1.26",
    "releaseVersion": "1.26.12-20240329",
    "createdAt": "2024-04-08T11:42:10.555000-04:00",
```

```
"modifiedAt": "2024-04-08T11:44:12.402000-04:00",
"status": "ACTIVE",
"capacityType": "ON_DEMAND",
"scalingConfig": {
  "minSize": 1,
  "maxSize": 3,
  "desiredSize": 1
},
"instanceTypes": [
  "t3.medium"
],
"subnets": [
  "subnet-0e2907431c9988b72",
  "subnet-04ad87f71c6e5ab4d",
  "subnet-09d912bb63ef21b9a"
],
"amiType": "AL2_x86_64",
"nodeRole": "arn:aws:iam::111122223333:role/role-name",
"labels": {},
"resources": {
  "autoScalingGroups": [
    {
      "name": "eks-my-eks-nodegroup-a8c75f2f-df78-
a72f-4063-4b69af3de5b1"
    }
  ]
},
"diskSize": 20,
"health": {
  "issues": []
},
"updateConfig": {
  "maxUnavailable": 1
},
"tags": {}
}
}
```

- For API details, see [DescribeNodegroup](#) in *AWS CLI Command Reference*.

describe-update

The following code example shows how to use describe-update.

AWS CLI

Example 1: To describe an update for a cluster

The following describe-update example describes an update for a cluster named.

```
aws eks describe-update \  
  --name my-eks-cluster \  
  --update-id 10bddb13-a71b-425a-b0a6-71cd03e59161
```

Output:

```
{  
  "update": {  
    "id": "10bddb13-a71b-425a-b0a6-71cd03e59161",  
    "status": "Successful",  
    "type": "EndpointAccessUpdate",  
    "params": [  
      {  
        "type": "EndpointPublicAccess",  
        "value": "false"  
      },  
      {  
        "type": "EndpointPrivateAccess",  
        "value": "true"  
      }  
    ],  
    "createdAt": "2024-03-14T10:01:26.297000-04:00",  
    "errors": []  
  }  
}
```

For more information, see [Updating an Amazon EKS cluster Kubernetes version](#) in the *Amazon EKS User Guide*.

Example 2: To describe an update for a cluster

The following describe-update example describes an update for a cluster named.

```
aws eks describe-update \  
  --name my-eks-cluster \  
  --update-id e4994991-4c0f-475a-a040-427e6da52966
```

Output:

```
{
  "update": {
    "id": "e4994991-4c0f-475a-a040-427e6da52966",
    "status": "Successful",
    "type": "AssociateEncryptionConfig",
    "params": [
      {
        "type": "EncryptionConfig",
        "value": "[{\"resources\":[\"secrets\"],\"provider\":{\"keyArn\":
\\\"arn:aws:kms:region-code:account:key/key\\\"}]]\"
      }
    ],
    "createdAt": "2024-03-14T11:01:26.297000-04:00",
    "errors": []
  }
}
```

For more information, see [Updating an Amazon EKS cluster Kubernetes version](#) in the *Amazon EKS User Guide*.

Example 3: To describe an update for a cluster

The following describe-update example describes an update for a cluster named.

```
aws eks describe-update \
  --name my-eks-cluster \
  --update-id b5f0ba18-9a87-4450-b5a0-825e6e84496f
```

Output:

```
{
  "update": {
    "id": "b5f0ba18-9a87-4450-b5a0-825e6e84496f",
    "status": "Successful",
    "type": "VersionUpdate",
    "params": [
      {
        "type": "Version",
        "value": "1.29"
      },
      {

```

```

        "type": "PlatformVersion",
        "value": "eks.1"
      }
    ],
    "createdAt": "2024-03-14T12:05:26.297000-04:00",
    "errors": []
  }
}

```

For more information, see [Updating an Amazon EKS cluster Kubernetes version](#) in the *Amazon EKS User Guide*.

- For API details, see [DescribeUpdate](#) in *AWS CLI Command Reference*.

disassociate-identity-provider-config

The following code example shows how to use `disassociate-identity-provider-config`.

AWS CLI

Disassociate identity provider to your Amazon EKS Cluster

The following `disassociate-identity-provider-config` example disassociates an identity provider to your Amazon EKS Cluster.

```

aws eks disassociate-identity-provider-config \
  --cluster-name my-eks-cluster \
  --identity-provider-config 'type=oidc,name=my-identity-provider'

```

Output:

```

{
  "update": {
    "id": "5f78d14e-c57b-4857-a3e4-cf664ae20949",
    "status": "InProgress",
    "type": "DisassociateIdentityProviderConfig",
    "params": [
      {
        "type": "IdentityProviderConfig",
        "value": "[]"
      }
    ]
  },

```

```
    "createdAt": "2024-04-11T13:53:43.314000-04:00",
    "errors": []
  }
}
```

For more information, see [Authenticate users for your cluster from an OpenID Connect identity provider - Disassociate an OIDC identity provider from your cluster](#) in the *Amazon EKS User Guide*.

- For API details, see [DisassociateIdentityProviderConfig](#) in *AWS CLI Command Reference*.

get-token

The following code example shows how to use `get-token`.

AWS CLI

Example 1: Get an authentication token for an Amazon EKS Cluster named ``my-eks-cluster``

The following `get-token` example gets an authentication token for an Amazon EKS Cluster named `my-eks-cluster`.

```
aws eks get-token \
  --cluster-name my-eks-cluster
```

Output:

```
{
  "kind": "ExecCredential",
  "apiVersion": "client.authentication.k8s.io/v1beta1",
  "spec": {},
  "status": {
    "expirationTimestamp": "2024-04-11T20:59:56Z",
    "token": "k8s-aws-v1.EXAMPLE_TOKEN_DATA_STRING..."
  }
}
```

Example 2: Gets an authentication token for an Amazon EKS Cluster named ``my-eks-cluster`` by assuming this roleARN for credentials when signing the token

The following `get-token` example gets an authentication token for an Amazon EKS Cluster named `my-eks-cluster` by assuming this roleARN for credentials when signing the token.

```
aws eks get-token \  
  --cluster-name my-eks-cluster \  
  --role-arn arn:aws:iam::111122223333:role/eksctl-EKS-Linux-Cluster-v1-24-  
cluster-ServiceRole-j1k7AfTIQtnM
```

Output:

```
{  
  "kind": "ExecCredential",  
  "apiVersion": "client.authentication.k8s.io/v1beta1",  
  "spec": {},  
  "status": {  
    "expirationTimestamp": "2024-04-11T21:05:26Z",  
    "token": "k8s-aws-v1.EXAMPLE_TOKEN_DATA_STRING..."  
  }  
}
```

- For API details, see [GetToken](#) in *AWS CLI Command Reference*.

list-addons

The following code example shows how to use `list-addons`.

AWS CLI

List all the installed add-ons in your Amazon EKS cluster named `my-eks-cluster``

The following `list-addons` example lists all the installed add-ons in your Amazon EKS cluster named `my-eks-cluster`.

```
aws eks list-addons \  
  --cluster-name my-eks-cluster
```

Output:

```
{  
  "addons": [  
    "kube-proxy",  
    "vpc-cni"  
  ]  
}
```

- For API details, see [ListAddons](#) in *AWS CLI Command Reference*.

list-clusters

The following code example shows how to use `list-clusters`.

AWS CLI

To list all the installed add-ons in your Amazon EKS cluster named ``my-eks-cluster``

The following `list-clusters` example lists all the installed add-ons in your Amazon EKS cluster named `my-eks-cluster`.

```
aws eks list-clusters
```

Output:

```
{
  "clusters": [
    "prod",
    "qa",
    "stage",
    "my-eks-cluster"
  ]
}
```

- For API details, see [ListClusters](#) in *AWS CLI Command Reference*.

list-fargate-profiles

The following code example shows how to use `list-fargate-profiles`.

AWS CLI

To list all the fargate profiles in your Amazon EKS cluster named ``my-eks-cluster``

The following `list-fargate-profiles` example lists all the fargate profiles in your Amazon EKS cluster named `my-eks-cluster`.

```
aws eks list-fargate-profiles \
```

```
--cluster-name my-eks-cluster
```

Output:

```
{
  "fargateProfileNames": [
    "my-fargate-profile"
  ]
}
```

- For API details, see [ListFargateProfiles](#) in *AWS CLI Command Reference*.

list-identity-provider-configs

The following code example shows how to use `list-identity-provider-configs`.

AWS CLI**List identity providers associated to an Amazon EKS Cluster**

The following `list-identity-provider-configs` example lists identity provider associated to an Amazon EKS Cluster.

```
aws eks list-identity-provider-configs \
  --cluster-name my-eks-cluster
```

Output:

```
{
  "identityProviderConfigs": [
    {
      "type": "oidc",
      "name": "my-identity-provider"
    }
  ]
}
```

For more information, see [Authenticate users for your cluster from an OpenID Connect identity provider](#) in the *Amazon EKS User Guide*.

- For API details, see [ListIdentityProviderConfigs](#) in *AWS CLI Command Reference*.

list-nodegroups

The following code example shows how to use `list-nodegroups`.

AWS CLI

List all the node groups in an Amazon EKS cluster

The following `list-nodegroups` example list all the node groups in an Amazon EKS cluster.

```
aws eks list-nodegroups \
  --cluster-name my-eks-cluster
```

Output:

```
{
  "nodegroups": [
    "my-eks-managed-node-group",
    "my-eks-nodegroup"
  ]
}
```

- For API details, see [ListNodegroups](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

Example 1: To list all the tags for an Amazon EKS Cluster ARN

The following `list-tags-for-resource` example lists all the tags for an Amazon EKS Cluster ARN.

```
aws eks list-tags-for-resource \
  --resource-arn arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster
```

Output:

```
{
  "tags": {
```



```

    "aws:cloudformation:stack-name": "eksctl-my-eks-cluster-cluster",
    "alpha.eksctl.io/cluster-name": "my-eks-cluster",
    "karpenter.sh/discovery": "my-eks-cluster",
    "aws:cloudformation:stack-id": "arn:aws:cloudformation:us-
east-2:111122223333:stack/eksctl-my-eks-cluster-cluster/e752ea00-e217-11ee-
beae-0a9599c8c7ed",
    "auto-delete": "no",
    "eksctl.cluster.k8s.io/v1alpha1/cluster-name": "my-eks-cluster",
    "EKS-Cluster-Name": "my-eks-cluster",
    "alpha.eksctl.io/cluster-oidc-enabled": "true",
    "aws:cloudformation:logical-id": "ControlPlane",
    "alpha.eksctl.io/eksctl-version": "0.173.0-dev
+a7ee89342.2024-03-01T03:40:57Z",
    "Name": "eksctl-my-eks-cluster-cluster/ControlPlane"
  }
}

```

Example 2: To list all the tags for an Amazon EKS Node group ARN

The following `list-tags-for-resource` example lists all the tags for an Amazon EKS Node group ARN.

```

aws eks list-tags-for-resource \
  --resource-arn arn:aws:eks:us-east-2:111122223333:nodegroup/my-eks-cluster/my-
eks-managed-node-group/60c71ed2-2cfb-020f-a5f4-ad32477f198c

```

Output:

```

{
  "tags": {
    "aws:cloudformation:stack-name": "eksctl-my-eks-cluster-nodegroup-my-eks-
managed-node-group",
    "aws:cloudformation:stack-id": "arn:aws:cloudformation:us-
east-2:111122223333:stack/eksctl-my-eks-cluster-nodegroup-my-eks-managed-node-group/
eaa20310-e219-11ee-b851-0ab9ad8228ff",
    "eksctl.cluster.k8s.io/v1alpha1/cluster-name": "my-eks-cluster",
    "EKS-Cluster-Name": "my-eks-cluster",
    "alpha.eksctl.io/nodegroup-type": "managed",
    "NodeGroup Name 1": "my-eks-managed-node-group",
    "k8s.io/cluster-autoscaler/enabled": "true",
    "nodegroup-role": "worker",
    "alpha.eksctl.io/cluster-name": "my-eks-cluster",
    "alpha.eksctl.io/nodegroup-name": "my-eks-managed-node-group",

```

```

    "karpenter.sh/discovery": "my-eks-cluster",
    "NodeGroup Name 2": "AmazonLinux-Linux-Managed-NG-v1-26-v1",
    "auto-delete": "no",
    "k8s.io/cluster-autoscaler/my-eks-cluster": "owned",
    "aws:cloudformation:logical-id": "ManagedNodeGroup",
    "alpha.eksctl.io/eksctl-version": "0.173.0-dev
+a7ee89342.2024-03-01T03:40:57Z"
  }
}

```

Example 3: To list all the tags on an Amazon EKS Fargate profile ARN

The following `list-tags-for-resource` example lists all the tags for an Amazon EKS Fargate profile ARN.

```

aws eks list-tags-for-resource \
  --resource-arn arn:aws:eks:us-east-2:111122223333:fargateprofile/my-eks-cluster/
my-fargate-profile/d6c76780-e541-0725-c816-36754cab734b

```

Output:

```

{
  "tags": {
    "eks-fargate-profile-key-2": "value-2",
    "eks-fargate-profile-key-1": "value-1"
  }
}

```

Example 4: To list all the tags for an Amazon EKS Add-on ARN

The following `list-tags-for-resource` example lists all the tags for an Amazon EKS Add-on ARN.

```

aws eks list-tags-for-resource \
  --resource-arn arn:aws:eks:us-east-2:111122223333:addon/my-eks-cluster/vpc-
cni/0ec71efc-98dd-3203-60b0-4b939b2a5e5f

```

Output:

```

{
  "tags": {

```

```
    "eks-addon-key-2": "value-2",  
    "eks-addon-key-1": "value-1"  
  }  
}
```

Example 5: To list all the tags for an Amazon EKS OIDC identity provider ARN

The following `list-tags-for-resource` example lists all the tags for an Amazon EKS OIDC identity provider ARN.

```
aws eks list-tags-for-resource \  
  --resource-arn arn:aws:eks:us-east-2:111122223333:identityproviderconfig/my-eks-  
cluster/oidc/my-identity-provider/8ac76722-78e4-cec1-ed76-d49eea058622
```

Output:

```
{  
  "tags": {  
    "my-identity-provider": "test"  
  }  
}
```

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

list-update

The following code example shows how to use `list-update`.

AWS CLI

Example 1: To lists the updates associated with an Amazon EKS Cluster name

The following `list-updates` example lists all the update IDs for an Amazon EKS Cluster name.

```
aws eks list-updates \  
  --name my-eks-cluster
```

Output:

```
{
```

```
"updateIds": [  
  "5f78d14e-c57b-4857-a3e4-cf664ae20949",  
  "760e5a3f-adad-48c7-88d3-7ac283c09c26",  
  "cd4ec863-bc55-47d5-a377-3971502f529b",  
  "f12657ce-e869-4f17-b158-a82ab8b7d937"  
]  
}
```

Example 2: To list all the update IDs for an Amazon EKS Node group

The following `list-updates` example lists all the update IDs for an Amazon EKS Node group.

```
aws eks list-updates \  
  --name my-eks-cluster \  
  --nodegroup-name my-eks-managed-node-group
```

Output:

```
{  
  "updateIds": [  
    "8c6c1bef-61fe-42ac-a242-89412387b8e7"  
  ]  
}
```

Example 3: To list all the update IDs on an Amazon EKS Add-one

The following `list-updates` example lists all the update IDs for an Amazon EKS Add-on.

```
aws eks list-updates \  
  --name my-eks-cluster \  
  --addon-name vpc-cni
```

Output:

```
{  
  "updateIds": [  
    "9cdba8d4-79fb-3c83-afe8-00b508d33268"  
  ]  
}
```

- For API details, see [ListUpdate](#) in *AWS CLI Command Reference*.

list-updates

The following code example shows how to use `list-updates`.

AWS CLI

To list the updates for a cluster

This example command lists the current updates for a cluster named `example` in your default region.

Command:

```
aws eks list-updates --name example
```

Output:

```
{
  "updateIds": [
    "10bddb13-a71b-425a-b0a6-71cd03e59161"
  ]
}
```

- For API details, see [ListUpdates](#) in *AWS CLI Command Reference*.

register-cluster

The following code example shows how to use `register-cluster`.

AWS CLI

Example 1: Register an external EKS_ANYWHERE Kubernetes cluster to Amazon EKS

The following `register-cluster` example registers an external EKS_ANYWHERE Kubernetes cluster to Amazon EKS.

```
aws eks register-cluster \
  --name my-eks-anywhere-cluster \
  --connector-config 'roleArn=arn:aws:iam::111122223333:role/
AmazonEKSCoordinatorAgentRole,provider=EKS_ANYWHERE'
```

Output:

```
{
  "cluster": {
    "name": "my-eks-anywhere-cluster",
    "arn": "arn:aws:eks:us-east-2:111122223333:cluster/my-eks-anywhere-cluster",
    "createdAt": "2024-04-12T12:38:37.561000-04:00",
    "status": "PENDING",
    "tags": {},
    "connectorConfig": {
      "activationId": "xxxxxxxxACTIVATION_IDxxxxxxxx",
      "activationCode": "xxxxxxxxACTIVATION_CODExxxxxxxx",
      "activationExpiry": "2024-04-15T12:38:37.082000-04:00",
      "provider": "EKS_ANYWHERE",
      "roleArn": "arn:aws:iam::111122223333:role/AmazonEKSCheckpointAgentRole"
    }
  }
}
```

For more information, see [Connecting an external cluster](#) in the *Amazon EKS User Guide*.

Example 2: Register any external Kubernetes cluster to Amazon EKS

The following `register-cluster` example registers an external `EKS_ANYWHERE` Kubernetes cluster to Amazon EKS.

```
aws eks register-cluster \
  --name my-eks-anywhere-cluster \
  --connector-config 'roleArn=arn:aws:iam::111122223333:role/
AmazonEKSCheckpointAgentRole,provider=OTHER'
```

Output:

```
{
  "cluster": {
    "name": "my-onprem-k8s-cluster",
    "arn": "arn:aws:eks:us-east-2:111122223333:cluster/my-onprem-k8s-cluster",
    "createdAt": "2024-04-12T12:42:10.861000-04:00",
    "status": "PENDING",
    "tags": {},
    "connectorConfig": {
      "activationId": "xxxxxxxxACTIVATION_IDxxxxxxxx",
      "activationCode": "xxxxxxxxACTIVATION_CODExxxxxxxx",
      "activationExpiry": "2024-04-15T12:42:10.339000-04:00",

```

```
        "provider": "OTHER",
        "roleArn": "arn:aws:iam::111122223333:role/AmazonEKSCollectorAgentRole"
    }
}
```

For more information, see [Connecting an external cluster](#) in the *Amazon EKS User Guide*.

- For API details, see [RegisterCluster](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

Example 1: To add the specified tags to an Amazon EKS Cluster

The following `tag-resource` example adds the specified tags to an Amazon EKS Cluster.

```
aws eks tag-resource \  
  --resource-arn arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster \  
  --tag 'my-eks-cluster-test-1=test-value-1,my-eks-cluster-dev-1=dev-value-2'
```

This command produces no output.

Example 2: To add the specified tags to an Amazon EKS Node group

The following `tag-resource` example adds the specified tags to an Amazon EKS Node group.

```
aws eks tag-resource \  
  --resource-arn arn:aws:eks:us-east-2:111122223333:nodegroup/my-eks-cluster/my-eks-managed-node-group/60c71ed2-2cfb-020f-a5f4-ad32477f198c \  
  --tag 'my-eks-nodegroup-test-1=test-value-1,my-eks-nodegroup-dev-1=dev-value-2'
```

This command produces no output.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

Example 1: To deletes the specified tags from an Amazon EKS Cluster

The following `untag-resource` example deletes the specified tags from an Amazon EKS Cluster.

```
aws eks untag-resource \  
  --resource-arn arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster \  
  --tag-keys "my-eks-cluster-test-1" "my-eks-cluster-dev-1"
```

This command produces no output.

Example 2: To deletes the specified tags from an Amazon EKS Node group

The following `untag-resource` example deletes the specified tags from an Amazon EKS Node group.

```
aws eks untag-resource \  
  --resource-arn arn:aws:eks:us-east-2:111122223333:nodegroup/my-eks-cluster/my-  
eks-managed-node-group/60c71ed2-2cfb-020f-a5f4-ad32477f198c \  
  --tag-keys "my-eks-nodegroup-test-1" "my-eks-nodegroup-dev-1"
```

This command produces no output.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-addon

The following code example shows how to use `update-addon`.

AWS CLI

Example 1. To update an Amazon EKS add-on with service account role ARN

The following `update-addon` example command updates an Amazon EKS add-on with service account role ARN.

```
aws eks update-addon \  
  --cluster-name my-eks-cluster \  
  --addon-name vpc-cni \  
  --service-account-role-arn arn:aws:iam::111122223333:role/my-eks-vpc-cni-role
```



```
--service-account-role-arn arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-
addon-vpc-cni-Role1-Yfakrq0C1UTm
```

Output:

```
{
  "update": {
    "id": "c00d2de2-c2e4-3d30-929e-46b8edec2ce4",
    "status": "InProgress",
    "type": "AddonUpdate",
    "params": [
      {
        "type": "ServiceAccountRoleArn",
        "value": "arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-
addon-vpc-cni-Role1-Yfakrq0C1UTm"
      }
    ],
    "updatedAt": "2024-04-12T16:04:55.614000-04:00",
    "errors": []
  }
}
```

For more information, see [Managing Amazon EKS add-ons - Updating an add-on](#) in the *Amazon EKS User Guide*.

Example 2. To update an Amazon EKS add-on with specific add-on version

The following update-addon example command updates an Amazon EKS add-on with specific add-on version.

```
aws eks update-addon \
  --cluster-name my-eks-cluster \
  --addon-name vpc-cni \
  --service-account-role-arn arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-
addon-vpc-cni-Role1-Yfakrq0C1UTm \
  --addon-version v1.16.4-eksbuild.2
```

Output:

```
{
  "update": {
    "id": "f58dc0b0-2b18-34bd-bc6a-e4abc0011f36",
```

```

    "status": "InProgress",
    "type": "AddonUpdate",
    "params": [
      {
        "type": "AddonVersion",
        "value": "v1.16.4-eksbuild.2"
      },
      {
        "type": "ServiceAccountRoleArn",
        "value": "arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-
addon-vpc-cni-Role1-Yfakrq0C1UTm"
      }
    ],
    "createdAt": "2024-04-12T16:07:16.550000-04:00",
    "errors": []
  }
}

```

For more information, see [Managing Amazon EKS add-ons - Updating an add-on](#) in the *Amazon EKS User Guide*.

Example 3. To update an Amazon EKS add-on with custom configuration values and resolve conflicts details

The following update-addon example command updates an Amazon EKS add-on with custom configuration values and resolve conflicts details.

```

aws eks update-addon \
  --cluster-name my-eks-cluster \
  --addon-name vpc-cni \
  --service-account-role-arn arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-
addon-vpc-cni-Role1-Yfakrq0C1UTm \
  --addon-version v1.16.4-eksbuild.2 \
  --configuration-values '{"resources": {"limits":{"cpu":"100m"}, "requests":
{"cpu":"50m"}}}' \
  --resolve-conflicts PRESERVE

```

Output:

```

{
  "update": {
    "id": "cd9f2173-a8d8-3004-a90f-032f14326520",

```

```

    "status": "InProgress",
    "type": "AddonUpdate",
    "params": [
      {
        "type": "AddonVersion",
        "value": "v1.16.4-eksbuild.2"
      },
      {
        "type": "ServiceAccountRoleArn",
        "value": "arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-
addon-vpc-cni-Role1-Yfakrq0C1UTm"
      },
      {
        "type": "ResolveConflicts",
        "value": "PRESERVE"
      },
      {
        "type": "ConfigurationValues",
        "value": "{\"resources\": {\"limits\": {\"cpu\": \"100m\"}, \"requests
\": {\"cpu\": \"50m\"}}}"
      }
    ],
    "createdAt": "2024-04-12T16:16:27.363000-04:00",
    "errors": []
  }
}

```

For more information, see [Managing Amazon EKS add-ons - Updating an add-on](#) in the *Amazon EKS User Guide*.

Example 4. To update an Amazon EKS add-on with custom JSON configuration values file

The following update-addon example command updates an Amazon EKS add-on with custom JSON configuration values and resolve conflicts details.

```

aws eks update-addon \
  --cluster-name my-eks-cluster \
  --addon-name vpc-cni \
  --service-account-role-arn arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-
addon-vpc-cni-Role1-Yfakrq0C1UTm \
  --addon-version v1.17.1-eksbuild.1 \
  --configuration-values 'file://configuration-values.json' \
  --resolve-conflicts PRESERVE

```

Contents of configuration-values.json:

```
{
  "resources": {
    "limits": {
      "cpu": "100m"
    },
    "requests": {
      "cpu": "50m"
    }
  },
  "env": {
    "AWS_VPC_K8S_CNI_LOGLEVEL": "ERROR"
  }
}
```

Output:

```
{
  "update": {
    "id": "6881a437-174f-346b-9a63-6e91763507cc",
    "status": "InProgress",
    "type": "AddonUpdate",
    "params": [
      {
        "type": "AddonVersion",
        "value": "v1.17.1-eksbuild.1"
      },
      {
        "type": "ServiceAccountRoleArn",
        "value": "arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-
addon-vpc-cni-Role1-Yfakrq0C1UTm"
      },
      {
        "type": "ResolveConflicts",
        "value": "PRESERVE"
      },
      {
        "type": "ConfigurationValues",
        "value": "{\n  \"resources\": {\n    \"limits\": {\n
      \"cpu\": \"100m\"\n    },\n    \"requests\": {\n      \"cpu\": \"50m
      \"\n    }\n  },\n  \"env\": {\n    \"AWS_VPC_K8S_CNI_LOGLEVEL\": \"ERROR
      \"\n  }\n}"
      }
    ]
  }
}
```

```

    }
  ],
  "createdAt": "2024-04-12T16:22:55.519000-04:00",
  "errors": []
}
}

```

For more information, see [Managing Amazon EKS add-ons - Updating an add-on](#) in the *Amazon EKS User Guide*.

Example 5. To update an Amazon EKS add-on with custom YAML configuration values file

The following update-addon example command updates an Amazon EKS add-on with custom YAML configuration values and resolve conflicts details.

```

aws eks update-addon \
  --cluster-name my-eks-cluster \
  --addon-name vpc-cni \
  --service-account-role-arn arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-
addon-vpc-cni-Role1-Yfakrq0C1UTm \
  --addon-version v1.18.0-eksbuild.1 \
  --configuration-values 'file://configuration-values.yaml' \
  --resolve-conflicts PRESERVE

```

Contents of configuration-values.yaml:

```

resources:
  limits:
    cpu: '100m'
  requests:
    cpu: '50m'
env:
  AWS_VPC_K8S_CNI_LOGLEVEL: 'DEBUG'

```

Output:

```

{
  "update": {
    "id": "a067a4c9-69d0-3769-ace9-d235c5b16701",
    "status": "InProgress",
    "type": "AddonUpdate",
    "params": [

```

```

    {
      "type": "AddonVersion",
      "value": "v1.18.0-eksbuild.1"
    },
    {
      "type": "ServiceAccountRoleArn",
      "value": "arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-
addon-vpc-cni-Role1-Yfakrq0C1UTm"
    },
    {
      "type": "ResolveConflicts",
      "value": "PRESERVE"
    },
    {
      "type": "ConfigurationValues",
      "value": "resources:\n      limits:\n          cpu: '100m'\n
requests:\n      cpu: '50m'\nenv:\n      AWS_VPC_K8S_CNI_LOGLEVEL: 'DEBUG'"
    }
  ],
  "createdAt": "2024-04-12T16:25:07.212000-04:00",
  "errors": []
}
}

```

For more information, see [Managing Amazon EKS add-ons - Updating an add-on](#) in the *Amazon EKS User Guide*.

- For API details, see [UpdateAddon](#) in *AWS CLI Command Reference*.

update-cluster-config

The following code example shows how to use `update-cluster-config`.

AWS CLI

To update cluster endpoint access

This example command updates a cluster to disable endpoint public access and enable private endpoint access.

Command:

```
aws eks update-cluster-config --name example \
```

```
--resources-vpc-config endpointPublicAccess=false,endpointPrivateAccess=true
```

Output:

```
{
  "update": {
    "id": "ec883c93-2e9e-407c-a22f-8f6fa6e67d4f",
    "status": "InProgress",
    "type": "EndpointAccessUpdate",
    "params": [
      {
        "type": "EndpointPublicAccess",
        "value": "false"
      },
      {
        "type": "EndpointPrivateAccess",
        "value": "true"
      }
    ],
    "createdAt": 1565806986.506,
    "errors": []
  }
}
```

To enable logging for a cluster

This example command enables all cluster control plane logging types for a cluster named example.

Command:

```
aws eks update-cluster-config --name example \
--logging '{"clusterLogging":[{"types":
["api","audit","authenticator","controllerManager","scheduler"],"enabled":true}]}'
```

Output:

```
{
  "update": {
    "id": "7551c64b-1d27-4b1e-9f8e-c45f056eb6fd",
    "status": "InProgress",
    "type": "LoggingUpdate",
```

```

    "params": [
      {
        "type": "ClusterLogging",
        "value": "{\"clusterLogging\": [{\"types\": [\"api\", \"audit\",
\\\"authenticator\\\", \"controllerManager\\\", \"scheduler\\\"], \"enabled\": true}]}"
      }
    ],
    "createdAt": 1565807210.37,
    "errors": []
  }
}

```

- For API details, see [UpdateClusterConfig](#) in *AWS CLI Command Reference*.

update-cluster-version

The following code example shows how to use `update-cluster-version`.

AWS CLI

To updates an Amazon EKS cluster named `my-eks-cluster` to the specified Kubernetes version

The following `update-cluster-version` example updates an Amazon EKS cluster to the specified Kubernetes version.

```

aws eks update-cluster-version \
  --name my-eks-cluster \
  --kubernetes-version 1.27

```

Output:

```

{
  "update": {
    "id": "e4091a28-ea14-48fd-a8c7-975aeb469e8a",
    "status": "InProgress",
    "type": "VersionUpdate",
    "params": [
      {
        "type": "Version",
        "value": "1.27"
      }
    ],
  }
}

```



```
    {
      "type": "PlatformVersion",
      "value": "eks.16"
    }
  ],
  "createdAt": "2024-04-12T16:56:01.082000-04:00",
  "errors": []
}
```

For more information, see [Updating an Amazon EKS cluster Kubernetes version](#) in the *Amazon EKS User Guide*.

- For API details, see [UpdateClusterVersion](#) in *AWS CLI Command Reference*.

update-kubeconfig

The following code example shows how to use `update-kubeconfig`.

AWS CLI

Example 1: Configures your kubectl by creating or updating the kubeconfig so that you can connect to an Amazon EKS Cluster named `my-eks-cluster``

The following `update-kubeconfig` example configures your kubectl by creating or updating the kubeconfig so that you can connect to an Amazon EKS Cluster named `my-eks-cluster`.

```
aws eks update-kubeconfig \
  --name my-eks-cluster
```

Output:

```
Updated context arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster in /Users/
xxx/.kube/config
```

For more information, see [Creating or updating a kubeconfig file for an Amazon EKS cluster](#) in the *Amazon EKS User Guide*.

Example 2: Configures your kubectl by creating or updating the kubeconfig (with `role-arn` option to assume a role for cluster authentication) so that you can connect to an Amazon EKS Cluster named `my-eks-cluster``

The following `update-kubeconfig` example configures your `kubectl` by creating or updating the kubeconfig (with `role-arn` option to assume a role for cluster authentication) so that you can connect to an Amazon EKS Cluster named `my-eks-cluster`.

```
aws eks update-kubeconfig \  
  --name my-eks-cluster \  
  --role-arn arn:aws:iam::111122223333:role/eksctl-EKS-Linux-Cluster-v1-24-  
cluster-ServiceRole-j1k7AfTIQtnM
```

Output:

```
Updated context arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster in /Users/  
xxx/.kube/config
```

For more information, see [Creating or updating a kubeconfig file for an Amazon EKS cluster](#) in the *Amazon EKS User Guide*.

Example 3: Configures your `kubectl` by creating or updating the kubeconfig (with `role-arn` option to assume a role for cluster authentication along with custom cluster alias and user-alias) so that you can connect to an Amazon EKS Cluster named `my-eks-cluster`

The following `update-kubeconfig` example configures your `kubectl` by creating or updating the kubeconfig (with `role-arn` option to assume a role for cluster authentication along with custom cluster alias and user-alias) so that you can connect to an Amazon EKS Cluster named `my-eks-cluster`.

```
aws eks update-kubeconfig \  
  --name my-eks-cluster \  
  --role-arn arn:aws:iam::111122223333:role/eksctl-EKS-Linux-Cluster-v1-24-  
cluster-ServiceRole-j1k7AfTIQtnM \  
  --alias stage-eks-cluster \  
  --user-alias john
```

Output:

```
Updated context stage-eks-cluster in /Users/dubaria/.kube/config
```

For more information, see [Creating or updating a kubeconfig file for an Amazon EKS cluster](#) in the *Amazon EKS User Guide*.

Example 4: Print kubeconfig file entries for review and configures your kubectl so that you can connect to an Amazon EKS Cluster named `my-eks-cluster`

The following update-kubeconfig example configures your kubectl by creating or updating the kubeconfig (with role-arn option to assume a role for cluster authentication along with custom cluster alias and user-alias) so that you can connect to an Amazon EKS Cluster named my-eks-cluster.

```
aws eks update-kubeconfig \  
  --name my-eks-cluster \  
  --role-arn arn:aws:iam::111122223333:role/eksctl-EKS-Linux-Cluster-v1-24-  
cluster-ServiceRole-j1k7AfTIQtnM \  
  --alias stage-eks-cluster \  
  --user-alias john \  
  --verbose
```

Output:

```
Updated context stage-eks-cluster in /Users/dubaria/.kube/config  
Entries:  
  
context:  
cluster: arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster  
user: john  
name: stage-eks-cluster  
  
name: john  
user:  
exec:  
  apiVersion: client.authentication.k8s.io/v1beta1  
  args:  
  - --region  
  - us-east-2  
  - eks  
  - get-token  
  - --cluster-name  
  - my-eks-cluster  
  - --output  
  - json  
  - --role  
  - arn:aws:iam::111122223333:role/eksctl-EKS-Linux-Cluster-v1-24-cluster-  
ServiceRole-j1k7AfTIQtnM
```

```

command: aws

cluster:
certificate-authority-data: xxx_CA_DATA_xxx
server: https://DALSJ343KE23J3RN45653DSKJTT647TYD.y14.us-east-2.eks.amazonaws.com
name: arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster

```

For more information, see [Creating or updating a kubeconfig file for an Amazon EKS cluster](#) in the *Amazon EKS User Guide*.

- For API details, see [UpdateKubeconfig](#) in *AWS CLI Command Reference*.

update-nodegroup-config

The following code example shows how to use `update-nodegroup-config`.

AWS CLI

Example 1: Update a managed node group to add new labels and taint to EKS worker node for an Amazon EKS cluster

The following `update-nodegroup-config` example updates a managed node group to add new labels and taint to EKS worker node for an Amazon EKS cluster.

```

aws eks update-nodegroup-config \
  --cluster-name my-eks-cluster \
  --nodegroup-name my-eks-nodegroup \
  --labels 'add0rUpdateLabels={my-eks-nodegroup-label-1=value-1,my-eks-nodegroup-label-2=value-2}' \
  --taints 'add0rUpdateTaints=[{key=taint-key-1,value=taint-value-1,effect=NO_EXECUTE}]'

```

Output:

```

{
  "update": {
    "id": "e66d21d3-bd8b-3ad1-a5aa-b196dc08c7c1",
    "status": "InProgress",
    "type": "ConfigUpdate",
    "params": [
      {
        "type": "LabelsToAdd",

```

```

        "value": "{\"my-eks-nodegroup-label-2\": \"value-2\", \"my-eks-
nodegroup-label-1\": \"value-1\"}"
      },
      {
        "type": "TaintsToAdd",
        "value": "[{\"effect\": \"NO_EXECUTE\", \"value\": \"taint-value-1\",
\\\"key\\\": \"taint-key-1\"}]"
      }
    ],
    "createdAt": "2024-04-08T12:05:19.161000-04:00",
    "errors": []
  }
}

```

For more information, see [Updating a managed node group](#) in the *Amazon EKS User Guide*.

Example 2: Update a managed node group to remove labels and taint for the EKS worker node for an Amazon EKS cluster

The following `update-nodegroup-config` example updates a managed node group to remove labels and taint for the EKS worker node for an Amazon EKS cluster.

```

aws eks update-nodegroup-config \
  --cluster-name my-eks-cluster \
  --nodegroup-name my-eks-nodegroup \
  --labels 'removeLabels=my-eks-nodegroup-label-1, my-eks-nodegroup-label-2' \
  --taints 'removeTaints=[{key=taint-key-1,value=taint-
value-1,effect=NO_EXECUTE}]'

```

Output:

```

{
  "update": {
    "id": "67a08692-9e59-3ace-a916-13929f44cec3",
    "status": "InProgress",
    "type": "ConfigUpdate",
    "params": [
      {
        "type": "LabelsToRemove",
        "value": "[\"my-eks-nodegroup-label-1\", \"my-eks-nodegroup-
label-2\"]"
      }
    ],
  },
}

```

```

    {
      "type": "TaintsToRemove",
      "value": "[{\"effect\":\"NO_EXECUTE\"},{\"value\":\"taint-value-1\"},
\\\"key\\\":\\\"taint-key-1\\\"}]"
    }
  ],
  "createdAt": "2024-04-08T12:17:31.817000-04:00",
  "errors": []
}
}

```

For more information, see [Updating a managed node group](#) in the *Amazon EKS User Guide*.

Example 3: Update a managed node group to remove and add labels and taint for the EKS worker node for an Amazon EKS cluster

The following `update-nodegroup-config` example updates a managed node group to remove and add labels and taint for the EKS worker node for an Amazon EKS cluster.

```

aws eks update-nodegroup-config \
  --cluster-name my-eks-cluster \
  --nodegroup-name my-eks-nodegroup \
  --labels 'addOrUpdateLabels={my-eks-nodegroup-new-label-1=new-value-1,my-eks-
nodegroup-new-label-2=new-value-2},removeLabels=my-eks-nodegroup-label-1, my-eks-
nodegroup-label-2' \
  --taints 'addOrUpdateTaints=[{key=taint-new-key-1,value=taint-new-
value-1,effect=PREFER_NO_SCHEDULE}],removeTaints=[{key=taint-key-1,value=taint-
value-1,effect=NO_EXECUTE}]'

```

Output:

```

{
  "update": {
    "id": "4a9c8c45-6ac7-3115-be71-d6412a2339b7",
    "status": "InProgress",
    "type": "ConfigUpdate",
    "params": [
      {
        "type": "LabelsToAdd",
        "value": "{\"my-eks-nodegroup-new-label-1\":\"new-value-1\",\"my-
eks-nodegroup-new-label-2\":\"new-value-2\"}"
      },
    ],
  },
}

```

```

    {
      "type": "LabelsToRemove",
      "value": "[\"my-eks-nodegroup-label-1\", \"my-eks-nodegroup-
label-2\"]"
    },
    {
      "type": "TaintsToAdd",
      "value": "[{\"effect\": \"PREFER_NO_SCHEDULE\", \"value\": \"taint-new-
value-1\", \"key\": \"taint-new-key-1\"}]"
    },
    {
      "type": "TaintsToRemove",
      "value": "[{\"effect\": \"NO_EXECUTE\", \"value\": \"taint-value-1\",
\"key\": \"taint-key-1\"}]"
    }
  ],
  "createdAt": "2024-04-08T12:30:55.486000-04:00",
  "errors": []
}
}

```

For more information, see [Updating a managed node group](#) in the *Amazon EKS User Guide*.

Example 4: Update a managed node group to update scaling-config and update-config for the EKS worker node for an Amazon EKS cluster

The following `update-nodegroup-config` example updates a managed node group to update scaling-config and update-config for the EKS worker node for an Amazon EKS cluster.

```

aws eks update-nodegroup-config \
  --cluster-name my-eks-cluster \
  --nodegroup-name my-eks-nodegroup \
  --scaling-config minSize=1,maxSize=5,desiredSize=2 \
  --update-config maxUnavailable=2

```

Output:

```

{
  "update": {
    "id": "a977160f-59bf-3023-805d-c9826e460aea",
    "status": "InProgress",
    "type": "ConfigUpdate",

```

```
    "params": [
      {
        "type": "MinSize",
        "value": "1"
      },
      {
        "type": "MaxSize",
        "value": "5"
      },
      {
        "type": "DesiredSize",
        "value": "2"
      },
      {
        "type": "MaxUnavailable",
        "value": "2"
      }
    ],
    "createdAt": "2024-04-08T12:35:17.036000-04:00",
    "errors": []
  }
}
```

For more information, see [Updating a managed node group](#) in the *Amazon EKS User Guide*.

- For API details, see [UpdateNodegroupConfig](#) in *AWS CLI Command Reference*.

update-nodegroup-version

The following code example shows how to use `update-nodegroup-version`.

AWS CLI

Example 1: Update the Kubernetes version or AMI version of an Amazon EKS managed node group

The following `update-nodegroup-version` example updates the Kubernetes version or AMI version of an Amazon EKS managed node group to the latest available version for your Kubernetes cluster.

```
aws eks update-nodegroup-version \
  --cluster-name my-eks-cluster \
  --nodegroup-name my-eks-nodegroup \
```



```
--no-force
```

Output:

```
{
  "update": {
    "id": "a94ebfc3-6bf8-307a-89e6-7dbaa36421f7",
    "status": "InProgress",
    "type": "VersionUpdate",
    "params": [
      {
        "type": "Version",
        "value": "1.26"
      },
      {
        "type": "ReleaseVersion",
        "value": "1.26.12-20240329"
      }
    ],
    "createdAt": "2024-04-08T13:16:00.724000-04:00",
    "errors": []
  }
}
```

For more information, see [Updating a managed node group](#) in the *Amazon EKS User Guide*.

Example 2: Update the Kubernetes version or AMI version of an Amazon EKS managed node group

The following `update-nodegroup-version` example updates the Kubernetes version or AMI version of an Amazon EKS managed node group to the specified AMI release version.

```
aws eks update-nodegroup-version \
  --cluster-name my-eks-cluster \
  --nodegroup-name my-eks-nodegroup \
  --kubernetes-version '1.26' \
  --release-version '1.26.12-20240307' \
  --no-force
```

Output:

```
{
```

```
"update": {
  "id": "4db06fe1-088d-336b-bdcd-3fdb94995fb7",
  "status": "InProgress",
  "type": "VersionUpdate",
  "params": [
    {
      "type": "Version",
      "value": "1.26"
    },
    {
      "type": "ReleaseVersion",
      "value": "1.26.12-20240307"
    }
  ],
  "createdAt": "2024-04-08T13:13:58.595000-04:00",
  "errors": []
}
}
```

For more information, see [Updating a managed node group](https://docs.aws.amazon.com/eks/latest/userguide/update-managed-node-group.html) - <https://docs.aws.amazon.com/eks/latest/userguide/update-managed-node-group.html> in the *Amazon EKS User Guide*.

- For API details, see [UpdateNodegroupVersion](#) in *AWS CLI Command Reference*.

Elastic Beanstalk examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Elastic Beanstalk.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

abort-environment-update

The following code example shows how to use `abort-environment-update`.

AWS CLI

To abort a deployment

The following command aborts a running application version deployment for an environment named `my-env`:

```
aws elasticbeanstalk abort-environment-update --environment-name my-env
```

- For API details, see [AbortEnvironmentUpdate](#) in *AWS CLI Command Reference*.

check-dns-availability

The following code example shows how to use `check-dns-availability`.

AWS CLI

To check the availability of a CNAME

The following command checks the availability of the subdomain `my-cname.elasticbeanstalk.com`:

```
aws elasticbeanstalk check-dns-availability --cname-prefix my-cname
```

Output:

```
{
  "Available": true,
  "FullyQualifiedCNAME": "my-cname.elasticbeanstalk.com"
}
```

- For API details, see [CheckDnsAvailability](#) in *AWS CLI Command Reference*.

create-application-version

The following code example shows how to use `create-application-version`.

AWS CLI

To create a new application version

The following command creates a new version, "v1" of an application named "MyApp":

```
aws elasticbeanstalk create-application-version --application-name MyApp
--version-label v1 --description MyAppv1 --source-bundle S3Bucket="my-
bucket",S3Key="sample.war" --auto-create-application
```

The application will be created automatically if it does not already exist, due to the auto-create-application option. The source bundle is a .war file stored in an s3 bucket named "my-bucket" that contains the Apache Tomcat sample application.

Output:

```
{
  "ApplicationVersion": {
    "ApplicationName": "MyApp",
    "VersionLabel": "v1",
    "Description": "MyAppv1",
    "DateCreated": "2015-02-03T23:01:25.412Z",
    "DateUpdated": "2015-02-03T23:01:25.412Z",
    "SourceBundle": {
      "S3Bucket": "my-bucket",
      "S3Key": "sample.war"
    }
  }
}
```

- For API details, see [CreateApplicationVersion](#) in *AWS CLI Command Reference*.

create-application

The following code example shows how to use create-application.

AWS CLI

To create a new application

The following command creates a new application named "MyApp":

```
aws elasticbeanstalk create-application --application-name MyApp --description "my application"
```

The `create-application` command only configures the application's name and description. To upload source code for the application, create an initial version of the application using `create-application-version`. `create-application-version` also has an `auto-create-application` option that lets you create the application and the application version in one step.

Output:

```
{
  "Application": {
    "ApplicationName": "MyApp",
    "ConfigurationTemplates": [],
    "DateUpdated": "2015-02-12T18:32:21.181Z",
    "Description": "my application",
    "DateCreated": "2015-02-12T18:32:21.181Z"
  }
}
```

- For API details, see [CreateApplication](#) in *AWS CLI Command Reference*.

create-configuration-template

The following code example shows how to use `create-configuration-template`.

AWS CLI

To create a configuration template

The following command creates a configuration template named `my-app-v1` from the settings applied to an environment with the id `e-rpqsewtp2j`:

```
aws elasticbeanstalk create-configuration-template --application-name my-app --template-name my-app-v1 --environment-id e-rpqsewtp2j
```

Output:

```
{
```

```
"ApplicationName": "my-app",
"TemplateName": "my-app-v1",
"DateCreated": "2015-08-12T18:40:39Z",
"DateUpdated": "2015-08-12T18:40:39Z",
"SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 8 Java 8"
}
```

- For API details, see [CreateConfigurationTemplate](#) in *AWS CLI Command Reference*.

create-environment

The following code example shows how to use `create-environment`.

AWS CLI

To create a new environment for an application

The following command creates a new environment for version "v1" of a java application named "my-app":

```
aws elasticbeanstalk create-environment --application-name my-app --environment-name
my-env --cname-prefix my-app --version-label v1 --solution-stack-name "64bit Amazon
Linux 2015.03 v2.0.0 running Tomcat 8 Java 8"
```

Output:

```
{
  "ApplicationName": "my-app",
  "EnvironmentName": "my-env",
  "VersionLabel": "v1",
  "Status": "Launching",
  "EnvironmentId": "e-izqpassy4h",
  "SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 8 Java 8",
  "CNAME": "my-app.elasticbeanstalk.com",
  "Health": "Grey",
  "Tier": {
    "Type": "Standard",
    "Name": "WebServer",
    "Version": " "
  },
  "DateUpdated": "2015-02-03T23:04:54.479Z",
```

```
"DateCreated": "2015-02-03T23:04:54.479Z"  
}
```

v1 is the label of an application version previously uploaded with `create-application-version`.

To specify a JSON file to define environment configuration options

The following `create-environment` command specifies that a JSON file with the name `myoptions.json` should be used to override values obtained from the solution stack or the configuration template:

```
aws elasticbeanstalk create-environment --environment-name sample-env --application-  
name sampleapp --option-settings file://myoptions.json
```

`myoptions.json` is a JSON object defining several settings:

```
[  
  {  
    "Namespace": "aws:elb:healthcheck",  
    "OptionName": "Interval",  
    "Value": "15"  
  },  
  {  
    "Namespace": "aws:elb:healthcheck",  
    "OptionName": "Timeout",  
    "Value": "8"  
  },  
  {  
    "Namespace": "aws:elb:healthcheck",  
    "OptionName": "HealthyThreshold",  
    "Value": "2"  
  },  
  {  
    "Namespace": "aws:elb:healthcheck",  
    "OptionName": "UnhealthyThreshold",  
    "Value": "3"  
  }  
]
```

For more information, see *Option Values* in the *AWS Elastic Beanstalk Developer Guide*.

- For API details, see [CreateEnvironment](#) in *AWS CLI Command Reference*.

create-storage-location

The following code example shows how to use `create-storage-location`.

AWS CLI

To create a storage location

The following command creates a storage location in Amazon S3:

```
aws elasticbeanstalk create-storage-location
```

Output:

```
{
  "S3Bucket": "elasticbeanstalk-us-west-2-0123456789012"
}
```

- For API details, see [CreateStorageLocation](#) in *AWS CLI Command Reference*.

delete-application-version

The following code example shows how to use `delete-application-version`.

AWS CLI

To delete an application version

The following command deletes an application version named `22a0-stage-150819_182129` for an application named `my-app`:

```
aws elasticbeanstalk delete-application-version --version-label 22a0-
stage-150819_182129 --application-name my-app
```

- For API details, see [DeleteApplicationVersion](#) in *AWS CLI Command Reference*.

delete-application

The following code example shows how to use `delete-application`.

AWS CLI

To delete an application

The following command deletes an application named `my-app`:

```
aws elasticbeanstalk delete-application --application-name my-app
```

- For API details, see [DeleteApplication](#) in *AWS CLI Command Reference*.

`delete-configuration-template`

The following code example shows how to use `delete-configuration-template`.

AWS CLI

To delete a configuration template

The following command deletes a configuration template named `my-template` for an application named `my-app`:

```
aws elasticbeanstalk delete-configuration-template --template-name my-template --application-name my-app
```

- For API details, see [DeleteConfigurationTemplate](#) in *AWS CLI Command Reference*.

`delete-environment-configuration`

The following code example shows how to use `delete-environment-configuration`.

AWS CLI

To delete a draft configuration

The following command deletes a draft configuration for an environment named `my-env`:

```
aws elasticbeanstalk delete-environment-configuration --environment-name my-env --application-name my-app
```

- For API details, see [DeleteEnvironmentConfiguration](#) in *AWS CLI Command Reference*.

describe-application-versions

The following code example shows how to use `describe-application-versions`.

AWS CLI

To view information about an application version

The following command retrieves information about an application version labeled `v2`:

```
aws elasticbeanstalk describe-application-versions --application-name my-app --
version-label "v2"
```

Output:

```
{
  "ApplicationVersions": [
    {
      "ApplicationName": "my-app",
      "VersionLabel": "v2",
      "Description": "update cover page",
      "DateCreated": "2015-07-23T01:32:26.079Z",
      "DateUpdated": "2015-07-23T01:32:26.079Z",
      "SourceBundle": {
        "S3Bucket": "elasticbeanstalk-us-west-2-015321684451",
        "S3Key": "my-app/5026-stage-150723_224258.war"
      }
    },
    {
      "ApplicationName": "my-app",
      "VersionLabel": "v1",
      "Description": "initial version",
      "DateCreated": "2015-07-23T22:26:10.816Z",
      "DateUpdated": "2015-07-23T22:26:10.816Z",
      "SourceBundle": {
        "S3Bucket": "elasticbeanstalk-us-west-2-015321684451",
        "S3Key": "my-app/5026-stage-150723_222618.war"
      }
    }
  ]
}
```

- For API details, see [DescribeApplicationVersions](#) in *AWS CLI Command Reference*.

describe-applications

The following code example shows how to use describe-applications.

AWS CLI

To view a list of applications

The following command retrieves information about applications in the current region:

```
aws elasticbeanstalk describe-applications
```

Output:

```
{
  "Applications": [
    {
      "ApplicationName": "ruby",
      "ConfigurationTemplates": [],
      "DateUpdated": "2015-08-13T21:05:44.376Z",
      "Versions": [
        "Sample Application"
      ],
      "DateCreated": "2015-08-13T21:05:44.376Z"
    },
    {
      "ApplicationName": "pythonsample",
      "Description": "Application created from the EB CLI using \"eb init\"",
      "Versions": [
        "Sample Application"
      ],
      "DateCreated": "2015-08-13T19:05:43.637Z",
      "ConfigurationTemplates": [],
      "DateUpdated": "2015-08-13T19:05:43.637Z"
    },
    {
      "ApplicationName": "nodejs-example",
      "ConfigurationTemplates": [],
      "DateUpdated": "2015-08-06T17:50:02.486Z",
      "Versions": [
        "add elasticache",
        "First Release"
      ],
      "DateCreated": "2015-08-06T17:50:02.486Z"
    }
  ]
}
```

```

    }
  ]
}

```

- For API details, see [DescribeApplications](#) in *AWS CLI Command Reference*.

describe-configuration-options

The following code example shows how to use `describe-configuration-options`.

AWS CLI

To view configuration options for an environment

The following command retrieves descriptions of all available configuration options for an environment named `my-env`:

```
aws elasticbeanstalk describe-configuration-options --environment-name my-env --
application-name my-app
```

Output (abbreviated):

```
{
  "Options": [
    {
      "Name": "JVMOptions",
      "UserDefined": false,
      "DefaultValue": "Xms=256m,Xmx=256m,XX:MaxPermSize=64m,JVM Options=",
      "ChangeSeverity": "RestartApplicationServer",
      "Namespace": "aws:cloudformation:template:parameter",
      "ValueType": "KeyValueList"
    },
    {
      "Name": "Interval",
      "UserDefined": false,
      "DefaultValue": "30",
      "ChangeSeverity": "NoInterruption",
      "Namespace": "aws:elb:healthcheck",
      "MaxValue": 300,
      "MinValue": 5,
      "ValueType": "Scalar"
    }
  ],
}
```

```

    ...
    {
      "Name": "LowerThreshold",
      "UserDefined": false,
      "DefaultValue": "2000000",
      "ChangeSeverity": "NoInterruption",
      "Namespace": "aws:autoscaling:trigger",
      "MinValue": 0,
      "ValueType": "Scalar"
    },
    {
      "Name": "ListenerEnabled",
      "UserDefined": false,
      "DefaultValue": "true",
      "ChangeSeverity": "Unknown",
      "Namespace": "aws:elb:listener",
      "ValueType": "Boolean"
    }
  ]
}

```

Available configuration options vary per platform and configuration version. For more information about namespaces and supported options, see *Option Values* in the *AWS Elastic Beanstalk Developer Guide*.

- For API details, see [DescribeConfigurationOptions](#) in *AWS CLI Command Reference*.

describe-configuration-settings

The following code example shows how to use `describe-configuration-settings`.

AWS CLI

To view configurations settings for an environment

The following command retrieves configuration settings for an environment named `my-env`:

```
aws elasticbeanstalk describe-configuration-settings --environment-name my-env --
application-name my-app
```

Output (abbreviated):

```
{
```

```

"ConfigurationSettings": [
  {
    "ApplicationName": "my-app",
    "EnvironmentName": "my-env",
    "Description": "Environment created from the EB CLI using \"eb create
\",
    "DeploymentStatus": "deployed",
    "DateCreated": "2015-08-13T19:16:25Z",
    "OptionSettings": [
      {
        "OptionName": "Availability Zones",
        "ResourceName": "AWSEBAutoScalingGroup",
        "Namespace": "aws:autoscaling:asg",
        "Value": "Any"
      },
      {
        "OptionName": "Cooldown",
        "ResourceName": "AWSEBAutoScalingGroup",
        "Namespace": "aws:autoscaling:asg",
        "Value": "360"
      },
      ...
      {
        "OptionName": "ConnectionDrainingTimeout",
        "ResourceName": "AWSEBLoadBalancer",
        "Namespace": "aws:elb:policies",
        "Value": "20"
      },
      {
        "OptionName": "ConnectionSettingIdleTimeout",
        "ResourceName": "AWSEBLoadBalancer",
        "Namespace": "aws:elb:policies",
        "Value": "60"
      }
    ],
    "DateUpdated": "2015-08-13T23:30:07Z",
    "SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 8
Java 8"
  }
]
}

```

For more information about namespaces and supported options, see [Option Values](#) in the *AWS Elastic Beanstalk Developer Guide*.

- For API details, see [DescribeConfigurationSettings](#) in *AWS CLI Command Reference*.

describe-environment-health

The following code example shows how to use `describe-environment-health`.

AWS CLI

To view environment health

The following command retrieves overall health information for an environment named `my-env`:

```
aws elasticbeanstalk describe-environment-health --environment-name my-env --attribute-names All
```

Output:

```
{
  "Status": "Ready",
  "EnvironmentName": "my-env",
  "Color": "Green",
  "ApplicationMetrics": {
    "Duration": 10,
    "Latency": {
      "P99": 0.004,
      "P75": 0.002,
      "P90": 0.003,
      "P95": 0.004,
      "P85": 0.003,
      "P10": 0.001,
      "P999": 0.004,
      "P50": 0.001
    },
    "RequestCount": 45,
    "StatusCodes": {
      "Status3xx": 0,
      "Status2xx": 45,
      "Status5xx": 0,
      "Status4xx": 0
    }
  }
}
```

```
    }
  },
  "RefreshedAt": "2015-08-20T21:09:18Z",
  "HealthStatus": "Ok",
  "InstancesHealth": {
    "Info": 0,
    "Ok": 1,
    "Unknown": 0,
    "Severe": 0,
    "Warning": 0,
    "Degraded": 0,
    "NoData": 0,
    "Pending": 0
  },
  "Causes": []
}
```

Health information is only available for environments with enhanced health reporting enabled. For more information, see [Enhanced Health Reporting and Monitoring](#) in the *AWS Elastic Beanstalk Developer Guide*.

- For API details, see [DescribeEnvironmentHealth](#) in *AWS CLI Command Reference*.

describe-environment-resources

The following code example shows how to use `describe-environment-resources`.

AWS CLI

To view information about the AWS resources in your environment

The following command retrieves information about resources in an environment named `my-env`:

```
aws elasticbeanstalk describe-environment-resources --environment-name my-env
```

Output:

```
{
  "EnvironmentResources": {
    "EnvironmentName": "my-env",
    "AutoScalingGroups": [
```



```

        {
            "Name": "awseb-e-qu3fyyjyjs-stack-AWSEBAutoScalingGroup-
QSB2Z088SXZT"
        }
    ],
    "Triggers": [],
    "LoadBalancers": [
        {
            "Name": "awseb-e-q-AWSEBLoa-1EEPZ0K98BIF0"
        }
    ],
    "Queues": [],
    "Instances": [
        {
            "Id": "i-0c91c786"
        }
    ],
    "LaunchConfigurations": [
        {
            "Name": "awseb-e-qu3fyyjyjs-stack-
AWSEBAutoScalingLaunchConfiguration-1UUVQIBC96TQ2"
        }
    ]
}
}
}

```

- For API details, see [DescribeEnvironmentResources](#) in *AWS CLI Command Reference*.

describe-environments

The following code example shows how to use `describe-environments`.

AWS CLI

To view information about an environment

The following command retrieves information about an environment named `my-env`:

```
aws elasticbeanstalk describe-environments --environment-names my-env
```

Output:

```
{
```

```
"Environments": [  
  {  
    "ApplicationName": "my-app",  
    "EnvironmentName": "my-env",  
    "VersionLabel": "7f58-stage-150812_025409",  
    "Status": "Ready",  
    "EnvironmentId": "e-rpqsewtp2j",  
    "EndpointURL": "awseb-e-w-AWSEBLoa-1483140XB0Q4L-109QXY8121.us-  
west-2.elb.amazonaws.com",  
    "SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 8  
Java 8",  
    "CNAME": "my-env.elasticbeanstalk.com",  
    "Health": "Green",  
    "AbortableOperationInProgress": false,  
    "Tier": {  
      "Version": " ",  
      "Type": "Standard",  
      "Name": "WebServer"  
    },  
    "DateUpdated": "2015-08-12T18:16:55.019Z",  
    "DateCreated": "2015-08-07T20:48:49.599Z"  
  }  
]
```

- For API details, see [DescribeEnvironments](#) in *AWS CLI Command Reference*.

describe-events

The following code example shows how to use `describe-events`.

AWS CLI

To view events for an environment

The following command retrieves events for an environment named `my-env`:

```
aws elasticbeanstalk describe-events --environment-name my-env
```

Output (abbreviated):

```
{
```

```

"Events": [
  {
    "ApplicationName": "my-app",
    "EnvironmentName": "my-env",
    "Message": "Environment health has transitioned from Info to Ok.",
    "EventDate": "2015-08-20T07:06:53.535Z",
    "Severity": "INFO"
  },
  {
    "ApplicationName": "my-app",
    "EnvironmentName": "my-env",
    "Severity": "INFO",
    "RequestId": "b7f3960b-4709-11e5-ba1e-07e16200da41",
    "Message": "Environment update completed successfully.",
    "EventDate": "2015-08-20T07:06:02.049Z"
  },
  ...
  {
    "ApplicationName": "my-app",
    "EnvironmentName": "my-env",
    "Severity": "INFO",
    "RequestId": "ca8dfbf6-41ef-11e5-988b-651aa638f46b",
    "Message": "Using elasticbeanstalk-us-west-2-012445113685 as Amazon S3
storage bucket for environment data.",
    "EventDate": "2015-08-13T19:16:27.561Z"
  },
  {
    "ApplicationName": "my-app",
    "EnvironmentName": "my-env",
    "Severity": "INFO",
    "RequestId": "cdfba8f6-41ef-11e5-988b-65638f41aa6b",
    "Message": "createEnvironment is starting.",
    "EventDate": "2015-08-13T19:16:26.581Z"
  }
]
}

```

- For API details, see [DescribeEvents](#) in *AWS CLI Command Reference*.

describe-instances-health

The following code example shows how to use `describe-instances-health`.

AWS CLI

To view environment health

The following command retrieves health information for instances in an environment named `my-env`:

```
aws elasticbeanstalk describe-instances-health --environment-name my-env --
attribute-names All
```

Output:

```
{
  "InstanceHealthList": [
    {
      "InstanceId": "i-08691cc7",
      "ApplicationMetrics": {
        "Duration": 10,
        "Latency": {
          "P99": 0.006,
          "P75": 0.002,
          "P90": 0.004,
          "P95": 0.005,
          "P85": 0.003,
          "P10": 0.0,
          "P999": 0.006,
          "P50": 0.001
        },
        "RequestCount": 48,
        "StatusCodes": {
          "Status3xx": 0,
          "Status2xx": 47,
          "Status5xx": 0,
          "Status4xx": 1
        }
      },
      "System": {
        "LoadAverage": [
          0.0,
          0.02,
          0.05
        ],
        "CPUUtilization": {
```

```
        "SoftIRQ": 0.1,  
        "IOWait": 0.2,  
        "System": 0.3,  
        "Idle": 97.8,  
        "User": 1.5,  
        "IRQ": 0.0,  
        "Nice": 0.1  
    }  
},  
"Color": "Green",  
"HealthStatus": "Ok",  
"LaunchedAt": "2015-08-13T19:17:09Z",  
"Causes": []  
}  
],  
"RefreshedAt": "2015-08-20T21:09:08Z"  
}
```

Health information is only available for environments with enhanced health reporting enabled. For more information, see [Enhanced Health Reporting and Monitoring](#) in the *AWS Elastic Beanstalk Developer Guide*.

- For API details, see [DescribeInstancesHealth](#) in *AWS CLI Command Reference*.

list-available-solution-stacks

The following code example shows how to use `list-available-solution-stacks`.

AWS CLI

To view solution stacks

The following command lists solution stacks for all currently available platform configurations and any that you have used in the past:

```
aws elasticbeanstalk list-available-solution-stacks
```

Output (abbreviated):

```
{  
  "SolutionStacks": [  
    {
```

```

    "64bit Amazon Linux 2015.03 v2.0.0 running Node.js",
    "64bit Amazon Linux 2015.03 v2.0.0 running PHP 5.6",
    "64bit Amazon Linux 2015.03 v2.0.0 running PHP 5.5",
    "64bit Amazon Linux 2015.03 v2.0.0 running PHP 5.4",
    "64bit Amazon Linux 2015.03 v2.0.0 running Python 3.4",
    "64bit Amazon Linux 2015.03 v2.0.0 running Python 2.7",
    "64bit Amazon Linux 2015.03 v2.0.0 running Python",
    "64bit Amazon Linux 2015.03 v2.0.0 running Ruby 2.2 (Puma)",
    "64bit Amazon Linux 2015.03 v2.0.0 running Ruby 2.2 (Passenger Standalone)",
    "64bit Amazon Linux 2015.03 v2.0.0 running Ruby 2.1 (Puma)",
    "64bit Amazon Linux 2015.03 v2.0.0 running Ruby 2.1 (Passenger Standalone)",
    "64bit Amazon Linux 2015.03 v2.0.0 running Ruby 2.0 (Puma)",
    "64bit Amazon Linux 2015.03 v2.0.0 running Ruby 2.0 (Passenger Standalone)",
    "64bit Amazon Linux 2015.03 v2.0.0 running Ruby 1.9.3",
    "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 8 Java 8",
    "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 7 Java 7",
    "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 7 Java 6",
    "64bit Windows Server Core 2012 R2 running IIS 8.5",
    "64bit Windows Server 2012 R2 running IIS 8.5",
    "64bit Windows Server 2012 running IIS 8",
    "64bit Windows Server 2008 R2 running IIS 7.5",
    "64bit Amazon Linux 2015.03 v2.0.0 running Docker 1.6.2",
    "64bit Amazon Linux 2015.03 v2.0.0 running Multi-container Docker 1.6.2
(Generic)",
    "64bit Debian jessie v2.0.0 running GlassFish 4.1 Java 8 (Preconfigured -
Docker)",
    "64bit Debian jessie v2.0.0 running GlassFish 4.0 Java 7 (Preconfigured -
Docker)",
    "64bit Debian jessie v2.0.0 running Go 1.4 (Preconfigured - Docker)",
    "64bit Debian jessie v2.0.0 running Go 1.3 (Preconfigured - Docker)",
    "64bit Debian jessie v2.0.0 running Python 3.4 (Preconfigured - Docker)",
  ],
  "SolutionStackDetails": [
    {
      "PermittedFileTypes": [
        "zip"
      ],
      "SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.0 running Node.js"
    },
    ...
  ]
}

```

- For API details, see [ListAvailableSolutionStacks](#) in *AWS CLI Command Reference*.

rebuild-environment

The following code example shows how to use `rebuild-environment`.

AWS CLI

To rebuild an environment

The following command terminates and recreates the resources in an environment named `my-env`:

```
aws elasticbeanstalk rebuild-environment --environment-name my-env
```

- For API details, see [RebuildEnvironment](#) in *AWS CLI Command Reference*.

request-environment-info

The following code example shows how to use `request-environment-info`.

AWS CLI

To request tailed logs

The following command requests logs from an environment named `my-env`:

```
aws elasticbeanstalk request-environment-info --environment-name my-env --info-type tail
```

After requesting logs, retrieve their location with `retrieve-environment-info`.

- For API details, see [RequestEnvironmentInfo](#) in *AWS CLI Command Reference*.

restart-app-server

The following code example shows how to use `restart-app-server`.

AWS CLI

To restart application servers

The following command restarts application servers on all instances in an environment named `my-env`:

```
aws elasticbeanstalk restart-app-server --environment-name my-env
```

- For API details, see [RestartAppServer](#) in *AWS CLI Command Reference*.

retrieve-environment-info

The following code example shows how to use `retrieve-environment-info`.

AWS CLI

To retrieve tailed logs

The following command retrieves a link to logs from an environment named `my-env`:

```
aws elasticbeanstalk retrieve-environment-info --environment-name my-env --info-type tail
```

Output:

```
{
  "EnvironmentInfo": [
    {
      "SampleTimestamp": "2015-08-20T22:23:17.703Z",
      "Message": "https://elasticbeanstalk-us-west-2-0123456789012.s3.amazonaws.com/resources/environments/logs/tail/e-fyqyju3yjs/i-09c1c867/TailLogs-1440109397703.out?AWSAccessKeyId=AKGPT4J56IAJ2EUBL5CQ&Expires=1440195891&Signature=n%2BEa10V6A2HI0x4Rcfb7LT16bBM%3D",
      "InfoType": "tail",
      "Ec2InstanceId": "i-09c1c867"
    }
  ]
}
```

View the link in a browser. Prior to retrieval, logs must be requested with `request-environment-info`.

- For API details, see [RetrieveEnvironmentInfo](#) in *AWS CLI Command Reference*.

swap-environment-cnames

The following code example shows how to use `swap-environment-cnames`.

AWS CLI

To swap environment CNAMEs

The following command swaps the assigned subdomains of two environments:

```
aws elasticbeanstalk swap-environment-cnames --source-environment-name my-env-blue
--destination-environment-name my-env-green
```

- For API details, see [SwapEnvironmentCnames](#) in *AWS CLI Command Reference*.

terminate-environment

The following code example shows how to use `terminate-environment`.

AWS CLI

To terminate an environment

The following command terminates an Elastic Beanstalk environment named `my-env`:

```
aws elasticbeanstalk terminate-environment --environment-name my-env
```

Output:

```
{
  "ApplicationName": "my-app",
  "EnvironmentName": "my-env",
  "Status": "Terminating",
  "EnvironmentId": "e-fh2eravpns",
  "EndpointURL": "awseb-e-f-AWSEBLoa-1I9XUMP4-8492WNUP202574.us-
west-2.elb.amazonaws.com",
  "SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 8 Java
8",
  "CNAME": "my-env.elasticbeanstalk.com",
  "Health": "Grey",
  "AbortableOperationInProgress": false,
  "Tier": {
```

```
    "Version": " ",
    "Type": "Standard",
    "Name": "WebServer"
  },
  "DateUpdated": "2015-08-12T19:05:54.744Z",
  "DateCreated": "2015-08-12T18:52:53.622Z"
}
```

- For API details, see [TerminateEnvironment](#) in *AWS CLI Command Reference*.

update-application-version

The following code example shows how to use `update-application-version`.

AWS CLI

To change an application version's description

The following command updates the description of an application version named `22a0-stage-150819_185942`:

```
aws elasticbeanstalk update-application-version --version-label 22a0-
stage-150819_185942 --application-name my-app --description "new description"
```

Output:

```
{
  "ApplicationVersion": {
    "ApplicationName": "my-app",
    "VersionLabel": "22a0-stage-150819_185942",
    "Description": "new description",
    "DateCreated": "2015-08-19T18:59:17.646Z",
    "DateUpdated": "2015-08-20T22:53:28.871Z",
    "SourceBundle": {
      "S3Bucket": "elasticbeanstalk-us-west-2-0123456789012",
      "S3Key": "my-app/22a0-stage-150819_185942.war"
    }
  }
}
```

- For API details, see [UpdateApplicationVersion](#) in *AWS CLI Command Reference*.

update-application

The following code example shows how to use update-application.

AWS CLI

To change an application's description

The following command updates the description of an application named my-app:

```
aws elasticbeanstalk update-application --application-name my-app --description "my Elastic Beanstalk application"
```

Output:

```
{
  "Application": {
    "ApplicationName": "my-app",
    "Description": "my Elastic Beanstalk application",
    "Versions": [
      "2fba-stage-150819_234450",
      "bf07-stage-150820_214945",
      "93f8",
      "fd7c-stage-150820_000431",
      "22a0-stage-150819_185942"
    ],
    "DateCreated": "2015-08-13T19:15:50.449Z",
    "ConfigurationTemplates": [],
    "DateUpdated": "2015-08-20T22:34:56.195Z"
  }
}
```

- For API details, see [UpdateApplication](#) in *AWS CLI Command Reference*.

update-configuration-template

The following code example shows how to use update-configuration-template.

AWS CLI

To update a configuration template

The following command removes the configured CloudWatch custom health metrics configuration `ConfigDocument` from a saved configuration template named `my-template`:

```
aws elasticbeanstalk update-configuration-template --template-name my-template --application-name my-app --options-to-remove Namespace=aws:elasticbeanstalk:healthreporting:system,OptionName=ConfigDocument
```

Output:

```
{
  "ApplicationName": "my-app",
  "TemplateName": "my-template",
  "DateCreated": "2015-08-20T22:39:31Z",
  "DateUpdated": "2015-08-20T22:43:11Z",
  "SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 8 Java 8"
}
```

For more information about namespaces and supported options, see [Option Values](#) in the *AWS Elastic Beanstalk Developer Guide*.

- For API details, see [UpdateConfigurationTemplate](#) in *AWS CLI Command Reference*.

update-environment

The following code example shows how to use `update-environment`.

AWS CLI

To update an environment to a new version

The following command updates an environment named `"my-env"` to version `"v2"` of the application to which it belongs:

```
aws elasticbeanstalk update-environment --environment-name my-env --version-label v2
```

This command requires that the `"my-env"` environment already exists and belongs to an application that has a valid application version with the label `"v2"`.

Output:

```
{
```

```
"ApplicationName": "my-app",
"EnvironmentName": "my-env",
"VersionLabel": "v2",
"Status": "Updating",
"EnvironmentId": "e-szqipays4h",
"EndpointURL": "awseb-e-i-AWSEBLoa-1RD LX6TC9VUA0-0123456789.us-
west-2.elb.amazonaws.com",
"SolutionStackName": "64bit Amazon Linux running Tomcat 7",
"CNAME": "my-env.elasticbeanstalk.com",
"Health": "Grey",
"Tier": {
  "Version": " ",
  "Type": "Standard",
  "Name": "WebServer"
},
"DateUpdated": "2015-02-03T23:12:29.119Z",
"DateCreated": "2015-02-03T23:04:54.453Z"
}
```

To set an environment variable

The following command sets the value of the "PARAM1" variable in the "my-env" environment to "ParamValue":

```
aws elasticbeanstalk update-environment --environment-name my-env --option-settings
Namespace=aws:elasticbeanstalk:application:environment,OptionName=PARAM1,Value=ParamValue
```

The `option-settings` parameter takes a namespace in addition to the name and value of the variable. Elastic Beanstalk supports several namespaces for options in addition to environment variables.

To configure option settings from a file

The following command configures several options in the `aws:elb:loadbalancer` namespace from a file:

```
aws elasticbeanstalk update-environment --environment-name my-env --option-settings
file://options.json
```

`options.json` is a JSON object defining several settings:

```
[
```

```

{
  "Namespace": "aws:elb:healthcheck",
  "OptionName": "Interval",
  "Value": "15"
},
{
  "Namespace": "aws:elb:healthcheck",
  "OptionName": "Timeout",
  "Value": "8"
},
{
  "Namespace": "aws:elb:healthcheck",
  "OptionName": "HealthyThreshold",
  "Value": "2"
},
{
  "Namespace": "aws:elb:healthcheck",
  "OptionName": "UnhealthyThreshold",
  "Value": "3"
}
]

```

Output:

```

{
  "ApplicationName": "my-app",
  "EnvironmentName": "my-env",
  "VersionLabel": "7f58-stage-150812_025409",
  "Status": "Updating",
  "EnvironmentId": "e-wtp2rqpsej",
  "EndpointURL": "awseb-e-w-AWSEBLoa-14XB83101Q4L-104QXY80921.sa-
east-1.elb.amazonaws.com",
  "SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.0 running Tomcat 8 Java
8",
  "CNAME": "my-env.elasticbeanstalk.com",
  "Health": "Grey",
  "AbortableOperationInProgress": true,
  "Tier": {
    "Version": " ",
    "Type": "Standard",
    "Name": "WebServer"
  },
  "DateUpdated": "2015-08-12T18:15:23.804Z",

```

```
"DateCreated": "2015-08-07T20:48:49.599Z"
}
```

For more information about namespaces and supported options, see [Option Values](#) in the *AWS Elastic Beanstalk Developer Guide*.

- For API details, see [UpdateEnvironment](#) in *AWS CLI Command Reference*.

validate-configuration-settings

The following code example shows how to use `validate-configuration-settings`.

AWS CLI

To validate configuration settings

The following command validates a CloudWatch custom metrics config document:

```
aws elasticbeanstalk validate-configuration-settings --application-name my-app --
environment-name my-env --option-settings file://options.json
```

`options.json` is a JSON document that includes one or more configuration settings to `validate`:

```
[
  {
    "Namespace": "aws:elasticbeanstalk:healthreporting:system",
    "OptionName": "ConfigDocument",
    "Value": "{\"CloudWatchMetrics\": {\"Environment\":
{\\\"ApplicationLatencyP99.9\\\": null,\\\"InstancesSevere\\\": 60,
\\\"ApplicationLatencyP90\\\": 60,\\\"ApplicationLatencyP99\\\": null,
\\\"ApplicationLatencyP95\\\": 60,\\\"InstancesUnknown\\\": 60,\\\"ApplicationLatencyP85\\\":
60,\\\"InstancesInfo\\\": null,\\\"ApplicationRequests2xx\\\": null,\\\"InstancesDegraded
\\\": null,\\\"InstancesWarning\\\": 60,\\\"ApplicationLatencyP50\\\": 60,
\\\"ApplicationRequestsTotal\\\": null,\\\"InstancesNoData\\\": null,\\\"InstancesPending
\\\": 60,\\\"ApplicationLatencyP10\\\": null,\\\"ApplicationRequests5xx\\\": null,
\\\"ApplicationLatencyP75\\\": null,\\\"Instances0k\\\": 60,\\\"ApplicationRequests3xx\\\":
null,\\\"ApplicationRequests4xx\\\": null},\\\"Instance\\\": {\\\"ApplicationLatencyP99.9\\\":
null,\\\"ApplicationLatencyP90\\\": 60,\\\"ApplicationLatencyP99\\\": null,
\\\"ApplicationLatencyP95\\\": null,\\\"ApplicationLatencyP85\\\": null,\\\"CPUUser\\\": 60,
\\\"ApplicationRequests2xx\\\": null,\\\"CPUIdle\\\": null,\\\"ApplicationLatencyP50\\\":
```

```

    null,\"ApplicationRequestsTotal\": 60,\"RootFilesystemUtil\": null,
    \"LoadAverage1min\": null,\"CPUirq\": null,\"CPUNice\": 60,\"CPUiowait\": 60,
    \"ApplicationLatencyP10\": null,\"LoadAverage5min\": null,\"ApplicationRequests5xx
    \": null,\"ApplicationLatencyP75\": 60,\"CPUSystem\": 60,\"ApplicationRequests3xx\":
    60,\"ApplicationRequests4xx\": null,\"InstanceHealth\": null,\"CPUSoftirq\": 60}},
    \"Version\": 1}
  }
]

```

If the options that you specify are valid for the specified environment, Elastic Beanstalk returns an empty Messages array:

```

{
  "Messages": []
}

```

If validation fails, the response will include information about the error:

```

{
  "Messages": [
    {
      "OptionName": "ConfigDocumet",
      "Message": "Invalid option specification (Namespace:
      'aws:elasticbeanstalk:healthreporting:system', OptionName: 'ConfigDocumet'):
      Unknown configuration setting.",
      "Namespace": "aws:elasticbeanstalk:healthreporting:system",
      "Severity": "error"
    }
  ]
}

```

For more information about namespaces and supported options, see Option Values in the *AWS Elastic Beanstalk Developer Guide*.

- For API details, see [ValidateConfigurationSettings](#) in *AWS CLI Command Reference*.

Elastic Load Balancing - Version 1 examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Elastic Load Balancing - Version 1.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-tags

The following code example shows how to use add-tags.

AWS CLI

To add a tag to a load balancer

This example adds tags to the specified load balancer.

Command:

```
aws elb add-tags --load-balancer-name my-load-balancer --tags
"Key=project,Value=lima" "Key=department,Value=digital-media"
```

- For API details, see [AddTags](#) in *AWS CLI Command Reference*.

apply-security-groups-to-load-balancer

The following code example shows how to use apply-security-groups-to-load-balancer.

AWS CLI

To associate a security group with a load balancer in a VPC

This example associates a security group with the specified load balancer in a VPC.

Command:

```
aws elb apply-security-groups-to-load-balancer --load-balancer-name my-load-balancer
--security-groups sg-fc448899
```

Output:

```
{
  "SecurityGroups": [
    "sg-fc448899"
  ]
}
```

- For API details, see [ApplySecurityGroupsToLoadBalancer](#) in *AWS CLI Command Reference*.

attach-load-balancer-to-subnets

The following code example shows how to use `attach-load-balancer-to-subnets`.

AWS CLI

To attach subnets to a load balancer

This example adds the specified subnet to the set of configured subnets for the specified load balancer.

Command:

```
aws elb attach-load-balancer-to-subnets --load-balancer-name my-load-balancer --
subnets subnet-0ecac448
```

Output:

```
{
  "Subnets": [
    "subnet-15aaab61",
    "subnet-0ecac448"
  ]
}
```

```
}
```

- For API details, see [AttachLoadBalancerToSubnets](#) in *AWS CLI Command Reference*.

configure-health-check

The following code example shows how to use `configure-health-check`.

AWS CLI

To specify the health check settings for your backend EC2 instances

This example specifies the health check settings used to evaluate the health of your backend EC2 instances.

Command:

```
aws elb configure-health-check --load-balancer-name my-load-balancer --health-check
  Target=HTTP:80/png,Interval=30,UnhealthyThreshold=2,HealthyThreshold=2,Timeout=3
```

Output:

```
{
  "HealthCheck": {
    "HealthyThreshold": 2,
    "Interval": 30,
    "Target": "HTTP:80/png",
    "Timeout": 3,
    "UnhealthyThreshold": 2
  }
}
```

- For API details, see [ConfigureHealthCheck](#) in *AWS CLI Command Reference*.

create-app-cookie-stickiness-policy

The following code example shows how to use `create-app-cookie-stickiness-policy`.

AWS CLI

To generate a stickiness policy for your HTTPS load balancer

This example generates a stickiness policy that follows the sticky session lifetimes of the application-generated cookie.

Command:

```
aws elb create-app-cookie-stickiness-policy --load-balancer-name my-load-balancer --policy-name my-app-cookie-policy --cookie-name my-app-cookie
```

- For API details, see [CreateAppCookieStickinessPolicy](#) in *AWS CLI Command Reference*.

create-lb-cookie-stickiness-policy

The following code example shows how to use `create-lb-cookie-stickiness-policy`.

AWS CLI

To generate a duration-based stickiness policy for your HTTPS load balancer

This example generates a stickiness policy with sticky session lifetimes controlled by the specified expiration period.

Command:

```
aws elb create-lb-cookie-stickiness-policy --load-balancer-name my-load-balancer --policy-name my-duration-cookie-policy --cookie-expiration-period 60
```

- For API details, see [CreateLbCookieStickinessPolicy](#) in *AWS CLI Command Reference*.

create-load-balancer-listeners

The following code example shows how to use `create-load-balancer-listeners`.

AWS CLI

To create HTTP listeners for a load balancer

This example creates a listener for your load balancer at port 80 using the HTTP protocol.

Command:

```
aws elb create-load-balancer-listeners --load-balancer-name my-load-balancer --  
listeners "Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80"
```

To create HTTPS listeners for a load balancer

This example creates a listener for your load balancer at port 443 using the HTTPS protocol.

Command:

```
aws elb create-load-balancer-listeners --load-balancer-name my-load-balancer --  
listeners  
"Protocol=HTTPS,LoadBalancerPort=443,InstanceProtocol=HTTP,InstancePort=80"
```

- For API details, see [CreateLoadBalancerListeners](#) in *AWS CLI Command Reference*.

create-load-balancer-policy

The following code example shows how to use `create-load-balancer-policy`.

AWS CLI

To create a policy that enables Proxy Protocol on a load balancer

This example creates a policy that enables Proxy Protocol on the specified load balancer.

Command:

```
aws elb create-load-balancer-policy --load-balancer-name my-load-balancer --policy-  
name my-ProxyProtocol-policy --policy-type-name ProxyProtocolPolicyType --policy-  
attributes AttributeName=ProxyProtocol,AttributeValue=true
```

To create an SSL negotiation policy using the recommended security policy

This example creates an SSL negotiation policy for the specified HTTPS load balancer using the recommended security policy.

Command:

```
aws elb create-load-balancer-policy --load-balancer-name my-load-  
balancer --policy-name my-SSLNegotiation-policy --policy-type-name
```

```
SSLNegotiationPolicyType --policy-attributes AttributeName=Reference-Security-
Policy,AttributeValue=ELBSecurityPolicy-2015-03
```

To create an SSL negotiation policy using a custom security policy

This example creates an SSL negotiation policy for your HTTPS load balancer using a custom security policy by enabling the protocols and the ciphers.

Command:

```
aws elb create-load-balancer-policy --load-balancer-name my-load-balancer --policy-
name my-SSLNegotiation-policy --policy-type-name SSLNegotiationPolicyType --policy-
attributes AttributeName=Protocol-SSLv3,AttributeValue=true AttributeName=Protocol-
TLSv1.1,AttributeValue=true AttributeName=DHE-RSA-AES256-SHA256,AttributeValue=true
AttributeName=Server-Defined-Cipher-Order,AttributeValue=true
```

To create a public key policy

This example creates a public key policy.

Command:

```
aws elb create-load-balancer-policy --load-balancer-name my-load-balancer --policy-
name my-PublicKey-policy --policy-type-name PublicKeyPolicyType --policy-attributes
AttributeName=PublicKey,AttributeValue=MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAwAYUjnfY
+dS74kj//c6x7R0tusUaeQCTgIUkayttRDWchuqo1pHC1u
+n5xxXnBBE2ejbb2WRsKIQ5rXEeixsjFpFsojpSQKkzhVGI6mJVZBJDVKSHmswnwLBdofLhzvllpovBPTHe
+o4haAWvDBALJU0pkSI1FecPHcs2hwx14zHoXy1e2k36A64nXW43wtfx5qcVSIxtCE0jnYRg7RPvybaGfQ
+v6Iaxb/+7J5kEvZhTFQId+bSiJImF1FSUT1W1xwzBZPUbcUkkXDj45vC2s3Z8E
+Lk7a3uZhvsQHLZnrFuWjBWGWvZ/MhZYgEXAMPLE
```

To create a backend server authentication policy

This example creates a backend server authentication policy that enables authentication on your backend instance using a public key policy.

Command:

```
aws elb create-load-balancer-policy --load-balancer-name my-load-
balancer --policy-name my-authentication-policy --policy-type-
```

```
name BackendServerAuthenticationPolicyType --policy-attributes
Attribute=PublicKeyPolicyName,AttributeValue=my-PublicKey-policy
```

- For API details, see [CreateLoadBalancerPolicy](#) in *AWS CLI Command Reference*.

create-load-balancer

The following code example shows how to use `create-load-balancer`.

AWS CLI

To create an HTTP load balancer

This example creates a load balancer with an HTTP listener in a VPC.

Command:

```
aws elb create-load-balancer --load-balancer-name my-load-balancer --listeners
"Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80" --subnets
subnet-15aaab61 --security-groups sg-a61988c3
```

Output:

```
{
  "DNSName": "my-load-balancer-1234567890.us-west-2.elb.amazonaws.com"
}
```

This example creates a load balancer with an HTTP listener in EC2-Classic.

Command:

```
aws elb create-load-balancer --load-balancer-name my-load-balancer --listeners
"Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80" --
availability-zones us-west-2a us-west-2b
```

Output:

```
{
  "DNSName": "my-load-balancer-123456789.us-west-2.elb.amazonaws.com"
}
```

```
}
```

To create an HTTPS load balancer

This example creates a load balancer with an HTTPS listener in a VPC.

Command:

```
aws elb create-load-balancer --load-balancer-name my-load-balancer --listeners  
"Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80"  
"Protocol=HTTPS,LoadBalancerPort=443,InstanceProtocol=HTTP,InstancePort=80,SSLCertificateId  
certificate/my-server-cert" --subnets subnet-15aaab61 --security-groups sg-a61988c3
```

Output:

```
{  
  "DNSName": "my-load-balancer-1234567890.us-west-2.elb.amazonaws.com"  
}
```

This example creates a load balancer with an HTTPS listener in EC2-Classic.

Command:

```
aws elb create-load-balancer --load-balancer-name my-load-balancer --listeners  
"Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80"  
"Protocol=HTTPS,LoadBalancerPort=443,InstanceProtocol=HTTP,InstancePort=80,SSLCertificateId  
certificate/my-server-cert" --availability-zones us-west-2a us-west-2b
```

Output:

```
{  
  "DNSName": "my-load-balancer-123456789.us-west-2.elb.amazonaws.com"  
}
```

To create an internal load balancer

This example creates an internal load balancer with an HTTP listener in a VPC.

Command:


```
aws elb create-load-balancer --load-balancer-name my-load-balancer --listeners
"Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80" --scheme
internal --subnets subnet-a85db0df --security-groups sg-a61988c3
```

Output:

```
{
  "DNSName": "internal-my-load-balancer-123456789.us-west-2.elb.amazonaws.com"
}
```

- For API details, see [CreateLoadBalancer](#) in *AWS CLI Command Reference*.

delete-load-balancer-listeners

The following code example shows how to use `delete-load-balancer-listeners`.

AWS CLI

To delete a listener from your load balancer

This example deletes the listener for the specified port from the specified load balancer.

Command:

```
aws elb delete-load-balancer-listeners --load-balancer-name my-load-balancer --load-
balancer-ports 80
```

- For API details, see [DeleteLoadBalancerListeners](#) in *AWS CLI Command Reference*.

delete-load-balancer-policy

The following code example shows how to use `delete-load-balancer-policy`.

AWS CLI

To delete a policy from your load balancer

This example deletes the specified policy from the specified load balancer. The policy must not be enabled on any listener.

Command:

```
aws elb delete-load-balancer-policy --load-balancer-name my-load-balancer --policy-name my-duration-cookie-policy
```

- For API details, see [DeleteLoadBalancerPolicy](#) in *AWS CLI Command Reference*.

delete-load-balancer

The following code example shows how to use `delete-load-balancer`.

AWS CLI**To delete a load balancer**

This example deletes the specified load balancer.

Command:

```
aws elb delete-load-balancer --load-balancer-name my-load-balancer
```

- For API details, see [DeleteLoadBalancer](#) in *AWS CLI Command Reference*.

deregister-instances-from-load-balancer

The following code example shows how to use `deregister-instances-from-load-balancer`.

AWS CLI**To deregister instances from a load balancer**

This example deregisters the specified instance from the specified load balancer.

Command:

```
aws elb deregister-instances-from-load-balancer --load-balancer-name my-load-balancer --instances i-d6f6fae3
```

Output:

```
{
  "Instances": [
    {
      "InstanceId": "i-207d9717"
    },
    {
      "InstanceId": "i-afefb49b"
    }
  ]
}
```

- For API details, see [DeregisterInstancesFromLoadBalancer](#) in *AWS CLI Command Reference*.

describe-account-limits

The following code example shows how to use `describe-account-limits`.

AWS CLI

To describe your Classic Load Balancer limits

The following `describe-account-limits` example displays details about the Classic Load Balancer limits for your AWS account.

```
aws elb describe-account-limits
```

Output:

```
{
  "Limits": [
    {
      "Name": "classic-load-balancers",
      "Max": "20"
    },
    {
      "Name": "classic-listeners",
      "Max": "100"
    },
    {
      "Name": "classic-registered-instances",
      "Max": "1000"
    }
  ]
}
```

```
    }  
  ]  
}
```

- For API details, see [DescribeAccountLimits](#) in *AWS CLI Command Reference*.

describe-instance-health

The following code example shows how to use `describe-instance-health`.

AWS CLI

To describe the health of the instances for a load balancer

This example describes the health of the instances for the specified load balancer.

Command:

```
aws elb describe-instance-health --load-balancer-name my-load-balancer
```

Output:

```
{  
  "InstanceStates": [  
    {  
      "InstanceId": "i-207d9717",  
      "ReasonCode": "N/A",  
      "State": "InService",  
      "Description": "N/A"  
    },  
    {  
      "InstanceId": "i-afefb49b",  
      "ReasonCode": "N/A",  
      "State": "InService",  
      "Description": "N/A"  
    }  
  ]  
}
```

To describe the health of an instance for a load balancer

This example describes the health of the specified instance for the specified load balancer.

Command:

```
aws elb describe-instance-health --load-balancer-name my-load-balancer --instances
i-7299c809
```

The following is an example response for an instance that is registering.

Output:

```
{
  "InstanceStates": [
    {
      "InstanceId": "i-7299c809",
      "ReasonCode": "ELB",
      "State": "OutOfService",
      "Description": "Instance registration is still in progress."
    }
  ]
}
```

The following is an example response for an unhealthy instance.

Output:

```
{
  "InstanceStates": [
    {
      "InstanceId": "i-7299c809",
      "ReasonCode": "Instance",
      "State": "OutOfService",
      "Description": "Instance has failed at least the UnhealthyThreshold number
of health checks consecutively."
    }
  ]
}
```

- For API details, see [DescribeInstanceHealth](#) in *AWS CLI Command Reference*.

describe-load-balancer-attributes

The following code example shows how to use `describe-load-balancer-attributes`.

AWS CLI

To describe the attributes of a load balancer

This example describes the attributes of the specified load balancer.

Command:

```
aws elb describe-load-balancer-attributes --load-balancer-name my-load-balancer
```

Output:

```
{
  "LoadBalancerAttributes": {
    "ConnectionDraining": {
      "Enabled": false,
      "Timeout": 300
    },
    "CrossZoneLoadBalancing": {
      "Enabled": true
    },
    "ConnectionSettings": {
      "IdleTimeout": 30
    },
    "AccessLog": {
      "Enabled": false
    }
  }
}
```

- For API details, see [DescribeLoadBalancerAttributes](#) in *AWS CLI Command Reference*.

describe-load-balancer-policies

The following code example shows how to use `describe-load-balancer-policies`.

AWS CLI

To describe all policies associated with a load balancer

This example describes all of the policies associated with the specified load balancer.

Command:

```
aws elb describe-load-balancer-policies --load-balancer-name my-load-balancer
```

Output:

```
{
  "PolicyDescriptions": [
    {
      "PolicyAttributeDescriptions": [
        {
          "AttributeName": "ProxyProtocol",
          "AttributeValue": "true"
        }
      ],
      "PolicyName": "my-ProxyProtocol-policy",
      "PolicyTypeName": "ProxyProtocolPolicyType"
    },
    {
      "PolicyAttributeDescriptions": [
        {
          "AttributeName": "CookieName",
          "AttributeValue": "my-app-cookie"
        }
      ],
      "PolicyName": "my-app-cookie-policy",
      "PolicyTypeName": "AppCookieStickinessPolicyType"
    },
    {
      "PolicyAttributeDescriptions": [
        {
          "AttributeName": "CookieExpirationPeriod",
          "AttributeValue": "60"
        }
      ],
      "PolicyName": "my-duration-cookie-policy",
      "PolicyTypeName": "LBCookieStickinessPolicyType"
    },
    .
    .
    .
  ]
}
```

To describe a specific policy associated with a load balancer

This example describes the specified policy associated with the specified load balancer.

Command:

```
aws elb describe-load-balancer-policies --load-balancer-name my-load-balancer --  
policy-name my-authentication-policy
```

Output:

```
{  
  "PolicyDescriptions": [  
    {  
      "PolicyAttributeDescriptions": [  
        {  
          "AttributeName": "PublicKeyPolicyName",  
          "AttributeValue": "my-PublicKey-policy"  
        }  
      ],  
      "PolicyName": "my-authentication-policy",  
      "PolicyTypeName": "BackendServerAuthenticationPolicyType"  
    }  
  ]  
}
```

- For API details, see [DescribeLoadBalancerPolicies](#) in *AWS CLI Command Reference*.

describe-load-balancer-policy-types

The following code example shows how to use `describe-load-balancer-policy-types`.

AWS CLI

To describe the load balancer policy types defined by Elastic Load Balancing

This example describes the load balancer policy types that you can use to create policy configurations for your load balancer.

Command:

```
aws elb describe-load-balancer-policy-types
```


Output:

```
{
  "PolicyTypeDescriptions": [
    {
      "PolicyAttributeTypeDescriptions": [
        {
          "Cardinality": "ONE",
          "AttributeName": "ProxyProtocol",
          "AttributeType": "Boolean"
        }
      ],
      "PolicyTypeName": "ProxyProtocolPolicyType",
      "Description": "Policy that controls whether to include the IP address and
port of the originating request for TCP messages. This policy operates on TCP/SSL
listeners only"
    },
    {
      "PolicyAttributeTypeDescriptions": [
        {
          "Cardinality": "ONE",
          "AttributeName": "PublicKey",
          "AttributeType": "String"
        }
      ],
      "PolicyTypeName": "PublicKeyPolicyType",
      "Description": "Policy containing a list of public keys to
accept when authenticating the back-end server(s). This policy cannot be
applied directly to back-end servers or listeners but must be part of a
BackendServerAuthenticationPolicyType."
    },
    {
      "PolicyAttributeTypeDescriptions": [
        {
          "Cardinality": "ONE",
          "AttributeName": "CookieName",
          "AttributeType": "String"
        }
      ],
      "PolicyTypeName": "AppCookieStickinessPolicyType",
      "Description": "Stickiness policy with session lifetimes controlled by the
lifetime of the application-generated cookie. This policy can be associated only
with HTTP/HTTPS listeners."
    },
  ],
}
```

```

    {
      "PolicyAttributeTypeDescriptions": [
        {
          "Cardinality": "ZERO_OR_ONE",
          "AttributeName": "CookieExpirationPeriod",
          "AttributeType": "Long"
        }
      ],
      "PolicyTypeName": "LBCookieStickinessPolicyType",
      "Description": "Stickiness policy with session lifetimes controlled by
the browser (user-agent) or a specified expiration period. This policy can be
associated only with HTTP/HTTPS listeners."
    },
    {
      "PolicyAttributeTypeDescriptions": [
        .
        .
        .
      ],
      "PolicyTypeName": "SSLNegotiationPolicyType",
      "Description": "Listener policy that defines the ciphers and protocols
that will be accepted by the load balancer. This policy can be associated only with
HTTPS/SSL listeners."
    },
    {
      "PolicyAttributeTypeDescriptions": [
        {
          "Cardinality": "ONE_OR_MORE",
          "AttributeName": "PublicKeyPolicyName",
          "AttributeType": "PolicyName"
        }
      ],
      "PolicyTypeName": "BackendServerAuthenticationPolicyType",
      "Description": "Policy that controls authentication to back-end server(s)
and contains one or more policies, such as an instance of a PublicKeyPolicyType.
This policy can be associated only with back-end servers that are using HTTPS/SSL."
    }
  ]
}

```

- For API details, see [DescribeLoadBalancerPolicyTypes](#) in *AWS CLI Command Reference*.

describe-load-balancers

The following code example shows how to use describe-load-balancers.

AWS CLI

To describe your load balancers

This example describes all of your load balancers.

Command:

```
aws elb describe-load-balancers
```

To describe one of your load balancers

This example describes the specified load balancer.

Command:

```
aws elb describe-load-balancers --load-balancer-name my-load-balancer
```

The following example response is for an HTTPS load balancer in a VPC.

Output:

```
{
  "LoadBalancerDescriptions": [
    {
      "Subnets": [
        "subnet-15aaab61"
      ],
      "CanonicalHostedZoneNameID": "Z3DZXE0EXAMPLE",
      "CanonicalHostedZoneName": "my-load-balancer-1234567890.us-west-2.elb.amazonaws.com",
      "ListenerDescriptions": [
        {
          "Listener": {
            "InstancePort": 80,
            "LoadBalancerPort": 80,
            "Protocol": "HTTP",
            "InstanceProtocol": "HTTP"
          },
          "PolicyNames": []
        }
      ]
    }
  ]
}
```

```
    },
    {
      "Listener": {
        "InstancePort": 443,
        "SSLCertificateId": "arn:aws:iam::123456789012:server-certificate/
my-server-cert",
        "LoadBalancerPort": 443,
        "Protocol": "HTTPS",
        "InstanceProtocol": "HTTPS"
      },
      "PolicyNames": [
        "ELBSecurityPolicy-2015-03"
      ]
    }
  ],
  "HealthCheck": {
    "HealthyThreshold": 2,
    "Interval": 30,
    "Target": "HTTP:80/png",
    "Timeout": 3,
    "UnhealthyThreshold": 2
  },
  "VPCId": "vpc-a01106c2",
  "BackendServerDescriptions": [
    {
      "InstancePort": 80,
      "PolicyNames": [
        "my-ProxyProtocol-policy"
      ]
    }
  ],
  "Instances": [
    {
      "InstanceId": "i-207d9717"
    },
    {
      "InstanceId": "i-afefb49b"
    }
  ],
  "DNSName": "my-load-balancer-1234567890.us-west-2.elb.amazonaws.com",
  "SecurityGroups": [
    "sg-a61988c3"
  ],
  "Policies": {
```

```

    "LBCookieStickinessPolicies": [
      {
        "PolicyName": "my-duration-cookie-policy",
        "CookieExpirationPeriod": 60
      }
    ],
    "AppCookieStickinessPolicies": [],
    "OtherPolicies": [
      "my-PublicKey-policy",
      "my-authentication-policy",
      "my-SSLNegotiation-policy",
      "my-ProxyProtocol-policy",
      "ELBSecurityPolicy-2015-03"
    ]
  },
  "LoadBalancerName": "my-load-balancer",
  "CreatedTime": "2015-03-19T03:24:02.650Z",
  "AvailabilityZones": [
    "us-west-2a"
  ],
  "Scheme": "internet-facing",
  "SourceSecurityGroup": {
    "OwnerAlias": "123456789012",
    "GroupName": "my-elb-sg"
  }
}
]
}

```

- For API details, see [DescribeLoadBalancers](#) in *AWS CLI Command Reference*.

describe-tags

The following code example shows how to use describe-tags.

AWS CLI

To describe the tags assigned to a load balancer

This example describes the tags assigned to the specified load balancer.

Command:

```
aws elb describe-tags --load-balancer-name my-load-balancer
```

Output:

```
{
  "TagDescriptions": [
    {
      "Tags": [
        {
          "Value": "lima",
          "Key": "project"
        },
        {
          "Value": "digital-media",
          "Key": "department"
        }
      ],
      "LoadBalancerName": "my-load-balancer"
    }
  ]
}
```

- For API details, see [DescribeTags](#) in *AWS CLI Command Reference*.

detach-load-balancer-from-subnets

The following code example shows how to use `detach-load-balancer-from-subnets`.

AWS CLI**To detach load balancers from subnets**

This example detaches the specified load balancer from the specified subnet.

Command:

```
aws elb detach-load-balancer-from-subnets --load-balancer-name my-load-balancer --
subnets subnet-0ecac448
```

Output:

```
{
  "Subnets": [
    "subnet-15aaab61"
  ]
}
```

- For API details, see [DetachLoadBalancerFromSubnets](#) in *AWS CLI Command Reference*.

disable-availability-zones-for-load-balancer

The following code example shows how to use `disable-availability-zones-for-load-balancer`.

AWS CLI

To disable Availability Zones for a load balancer

This example removes the specified Availability Zone from the set of Availability Zones for the specified load balancer.

Command:

```
aws elb disable-availability-zones-for-load-balancer --load-balancer-name my-load-balancer --availability-zones us-west-2a
```

Output:

```
{
  "AvailabilityZones": [
    "us-west-2b"
  ]
}
```

- For API details, see [DisableAvailabilityZonesForLoadBalancer](#) in *AWS CLI Command Reference*.

enable-availability-zones-for-load-balancer

The following code example shows how to use `enable-availability-zones-for-load-balancer`.

AWS CLI

To enable Availability Zones for a load balancer

This example adds the specified Availability Zone to the specified load balancer.

Command:

```
aws elb enable-availability-zones-for-load-balancer --load-balancer-name my-load-balancer --availability-zones us-west-2b
```

Output:

```
{
  "AvailabilityZones": [
    "us-west-2a",
    "us-west-2b"
  ]
}
```

- For API details, see [EnableAvailabilityZonesForLoadBalancer](#) in *AWS CLI Command Reference*.

modify-load-balancer-attributes

The following code example shows how to use `modify-load-balancer-attributes`.

AWS CLI

To modify the attributes of a load balancer

This example modifies the `CrossZoneLoadBalancing` attribute of the specified load balancer.

Command:

```
aws elb modify-load-balancer-attributes --load-balancer-name my-load-balancer --load-balancer-attributes "{\"CrossZoneLoadBalancing\":{\"Enabled\":true}}"
```

Output:

```
{
  "LoadBalancerAttributes": {
    "CrossZoneLoadBalancing": {
```



```
        "Enabled": true
      }
    },
    "LoadBalancerName": "my-load-balancer"
  }
```

This example modifies the `ConnectionDraining` attribute of the specified load balancer.

Command:

```
aws elb modify-load-balancer-attributes --load-balancer-name my-load-balancer
--load-balancer-attributes "{\"ConnectionDraining\":{\"Enabled\":true,\"Timeout
\":300}}"
```

Output:

```
{
  "LoadBalancerAttributes": {
    "ConnectionDraining": {
      "Enabled": true,
      "Timeout": 300
    }
  },
  "LoadBalancerName": "my-load-balancer"
}
```

- For API details, see [ModifyLoadBalancerAttributes](#) in *AWS CLI Command Reference*.

register-instances-with-load-balancer

The following code example shows how to use `register-instances-with-load-balancer`.

AWS CLI

To register instances with a load balancer

This example registers the specified instance with the specified load balancer.

Command:

```
aws elb register-instances-with-load-balancer --load-balancer-name my-load-balancer
--instances i-d6f6fae3
```

Output:

```
{
  "Instances": [
    {
      "InstanceId": "i-d6f6fae3"
    },
    {
      "InstanceId": "i-207d9717"
    },
    {
      "InstanceId": "i-afefb49b"
    }
  ]
}
```

- For API details, see [RegisterInstancesWithLoadBalancer](#) in *AWS CLI Command Reference*.

remove-tags

The following code example shows how to use `remove-tags`.

AWS CLI**To remove tags from a load balancer**

This example removes a tag from the specified load balancer.

Command:

```
aws elb remove-tags --load-balancer-name my-load-balancer --tags project
```

- For API details, see [RemoveTags](#) in *AWS CLI Command Reference*.

set-load-balancer-listener-ssl-certificate

The following code example shows how to use `set-load-balancer-listener-ssl-certificate`.

AWS CLI**To update the SSL certificate for an HTTPS load balancer**

This example replaces the existing SSL certificate for the specified HTTPS load balancer.

Command:

```
aws elb set-load-balancer-listener-ssl-certificate --load-balancer-name my-load-balancer --load-balancer-port 443 --ssl-certificate-id arn:aws:iam::123456789012:server-certificate/new-server-cert
```

- For API details, see [SetLoadBalancerListenerSslCertificate](#) in *AWS CLI Command Reference*.

set-load-balancer-policies-for-backend-server

The following code example shows how to use `set-load-balancer-policies-for-backend-server`.

AWS CLI

To replace the policies associated with a port for a backend instance

This example replaces the policies that are currently associated with the specified port.

Command:

```
aws elb set-load-balancer-policies-for-backend-server --load-balancer-name my-load-balancer --instance-port 80 --policy-names my-ProxyProtocol-policy
```

To remove all policies that are currently associated with a port on your backend instance

This example removes all policies associated with the specified port.

Command:

```
aws elb set-load-balancer-policies-for-backend-server --load-balancer-name my-load-balancer --instance-port 80 --policy-names []
```

To confirm that the policies are removed, use the `describe-load-balancer-policies` command.

- For API details, see [SetLoadBalancerPoliciesForBackendServer](#) in *AWS CLI Command Reference*.

set-load-balancer-policies-of-listener

The following code example shows how to use `set-load-balancer-policies-of-listener`.

AWS CLI

To replace the policies associated with a listener

This example replaces the policies that are currently associated with the specified listener.

Command:

```
aws elb set-load-balancer-policies-of-listener --load-balancer-name my-load-balancer
--load-balancer-port 443 --policy-names my-SSLNegotiation-policy
```

To remove all policies associated with your listener

This example removes all policies that are currently associated with the specified listener.

Command:

```
aws elb set-load-balancer-policies-of-listener --load-balancer-name my-load-balancer
--load-balancer-port 443 --policy-names []
```

To confirm that the policies are removed from the load balancer, use the `describe-load-balancer-policies` command.

- For API details, see [SetLoadBalancerPoliciesOfListener](#) in *AWS CLI Command Reference*.

Elastic Load Balancing - Version 2 examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Elastic Load Balancing - Version 2.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-listener-certificates

The following code example shows how to use `add-listener-certificates`.

AWS CLI

To add a certificate to a secure listener

This example adds the specified certificate to the specified secure listener.

Command:

```
aws elbv2 add-listener-certificates --listener-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2 --certificates CertificateArn=arn:aws:acm:us-west-2:123456789012:certificate/5cc54884-f4a3-4072-80be-05b9ba72f705
```

Output:

```
{
  "Certificates": [
    {
      "CertificateArn": "arn:aws:acm:us-west-2:123456789012:certificate/5cc54884-f4a3-4072-80be-05b9ba72f705",
      "IsDefault": false
    }
  ]
}
```

- For API details, see [AddListenerCertificates](#) in *AWS CLI Command Reference*.

add-tags

The following code example shows how to use `add-tags`.

AWS CLI

To add tags to a load balancer

The following `add-tags` example adds the `project` and `department` tags to the specified load balancer.

```
aws elbv2 add-tags \  
  --resource-arns arn:aws:elasticloadbalancing:us-  
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188 \  
  --tags "Key=project,Value=lima" "Key=department,Value=digital-media"
```

- For API details, see [AddTags](#) in *AWS CLI Command Reference*.

create-listener

The following code example shows how to use `create-listener`.

AWS CLI

Example 1: To create an HTTP listener

The following `create-listener` example creates an HTTP listener for the specified Application Load Balancer that forwards requests to the specified target group.

```
aws elbv2 create-listener \  
  --load-balancer-arn arn:aws:elasticloadbalancing:us-  
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188 \  
  --protocol HTTP \  
  --port 80 \  
  --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

For more information, see [Tutorial: Create an Application Load Balancer using the AWS CLI](#) in the *User Guide for Application Load Balancers*.

Example 2: To create an HTTPS listener

The following `create-listener` example creates an HTTPS listener for the specified Application Load Balancer that forwards requests to the specified target group. You must specify an SSL certificate for an HTTPS listener. You can create and manage certificates using AWS Certificate Manager (ACM). Alternatively, you can create a certificate using SSL/TLS tools,

get the certificate signed by a certificate authority (CA), and upload the certificate to AWS Identity and Access Management (IAM).

```
aws elbv2 create-listener \  
  --load-balancer-arn arn:aws:elasticloadbalancing:us-  
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188 \  
  --protocol HTTPS \  
  --port 443 \  
  --certificates CertificateArn=arn:aws:acm:us-  
west-2:123456789012:certificate/3dcb0a41-bd72-4774-9ad9-756919c40557 \  
  --ssl-policy ELBSecurityPolicy-2016-08 \  
  --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

For more information, see [Add an HTTPS listener](#) in the *User Guide for Application Load Balancers*.

Example 3: To create a TCP listener

The following `create-listener` example creates a TCP listener for the specified Network Load Balancer that forwards requests to the specified target group.

```
aws elbv2 create-listener \  
  --load-balancer-arn arn:aws:elasticloadbalancing:us-  
west-2:123456789012:loadbalancer/net/my-network-load-balancer/5d1b75f4f1cee11e \  
  --protocol TCP \  
  --port 80 \  
  --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-tcp-targets/b6bba954d1361c78
```

For more information, see [Tutorial: Create a Network Load Balancer using the AWS CLI](#) in the *User Guide for Network Load Balancers*.

Example 4: To create a TLS listener

The following `create-listener` example creates a TLS listener for the specified Network Load Balancer that forwards requests to the specified target group. You must specify an SSL certificate for a TLS listener.

```
aws elbv2 create-listener \  
  --load-balancer-arn arn:aws:elasticloadbalancing:us-  
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188 \  
  --protocol TLS \  
  --port 443 \  
  --certificates CertificateArn=arn:aws:acm:us-  
west-2:123456789012:certificate/3dcb0a41-bd72-4774-9ad9-756919c40557 \  
  --ssl-policy ELBSecurityPolicy-2016-08 \  
  --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

```
--protocol TLS \  
--port 443 \  
--certificates CertificateArn=arn:aws:acm:us-  
west-2:123456789012:certificate/3dcb0a41-bd72-4774-9ad9-756919c40557 \  
--ssl-policy ELBSecurityPolicy-2016-08 \  
--default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

For more information, see [TLS listeners for your Network Load Balancer](#) in the *User Guide for Network Load Balancers*.

Example 5: To create a UDP listener

The following `create-listener` example creates a UDP listener for the specified Network Load Balancer that forwards requests to the specified target group.

```
aws elbv2 create-listener \  
  --load-balancer-arn arn:aws:elasticloadbalancing:us-  
west-2:123456789012:loadbalancer/net/my-network-load-balancer/5d1b75f4f1cee11e \  
  --protocol UDP \  
  --port 53 \  
  --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-tcp-targets/b6bba954d1361c78
```

For more information, see [Tutorial: Create a Network Load Balancer using the AWS CLI](#) in the *User Guide for Network Load Balancers*.

Example 6: To create a listener for the specified gateway and forwarding

The following `create-listener` example creates a listener for the specified Gateway Load Balancer that forwards requests to the specified target group.

```
aws elbv2 create-listener \  
  --load-balancer-arn arn:aws:elasticloadbalancing:us-  
east-1:850631746142:loadbalancer/gwy/my-gateway-load-balancer/e0f9b3d5c7f7d3d6 \  
  --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-  
east-1:850631746142:targetgroup/my-glb-targets/007ca469fae3bb1615
```

Output:

```
{
```



```

    "Listeners": [
      {
        "ListenerArn": "arn:aws:elasticloadbalancing:us-
east-1:850631746142:listener/gwy/my-agw-lb-example2/e0f9b3d5c7f7d3d6/
afc127db15f925de",
        "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-
east-1:850631746142:loadbalancer/gwy/my-agw-lb-example2/e0f9b3d5c7f7d3d6",
        "DefaultActions": [
          {
            "Type": "forward",
            "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
east-1:850631746142:targetgroup/test-tg-agw-2/007ca469fae3bb1615",
            "ForwardConfig": {
              "TargetGroups": [
                {
                  "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
east-1:850631746142:targetgroup/test-tg-agw-2/007ca469fae3bb1615"
                }
              ]
            }
          }
        ]
      }
    ]
  }
}

```

For more information, see [Getting started with Gateway Load Balancers using the AWS CLI](#) in the *User Guide for Gateway Load Balancers*.

- For API details, see [CreateListener](#) in *AWS CLI Command Reference*.

create-load-balancer

The following code example shows how to use `create-load-balancer`.

AWS CLI

Example 1: To create an Internet-facing load balancer

The following `create-load-balancer` example creates an Internet-facing Application Load Balancer and enables the Availability Zones for the specified subnets.

```
aws elbv2 create-load-balancer \
```

```
--name my-load-balancer \  
--subnets subnet-b7d581c0 subnet-8360a9e7
```

Output:

```
{  
  "LoadBalancers": [  
    {  
      "Type": "application",  
      "Scheme": "internet-facing",  
      "IpAddressType": "ipv4",  
      "VpcId": "vpc-3ac0fb5f",  
      "AvailabilityZones": [  
        {  
          "ZoneName": "us-west-2a",  
          "SubnetId": "subnet-8360a9e7"  
        },  
        {  
          "ZoneName": "us-west-2b",  
          "SubnetId": "subnet-b7d581c0"  
        }  
      ],  
      "CreatedTime": "2017-08-25T21:26:12.920Z",  
      "CanonicalHostedZoneId": "Z2P70J7EXAMPLE",  
      "DNSName": "my-load-balancer-424835706.us-west-2.elb.amazonaws.com",  
      "SecurityGroups": [  
        "sg-5943793c"  
      ],  
      "LoadBalancerName": "my-load-balancer",  
      "State": {  
        "Code": "provisioning"  
      },  
      "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-  
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188"  
    }  
  ]  
}
```

For more information, see [Tutorial: Create an Application Load Balancer using the AWS CLI](#) in the *User Guide for Application Load Balancers*.

Example 2: To create an internal load balancer

The following `create-load-balancer` example creates an internal Application Load Balancer and enables the Availability Zones for the specified subnets.

```
aws elbv2 create-load-balancer \  
  --name my-internal-load-balancer \  
  --scheme internal \  
  --subnets subnet-b7d581c0 subnet-8360a9e7
```

Output:

```
{  
  "LoadBalancers": [  
    {  
      "Type": "application",  
      "Scheme": "internal",  
      "IpAddressType": "ipv4",  
      "VpcId": "vpc-3ac0fb5f",  
      "AvailabilityZones": [  
        {  
          "ZoneName": "us-west-2a",  
          "SubnetId": "subnet-8360a9e7"  
        },  
        {  
          "ZoneName": "us-west-2b",  
          "SubnetId": "subnet-b7d581c0"  
        }  
      ],  
      "CreatedTime": "2016-03-25T21:29:48.850Z",  
      "CanonicalHostedZoneId": "Z2P70J7EXAMPLE",  
      "DNSName": "internal-my-internal-load-balancer-1529930873.us-  
west-2.elb.amazonaws.com",  
      "SecurityGroups": [  
        "sg-5943793c"  
      ],  
      "LoadBalancerName": "my-internal-load-balancer",  
      "State": {  
        "Code": "provisioning"  
      },  
      "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-  
west-2:123456789012:loadbalancer/app/my-internal-load-balancer/5b49b8d4303115c2"  
    }  
  ]  
}
```

```
}
```

For more information, see [Tutorial: Create an Application Load Balancer using the AWS CLI](#) in the *User Guide for Application Load Balancers*.

Example 3: To create a Network Load Balancer

The following `create-load-balancer` example creates an Internet-facing Network Load Balancer and enables the Availability Zone for the specified subnet. It uses a subnet mapping to associate the specified Elastic IP address with the network interface used by the load balancer nodes for the Availability Zone.

```
aws elbv2 create-load-balancer \  
  --name my-network-load-balancer \  
  --type network \  
  --subnet-mappings SubnetId=subnet-b7d581c0,AllocationId=eipalloc-64d5890a
```

Output:

```
{  
  "LoadBalancers": [  
    {  
      "Type": "network",  
      "Scheme": "internet-facing",  
      "IpAddressType": "ipv4",  
      "VpcId": "vpc-3ac0fb5f",  
      "AvailabilityZones": [  
        {  
          "LoadBalancerAddresses": [  
            {  
              "IpAddress": "35.161.207.171",  
              "AllocationId": "eipalloc-64d5890a"  
            }  
          ],  
          "ZoneName": "us-west-2b",  
          "SubnetId": "subnet-5264e837"  
        }  
      ],  
      "CreatedTime": "2017-10-15T22:41:25.657Z",  
      "CanonicalHostedZoneId": "Z2P70J7EXAMPLE",  
      "DNSName": "my-network-load-balancer-5d1b75f4f1cee11e.elb.us-  
west-2.amazonaws.com",
```

```

        "LoadBalancerName": "my-network-load-balancer",
        "State": {
            "Code": "provisioning"
        },
        "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/net/my-network-load-balancer/5d1b75f4f1cee11e"
    }
]
}

```

For more information, see [Tutorial: Create a Network Load Balancer using the AWS CLI](#) in the *User Guide for Network Load Balancers*.

Example 4: To create a Gateway Load Balancer

The following `create-load-balancer` example creates a Gateway Load Balancer and enables the Availability Zones for the specified subnets.

```

aws elbv2 create-load-balancer \
  --name my-gateway-load-balancer \
  --type gateway \
  --subnets subnet-dc83f691 subnet-a62583f9

```

Output:

```

{
  "LoadBalancers": [
    {
      "Type": "gateway",
      "VpcId": "vpc-838475fe",
      "AvailabilityZones": [
        {
          "ZoneName": "us-east-1b",
          "SubnetId": "subnet-a62583f9"
        },
        {
          "ZoneName": "us-east-1a",
          "SubnetId": "subnet-dc83f691"
        }
      ],
      "CreatedTime": "2021-07-14T19:33:43.324000+00:00",
      "LoadBalancerName": "my-gateway-load-balancer",
      "State": {

```

```

        "Code": "provisioning"
      },
      "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-
east-1:850631746142:loadbalancer/gwy/my-gateway-load-balancer/dfbb5a7d32cdee79"
    }
  ]
}

```

For more information, see [Getting started with Gateway Load Balancers using the AWS CLI](#) in the *User Guide for Gateway Load Balancers*.

- For API details, see [CreateLoadBalancer](#) in *AWS CLI Command Reference*.

create-rule

The following code example shows how to use `create-rule`.

AWS CLI

Example 1: To create a rule using a path condition and a forward action

The following `create-rule` example creates a rule that forwards requests to the specified target group if the URL contains the specified pattern.

```

aws elbv2 create-rule \
  --listener-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/
my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2 \
  --priority 5 \
  --conditions file://conditions-pattern.json
  --actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067

```

Contents of `conditions-pattern.json`:

```

[
  {
    "Field": "path-pattern",
    "PathPatternConfig": {
      "Values": ["/images/*"]
    }
  }
]

```

Example 2: To create a rule using a host condition and a fixed response

The following `create-rule` example creates a rule that provides a fixed response if the hostname in the host header matches the specified hostname.

```
aws elbv2 create-rule \  
  --listener-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/  
my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2 \  
  --priority 10 \  
  --conditions file://conditions-host.json \  
  --actions file://actions-fixed-response.json
```

Contents of `conditions-host.json`

```
[  
  {  
    "Field": "host-header",  
    "HostHeaderConfig": {  
      "Values": ["*.example.com"]  
    }  
  }  
]
```

Contents of `actions-fixed-response.json`

```
[  
  {  
    "Type": "fixed-response",  
    "FixedResponseConfig": {  
      "MessageBody": "Hello world",  
      "StatusCode": "200",  
      "ContentType": "text/plain"  
    }  
  }  
]
```

Example 3: To create a rule using a source IP address condition, an authenticate action, and a forward action

The following `create-rule` example creates a rule that authenticates the user if the source IP address matches the specified IP address, and forwards the request to the specified target group if authentication is successful.

```
aws elbv2 create-rule \  
  --listener-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/  
my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2 \  
  --priority 20 \  
  --conditions file://conditions-source-ip.json \  
  --actions file://actions-authenticate.json
```

Contents of conditions-source-ip.json

```
[  
  {  
    "Field": "source-ip",  
    "SourceIpConfig": {  
      "Values": ["192.0.2.0/24", "198.51.100.10/32"]  
    }  
  }  
]
```

Contents of actions-authenticate.json

```
[  
  {  
    "Type": "authenticate-oidc",  
    "AuthenticateOidcConfig": {  
      "Issuer": "https://idp-issuer.com",  
      "AuthorizationEndpoint": "https://authorization-endpoint.com",  
      "TokenEndpoint": "https://token-endpoint.com",  
      "UserInfoEndpoint": "https://user-info-endpoint.com",  
      "ClientId": "abcdefghijklmnopqrstuvwxy123456789",  
      "ClientSecret": "123456789012345678901234567890",  
      "SessionCookieName": "my-cookie",  
      "SessionTimeout": 3600,  
      "Scope": "email",  
      "AuthenticationRequestExtraParams": {  
        "display": "page",  
        "prompt": "login"  
      },  
      "OnUnauthenticatedRequest": "deny"  
    },  
    "Order": 1  
  },  
  {  
    "Type": "authenticate-oidc",  
    "AuthenticateOidcConfig": {  
      "Issuer": "https://idp-issuer.com",  
      "AuthorizationEndpoint": "https://authorization-endpoint.com",  
      "TokenEndpoint": "https://token-endpoint.com",  
      "UserInfoEndpoint": "https://user-info-endpoint.com",  
      "ClientId": "abcdefghijklmnopqrstuvwxy123456789",  
      "ClientSecret": "123456789012345678901234567890",  
      "SessionCookieName": "my-cookie",  
      "SessionTimeout": 3600,  
      "Scope": "email",  
      "AuthenticationRequestExtraParams": {  
        "display": "page",  
        "prompt": "login"  
      },  
      "OnUnauthenticatedRequest": "deny"  
    },  
    "Order": 2  
  }  
]
```



```
    "Type": "forward",
    "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
east-1:880185128111:targetgroup/cli-test/642a97ecb0e0f26b",
    "Order": 2
  }
]
```

- For API details, see [CreateRule](#) in *AWS CLI Command Reference*.

create-target-group

The following code example shows how to use `create-target-group`.

AWS CLI

Example 1: To create a target group for an Application Load Balancer

The following `create-target-group` example creates a target group for an Application Load Balancer where you register targets by instance ID (the target type is `instance`). This target group uses the HTTP protocol, port 80, and the default health check settings for an HTTP target group.

```
aws elbv2 create-target-group \
  --name my-targets \
  --protocol HTTP \
  --port 80 \
  --target-type instance \
  --vpc-id vpc-3ac0fb5f
```

Output:

```
{
  "TargetGroups": [
    {
      "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067",
      "TargetGroupName": "my-targets",
      "Protocol": "HTTP",
      "Port": 80,
      "VpcId": "vpc-3ac0fb5f",
      "HealthCheckProtocol": "HTTP",
      "HealthCheckPort": "traffic-port",
```

```

    "HealthCheckEnabled": true,
    "HealthCheckIntervalSeconds": 30,
    "HealthCheckTimeoutSeconds": 5,
    "HealthyThresholdCount": 5,
    "UnhealthyThresholdCount": 2,
    "HealthCheckPath": "/",
    "Matcher": {
      "HttpCode": "200"
    },
    "TargetType": "instance",
    "ProtocolVersion": "HTTP1",
    "IpAddressType": "ipv4"
  }
]
}

```

For more information, see [Create a target group](#) in the *User Guide for Application Load Balancers*.

Example 2: To create a target group to route traffic from an Application Load Balancer to a Lambda function

The following `create-target-group` example creates a target group for an Application Load Balancer where the target is a Lambda function (the target type is `lambda`). Health checks are disabled for this target group by default.

```

aws elbv2 create-target-group \
  --name my-lambda-target \
  --target-type lambda

```

Output:

```

{
  "TargetGroups": [
    {
      "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-lambda-target/a3003e085dbb8ddc",
      "TargetGroupName": "my-lambda-target",
      "HealthCheckEnabled": false,
      "HealthCheckIntervalSeconds": 35,
      "HealthCheckTimeoutSeconds": 30,
      "HealthyThresholdCount": 5,

```

```

        "UnhealthyThresholdCount": 2,
        "HealthCheckPath": "/",
        "Matcher": {
            "HttpCode": "200"
        },
        "TargetType": "lambda",
        "IpAddressType": "ipv4"
    }
]
}

```

For more information, see [Lambda functions as targets](#) in the *User Guide for Application Load Balancers*.

Example 3: To create a target group for a Network Load Balancer

The following `create-target-group` example creates a target group for a Network Load Balancer where you register targets by IP address (the target type is `ip`). This target group uses the TCP protocol, port 80, and the default health check settings for a TCP target group.

```

aws elbv2 create-target-group \
  --name my-ip-targets \
  --protocol TCP \
  --port 80 \
  --target-type ip \
  --vpc-id vpc-3ac0fb5f

```

Output:

```

{
  "TargetGroups": [
    {
      "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-ip-targets/b6bba954d1361c78",
      "TargetGroupName": "my-ip-targets",
      "Protocol": "TCP",
      "Port": 80,
      "VpcId": "vpc-3ac0fb5f",
      "HealthCheckEnabled": true,
      "HealthCheckProtocol": "TCP",
      "HealthCheckPort": "traffic-port",
      "HealthCheckIntervalSeconds": 30,
    }
  ]
}

```

```

        "HealthCheckTimeoutSeconds": 10,
        "HealthyThresholdCount": 5,
        "UnhealthyThresholdCount": 2,
        "TargetType": "ip",
        "IpAddressType": "ipv4"
    }
]
}

```

For more information, see [Create a target group](#) in the *User Guide for Network Load Balancers*.

Example 4: To create a target group to route traffic from a Network Load Balancer to an Application Load Balancer

The following `create-target-group` example creates a target group for a Network Load Balancer where you register an Application Load Balancer as a target (the target type is `alb`).

```
aws elbv2 create-target-group --name my-alb-target --protocol TCP --port 80 --target-type alb
--vpc-id vpc-3ac0fb5f
```

Output:

```

{
  "TargetGroups": [
    {
      "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-alb-target/a3003e085dbb8ddc",
      "TargetGroupName": "my-alb-target",
      "Protocol": "TCP",
      "Port": 80,
      "VpcId": "vpc-838475fe",
      "HealthCheckProtocol": "HTTP",
      "HealthCheckPort": "traffic-port",
      "HealthCheckEnabled": true,
      "HealthCheckIntervalSeconds": 30,
      "HealthCheckTimeoutSeconds": 6,
      "HealthyThresholdCount": 5,
      "UnhealthyThresholdCount": 2,
      "HealthCheckPath": "/",
      "Matcher": {
        "HttpCode": "200-399"
      },
      "TargetType": "alb",
    }
  ]
}

```

```

        "IpAddressType": "ipv4"
      }
    ]
  }

```

For more information, see [Create a target group with an Application Load Balancer as the target](#) in the *User Guide for Network Load Balancers*.

Example 5: To create a target group for a Gateway Load Balancer

The following `create-target-group` example creates a target group for a Gateway Load Balancer where the target is an instance, and the target group protocol is GENEVE.

```

aws elbv2 create-target-group \
  --name my-glb-targetgroup \
  --protocol GENEVE \
  --port 6081 \
  --target-type instance \
  --vpc-id vpc-838475fe

```

Output:

```

{
  "TargetGroups": [
    {
      "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-glb-targetgroup/00c3d57eacd6f40b6f",
      "TargetGroupName": "my-glb-targetgroup",
      "Protocol": "GENEVE",
      "Port": 6081,
      "VpcId": "vpc-838475fe",
      "HealthCheckProtocol": "TCP",
      "HealthCheckPort": "80",
      "HealthCheckEnabled": true,
      "HealthCheckIntervalSeconds": 10,
      "HealthCheckTimeoutSeconds": 5,
      "HealthyThresholdCount": 5,
      "UnhealthyThresholdCount": 2,
      "TargetType": "instance"
    }
  ]
}

```

For more information, see [Create a target group <https://docs.aws.amazon.com/elasticloadbalancing/latest/gateway/create-target-group.html>](https://docs.aws.amazon.com/elasticloadbalancing/latest/gateway/create-target-group.html) in the *Gateway Load Balancer User Guide*.

- For API details, see [CreateTargetGroup](#) in *AWS CLI Command Reference*.

delete-listener

The following code example shows how to use `delete-listener`.

AWS CLI

To delete a listener

The following `delete-listener` example deletes the specified listener.

```
aws elbv2 delete-listener \  
  --listener-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/  
my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2
```

- For API details, see [DeleteListener](#) in *AWS CLI Command Reference*.

delete-load-balancer

The following code example shows how to use `delete-load-balancer`.

AWS CLI

To delete a load balancer

The following `delete-load-balancer` example deletes the specified load balancer.

```
aws elbv2 delete-load-balancer \  
  --load-balancer-arn arn:aws:elasticloadbalancing:us-  
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188
```

- For API details, see [DeleteLoadBalancer](#) in *AWS CLI Command Reference*.

delete-rule

The following code example shows how to use `delete-rule`.

AWS CLI

To delete a rule

The following `delete-rule` example deletes the specified rule.

```
aws elbv2 delete-rule \  
  --rule-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener-rule/  
  app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2/1291d13826f405c3
```

- For API details, see [DeleteRule](#) in *AWS CLI Command Reference*.

`delete-target-group`

The following code example shows how to use `delete-target-group`.

AWS CLI

To delete a target group

The following `delete-target-group` example deletes the specified target group.

```
aws elbv2 delete-target-group \  
  --target-group-arn arn:aws:elasticloadbalancing:us-  
  west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

This command produces no output.

For more information, see [Delete a load balancer](#) in the *Application Load Balancer Guide*.

- For API details, see [DeleteTargetGroup](#) in *AWS CLI Command Reference*.

`deregister-targets`

The following code example shows how to use `deregister-targets`.

AWS CLI

Example 1: To deregister a target from a target group

The following `deregister-targets` example removes the specified instance from the specified target group.

```
aws elbv2 deregister-targets \  
  --target-group-arn arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067 \  
  --targets Id=i-1234567890abcdef0
```

Example 2: To deregister a target registered using port overrides

The following `deregister-targets` example removes an instance from a target group that was registered using port overrides.

```
aws elbv2 deregister-targets \  
  --target-group-arn arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-internal-targets/3bb63f11dfb0faf9 \  
  --targets Id=i-1234567890abcdef0,Port=80 Id=i-1234567890abcdef0,Port=766
```

- For API details, see [DeregisterTargets](#) in *AWS CLI Command Reference*.

describe-account-limits

The following code example shows how to use `describe-account-limits`.

AWS CLI

To describe your Elastic Load Balancing limits

The following `describe-account-limits` example displays the Elastic Load Balancing limits for your AWS account in the current Region.

```
aws elbv2 describe-account-limits
```

Output:

```
{  
  "Limits": [  
    {  
      "Name": "target-groups",  
      "Max": "3000"  
    },  
    {  
      "Name": "targets-per-application-load-balancer",  
      "Max": "1000"  
    }  
  ]  
}
```



```
  },
  {
    "Name": "listeners-per-application-load-balancer",
    "Max": "50"
  },
  {
    "Name": "rules-per-application-load-balancer",
    "Max": "100"
  },
  {
    "Name": "network-load-balancers",
    "Max": "50"
  },
  {
    "Name": "targets-per-network-load-balancer",
    "Max": "3000"
  },
  {
    "Name": "targets-per-availability-zone-per-network-load-balancer",
    "Max": "500"
  },
  {
    "Name": "listeners-per-network-load-balancer",
    "Max": "50"
  },
  {
    "Name": "condition-values-per-alb-rule",
    "Max": "5"
  },
  {
    "Name": "condition-wildcards-per-alb-rule",
    "Max": "5"
  },
  {
    "Name": "target-groups-per-application-load-balancer",
    "Max": "100"
  },
  {
    "Name": "target-groups-per-action-on-application-load-balancer",
    "Max": "5"
  },
  {
    "Name": "target-groups-per-action-on-network-load-balancer",
    "Max": "1"
```

```
    },
    {
      "Name": "certificates-per-application-load-balancer",
      "Max": "25"
    },
    {
      "Name": "certificates-per-network-load-balancer",
      "Max": "25"
    },
    {
      "Name": "targets-per-target-group",
      "Max": "1000"
    },
    {
      "Name": "target-id-registrations-per-application-load-balancer",
      "Max": "1000"
    },
    {
      "Name": "network-load-balancer-enis-per-vpc",
      "Max": "1200"
    },
    {
      "Name": "application-load-balancers",
      "Max": "50"
    },
    {
      "Name": "gateway-load-balancers",
      "Max": "100"
    },
    {
      "Name": "gateway-load-balancers-per-vpc",
      "Max": "100"
    },
    {
      "Name": "geneve-target-groups",
      "Max": "100"
    },
    {
      "Name": "targets-per-availability-zone-per-gateway-load-balancer",
      "Max": "300"
    }
  ]
}
```

For more information, see [Quotas](#) in the *AWS General Reference*.

- For API details, see [DescribeAccountLimits](#) in *AWS CLI Command Reference*.

describe-listener-certificates

The following code example shows how to use `describe-listener-certificates`.

AWS CLI

To describe the certificates for a secure listener

This example describes the certificates for the specified secure listener.

Command:

```
aws elbv2 describe-listener-certificates --listener-arn
arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/my-load-
balancer/50dc6c495c0c9188/f2f7dc8efc522ab2
```

Output:

```
{
  "Certificates": [
    {
      "CertificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/5cc54884-f4a3-4072-80be-05b9ba72f705",
      "IsDefault": false
    },
    {
      "CertificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/3dcb0a41-bd72-4774-9ad9-756919c40557",
      "IsDefault": false
    },
    {
      "CertificateArn": "arn:aws:acm:us-west-2:123456789012:certificate/
fe59da96-6f58-4a22-8eed-6d0d50477e1d",
      "IsDefault": true
    }
  ]
}
```

- For API details, see [DescribeListenerCertificates](#) in *AWS CLI Command Reference*.

describe-listeners

The following code example shows how to use `describe-listeners`.

AWS CLI

To describe a listener

This example describes the specified listener.

Command:

```
aws elbv2 describe-listeners --listener-arns arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2
```

Output:

```
{
  "Listeners": [
    {
      "Port": 80,
      "Protocol": "HTTP",
      "DefaultActions": [
        {
          "TargetGroupArn": "arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067",
          "Type": "forward"
        }
      ],
      "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188",
      "ListenerArn": "arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2"
    }
  ]
}
```

To describe the listeners for a load balancer

This example describe the listeners for the specified load balancer.

Command:

```
aws elbv2 describe-listeners --load-balancer-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188
```

Output:

```
{
  "Listeners": [
    {
      "Port": 443,
      "Protocol": "HTTPS",
      "DefaultActions": [
        {
          "TargetGroupArn": "arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067",
          "Type": "forward"
        }
      ],
      "SslPolicy": "ELBSecurityPolicy-2015-05",
      "Certificates": [
        {
          "CertificateArn": "arn:aws:iam::123456789012:server-certificate/my-server-cert"
        }
      ],
      "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188",
      "ListenerArn": "arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/0467ef3c8400ae65"
    },
    {
      "Port": 80,
      "Protocol": "HTTP",
      "DefaultActions": [
        {
          "TargetGroupArn": "arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067",
          "Type": "forward"
        }
      ],
      "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188",
      "ListenerArn": "arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2"
    }
  ]
}
```

```

    }
  ]
}

```

- For API details, see [DescribeListeners](#) in *AWS CLI Command Reference*.

describe-load-balancer-attributes

The following code example shows how to use `describe-load-balancer-attributes`.

AWS CLI

To describe load balancer attributes

The following `describe-load-balancer-attributes` example displays the attributes of the specified load balancer.

```

aws elbv2 describe-load-balancer-attributes \
  --load-balancer-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188

```

The following example output show the attributes for an Application Load Balancer.

```

{
  "Attributes": [
    {
      "Value": "false",
      "Key": "access_logs.s3.enabled"
    },
    {
      "Value": "",
      "Key": "access_logs.s3.bucket"
    },
    {
      "Value": "",
      "Key": "access_logs.s3.prefix"
    },
    {
      "Value": "60",
      "Key": "idle_timeout.timeout_seconds"
    },
    {

```

```

        "Value": "false",
        "Key": "deletion_protection.enabled"
    },
    {
        "Value": "true",
        "Key": "routing.http2.enabled"
    }
]
}

```

The following example output includes the attributes for a Network Load Balancer.

```

{
  "Attributes": [
    {
      "Value": "false",
      "Key": "access_logs.s3.enabled"
    },
    {
      "Value": "",
      "Key": "access_logs.s3.bucket"
    },
    {
      "Value": "",
      "Key": "access_logs.s3.prefix"
    },
    {
      "Value": "false",
      "Key": "deletion_protection.enabled"
    },
    {
      "Value": "false",
      "Key": "load_balancing.cross_zone.enabled"
    }
  ]
}

```

- For API details, see [DescribeLoadBalancerAttributes](#) in *AWS CLI Command Reference*.

describe-load-balancers

The following code example shows how to use describe-load-balancers.

AWS CLI

To describe a load balancer

This example describes the specified load balancer.

Command:

```
aws elbv2 describe-load-balancers --load-balancer-arns
arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-
balancer/50dc6c495c0c9188
```

Output:

```
{
  "LoadBalancers": [
    {
      "Type": "application",
      "Scheme": "internet-facing",
      "IpAddressType": "ipv4",
      "VpcId": "vpc-3ac0fb5f",
      "AvailabilityZones": [
        {
          "ZoneName": "us-west-2a",
          "SubnetId": "subnet-8360a9e7"
        },
        {
          "ZoneName": "us-west-2b",
          "SubnetId": "subnet-b7d581c0"
        }
      ],
      "CreatedTime": "2016-03-25T21:26:12.920Z",
      "CanonicalHostedZoneId": "Z2P70J7EXAMPLE",
      "DNSName": "my-load-balancer-424835706.us-west-2.elb.amazonaws.com",
      "SecurityGroups": [
        "sg-5943793c"
      ],
      "LoadBalancerName": "my-load-balancer",
      "State": {
        "Code": "active"
      },
      "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188"
    }
  ]
}
```



```
]
}
```

To describe all load balancers

This example describes all of your load balancers.

Command:

```
aws elbv2 describe-load-balancers
```

- For API details, see [DescribeLoadBalancers](#) in *AWS CLI Command Reference*.

describe-rules

The following code example shows how to use `describe-rules`.

AWS CLI

Example 1: To describe a rule

The following `describe-rules` example displays details for the specified rule.

```
aws elbv2 describe-rules \  
  --rule-arns arn:aws:elasticloadbalancing:us-west-2:123456789012:listener-rule/  
  app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2/9683b2d02a6cabee
```

Example 2: To describe the rules for a listener

The following `describe-rules` example displays details for the rules for the specified listener. The output includes the default rule and any other rules that you've added.

```
aws elbv2 describe-rules \  
  --listener-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/  
  my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2
```

- For API details, see [DescribeRules](#) in *AWS CLI Command Reference*.

describe-ssl-policies

The following code example shows how to use `describe-ssl-policies`.

AWS CLI

Example 1: To list the policies used for SSL negotiation by load balancer type

The following `describe-ssl-policies` example displays the names of the policies that you can use for SSL negotiation with an Application Load Balancer. The example uses the `--query` parameter to display only the names of the policies.

```
aws elbv2 describe-ssl-policies \  
  --load-balancer-type application \  
  --query SslPolicies[*].Name
```

Output:

```
[  
  "ELBSecurityPolicy-2016-08",  
  "ELBSecurityPolicy-TLS13-1-2-2021-06",  
  "ELBSecurityPolicy-TLS13-1-2-Res-2021-06",  
  "ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06",  
  "ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06",  
  "ELBSecurityPolicy-TLS13-1-1-2021-06",  
  "ELBSecurityPolicy-TLS13-1-0-2021-06",  
  "ELBSecurityPolicy-TLS13-1-3-2021-06",  
  "ELBSecurityPolicy-TLS-1-2-2017-01",  
  "ELBSecurityPolicy-TLS-1-1-2017-01",  
  "ELBSecurityPolicy-TLS-1-2-Ext-2018-06",  
  "ELBSecurityPolicy-FS-2018-06",  
  "ELBSecurityPolicy-2015-05",  
  "ELBSecurityPolicy-TLS-1-0-2015-04",  
  "ELBSecurityPolicy-FS-1-2-Res-2019-08",  
  "ELBSecurityPolicy-FS-1-1-2019-08",  
  "ELBSecurityPolicy-FS-1-2-2019-08",  
  "ELBSecurityPolicy-FS-1-2-Res-2020-10"  
]
```

Example 2: To list the policies that support a specific protocol

The following `describe-ssl-policies` example displays the names of the policies that support the TLS 1.3 protocol. The example uses the `--query` parameter to display only the names of the policies.

```
aws elbv2 describe-ssl-policies \  
  --query SslPolicies[*].Name
```

```
--load-balancer-type application \  
--query SslPolicies[?contains(SslProtocols, 'TLSv1.3')].Name
```

Output:

```
[  
  "ELBSecurityPolicy-TLS13-1-2-2021-06",  
  "ELBSecurityPolicy-TLS13-1-2-Res-2021-06",  
  "ELBSecurityPolicy-TLS13-1-2-Ext1-2021-06",  
  "ELBSecurityPolicy-TLS13-1-2-Ext2-2021-06",  
  "ELBSecurityPolicy-TLS13-1-1-2021-06",  
  "ELBSecurityPolicy-TLS13-1-0-2021-06",  
  "ELBSecurityPolicy-TLS13-1-3-2021-06"  
]
```

Example 3: To display the ciphers for a policy

The following `describe-ssl-policies` example displays the names of the ciphers for the specified policy. The example uses the `--query` parameter to display only the cipher names. The first cipher in the list has priority 1, and the remaining ciphers are in priority order.

```
aws elbv2 describe-ssl-policies \  
  --names ELBSecurityPolicy-TLS13-1-2-2021-06 \  
  --query SslPolicies[*].Ciphers[*].Name
```

Output:

```
[  
  "TLS_AES_128_GCM_SHA256",  
  "TLS_AES_256_GCM_SHA384",  
  "TLS_CHACHA20_POLY1305_SHA256",  
  "ECDHE-ECDSA-AES128-GCM-SHA256",  
  "ECDHE-RSA-AES128-GCM-SHA256",  
  "ECDHE-ECDSA-AES128-SHA256",  
  "ECDHE-RSA-AES128-SHA256",  
  "ECDHE-ECDSA-AES256-GCM-SHA384",  
  "ECDHE-RSA-AES256-GCM-SHA384",  
  "ECDHE-ECDSA-AES256-SHA384",  
  "ECDHE-RSA-AES256-SHA384"  
]
```

For more information, see [Security policies](#) in the *User Guide for Application Load Balancers*.

- For API details, see [DescribeSslPolicies](#) in *AWS CLI Command Reference*.

describe-tags

The following code example shows how to use `describe-tags`.

AWS CLI

To describe the tags assigned to a load balancer

This example describes the tags assigned to the specified load balancer.

Command:

```
aws elbv2 describe-tags --resource-arns arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188
```

Output:

```
{
  "TagDescriptions": [
    {
      "ResourceArn": "arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188",
      "Tags": [
        {
          "Value": "lima",
          "Key": "project"
        },
        {
          "Value": "digital-media",
          "Key": "department"
        }
      ]
    }
  ]
}
```

- For API details, see [DescribeTags](#) in *AWS CLI Command Reference*.

describe-target-group-attributes

The following code example shows how to use `describe-target-group-attributes`.

AWS CLI

To describe target group attributes

The following `describe-target-group-attributes` example displays the attributes of the specified target group.

```
aws elbv2 describe-target-group-attributes \  
  --target-group-arn arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

The output includes the attributes if the protocol is HTTP or HTTPS and the target type is instance or ip.

```
{  
  "Attributes": [  
    {  
      "Value": "false",  
      "Key": "stickiness.enabled"  
    },  
    {  
      "Value": "300",  
      "Key": "deregistration_delay.timeout_seconds"  
    },  
    {  
      "Value": "lb_cookie",  
      "Key": "stickiness.type"  
    },  
    {  
      "Value": "86400",  
      "Key": "stickiness.lb_cookie.duration_seconds"  
    },  
    {  
      "Value": "0",  
      "Key": "slow_start.duration_seconds"  
    }  
  ]  
}
```

The following output includes the attributes if the protocol is HTTP or HTTPS and the target type is `lambda`.

```
{
  "Attributes": [
    {
      "Value": "false",
      "Key": "lambda.multi_value_headers.enabled"
    }
  ]
}
```

The following output includes the attributes if the protocol is TCP, TLS, UDP, or TCP_UDP.

```
{
  "Attributes": [
    {
      "Value": "false",
      "Key": "proxy_protocol_v2.enabled"
    },
    {
      "Value": "300",
      "Key": "deregistration_delay.timeout_seconds"
    }
  ]
}
```

- For API details, see [DescribeTargetGroupAttributes](#) in *AWS CLI Command Reference*.

describe-target-groups

The following code example shows how to use `describe-target-groups`.

AWS CLI

Example 1: To describe a target group

The following `describe-target-groups` example displays details for the specified target group.

```
aws elbv2 describe-target-groups \
```

```
--target-group-arns arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

Output:

```
{
  "TargetGroups": [
    {
      "TargetGroupArn": "arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067",
      "TargetGroupName": "my-targets",
      "Protocol": "HTTP",
      "Port": 80,
      "VpcId": "vpc-3ac0fb5f",
      "HealthCheckProtocol": "HTTP",
      "HealthCheckPort": "traffic-port",
      "HealthCheckEnabled": true,
      "HealthCheckIntervalSeconds": 30,
      "HealthCheckTimeoutSeconds": 5,
      "HealthyThresholdCount": 5,
      "UnhealthyThresholdCount": 2,
      "HealthCheckPath": "/",
      "Matcher": {
        "HttpCode": "200"
      },
      "LoadBalancerArns": [
        "arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188"
      ],
      "TargetType": "instance",
      "ProtocolVersion": "HTTP1",
      "IpAddressType": "ipv4"
    }
  ]
}
```

Example 2: To describe all target groups for a load balancer

The following `describe-target-groups` example displays details for all target groups for the specified load balancer. The example uses the `--query` parameter to display only the target group names.

```
aws elbv2 describe-target-groups \
```

```
--load-balancer-arn arn:aws:elasticloadbalancing:us-  
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188 \  
--query TargetGroups[*].TargetGroupName
```

Output:

```
[  
  "my-instance-targets",  
  "my-ip-targets",  
  "my-lambda-target"  
]
```

For more information, see [Target groups](#) in the *Application Load Balancers Guide*.

- For API details, see [DescribeTargetGroups](#) in *AWS CLI Command Reference*.

describe-target-health

The following code example shows how to use `describe-target-health`.

AWS CLI

Example 1: To describe the health of the targets for a target group

The following `describe-target-health` example displays health details for the targets of the specified target group. These targets are healthy.

```
aws elbv2 describe-target-health \  
  --target-group-arn arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

Output:

```
{  
  "TargetHealthDescriptions": [  
    {  
      "HealthCheckPort": "80",  
      "Target": {  
        "Id": "i-ceddc4d",  
        "Port": 80  
      },  
      "TargetHealth": {
```



```

        "State": "healthy"
      }
    },
    {
      "HealthCheckPort": "80",
      "Target": {
        "Id": "i-0f76fade",
        "Port": 80
      },
      "TargetHealth": {
        "State": "healthy"
      }
    }
  ]
}

```

Example 2: To describe the health of a target

The following describe-target-health example displays health details for the specified target. This target is healthy.

```

aws elbv2 describe-target-health \
  --targets Id=i-0f76fade,Port=80 \
  --target-group-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067

```

Output:

```

{
  "TargetHealthDescriptions": [
    {
      "HealthCheckPort": "80",
      "Target": {
        "Id": "i-0f76fade",
        "Port": 80
      },
      "TargetHealth": {
        "State": "healthy"
      }
    }
  ]
}

```

The following example output is for a target whose target group is not specified in an action for a listener. This target can't receive traffic from the load balancer.

```
{
  "TargetHealthDescriptions": [
    {
      "HealthCheckPort": "80",
      "Target": {
        "Id": "i-0f76fade",
        "Port": 80
      },
      "TargetHealth": {
        "State": "unused",
        "Reason": "Target.NotInUse",
        "Description": "Target group is not configured to receive traffic
from the load balancer"
      }
    }
  ]
}
```

The following example output is for a target whose target group was just specified in an action for a listener. The target is still being registered.

```
{
  "TargetHealthDescriptions": [
    {
      "HealthCheckPort": "80",
      "Target": {
        "Id": "i-0f76fade",
        "Port": 80
      },
      "TargetHealth": {
        "State": "initial",
        "Reason": "Elb.RegistrationInProgress",
        "Description": "Target registration is in progress"
      }
    }
  ]
}
```

The following example output is for an unhealthy target.

```
{
  "TargetHealthDescriptions": [
    {
      "HealthCheckPort": "80",
      "Target": {
        "Id": "i-0f76fade",
        "Port": 80
      },
      "TargetHealth": {
        "State": "unhealthy",
        "Reason": "Target.Timeout",
        "Description": "Connection to target timed out"
      }
    }
  ]
}
```

The following example output is for a target that is a Lambda function and health checks are disabled.

```
{
  "TargetHealthDescriptions": [
    {
      "Target": {
        "Id": "arn:aws:lambda:us-west-2:123456789012:function:my-function",
        "AvailabilityZone": "all",
      },
      "TargetHealth": {
        "State": "unavailable",
        "Reason": "Target.HealthCheckDisabled",
        "Description": "Health checks are not enabled for this target"
      }
    }
  ]
}
```

- For API details, see [DescribeTargetHealth](#) in *AWS CLI Command Reference*.

modify-listener

The following code example shows how to use `modify-listener`.

AWS CLI

Example 1: To change the default action to a forward action

The following `modify-listener` example changes the default action (to a **forward** action) for the specified listener.

```
aws elbv2 modify-listener \  
  --listener-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/  
my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2 \  
  --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-new-targets/2453ed029918f21f
```

Output:

```
{  
  "Listeners": [  
    {  
      "Protocol": "HTTP",  
      "DefaultActions": [  
        {  
          "TargetGroupArn": "arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-new-targets/2453ed029918f21f",  
          "Type": "forward"  
        }  
      ],  
      "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-  
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188",  
      "Port": 80,  
      "ListenerArn": "arn:aws:elasticloadbalancing:us-  
west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2"  
    }  
  ]  
}
```

Example 2: To change the default action to a redirect action

The following `modify-listener` example changes the default action to a **redirect** action for the specified listener.

```
aws elbv2 modify-listener \  
  --listener-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/  
my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2 \  
  --default-actions Type=redirect,TargetGroupArn=arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-new-targets/2453ed029918f21f
```

```
--default-actions Type=redirect,TargetGroupArn=arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-new-targets/2453ed029918f21f
```

Output:

```
{
  "Listeners": [
    {
      "Protocol": "HTTP",
      "DefaultActions": [
        {
          "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-new-targets/2453ed029918f21f",
          "Type": "redirect"
        }
      ],
      "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188",
      "Port": 80,
      "ListenerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2"
    }
  ]
}
```

Example 3: To change the server certificate

This example changes the server certificate for the specified HTTPS listener.

```
aws elbv2 modify-listener \
  --listener-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/
my-load-balancer/50dc6c495c0c9188/0467ef3c8400ae65 \
  --certificates CertificateArn=arn:aws:iam::123456789012:server-certificate/my-
new-server-cert
```

Output:

```
{
  "Listeners": [
    {
      "Protocol": "HTTPS",
      "DefaultActions": [
        {
```

```

        "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067",
        "Type": "forward"
    }
],
    "SslPolicy": "ELBSecurityPolicy-2015-05",
    "Certificates": [
        {
            "CertificateArn": "arn:aws:iam::123456789012:server-certificate/
my-new-server-cert"
        }
    ],
    "LoadBalancerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188",
    "Port": 443,
    "ListenerArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener/app/my-load-balancer/50dc6c495c0c9188/0467ef3c8400ae65"
}
]
}

```

- For API details, see [ModifyListener](#) in *AWS CLI Command Reference*.

modify-load-balancer-attributes

The following code example shows how to use `modify-load-balancer-attributes`.

AWS CLI

To enable deletion protection

This example enables deletion protection for the specified load balancer.

Command:

```

aws elbv2 modify-load-balancer-attributes --load-balancer-arn
arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-
balancer/50dc6c495c0c9188 --attributes Key=deletion_protection.enabled,Value=true

```

Output:

```

{
  "Attributes": [

```

```
{
  "Value": "true",
  "Key": "deletion_protection.enabled"
},
{
  "Value": "false",
  "Key": "access_logs.s3.enabled"
},
{
  "Value": "60",
  "Key": "idle_timeout.timeout_seconds"
},
{
  "Value": "",
  "Key": "access_logs.s3.prefix"
},
{
  "Value": "",
  "Key": "access_logs.s3.bucket"
}
]
```

To change the idle timeout

This example changes the idle timeout value for the specified load balancer.

Command:

```
aws elbv2 modify-load-balancer-attributes --load-balancer-arn
arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-
balancer/50dc6c495c0c9188 --attributes Key=idle_timeout.timeout_seconds,Value=30
```

Output:

```
{
  "Attributes": [
    {
      "Value": "30",
      "Key": "idle_timeout.timeout_seconds"
    },
    {
      "Value": "false",
```

```

    "Key": "access_logs.s3.enabled"
  },
  {
    "Value": "",
    "Key": "access_logs.s3.prefix"
  },
  {
    "Value": "true",
    "Key": "deletion_protection.enabled"
  },
  {
    "Value": "",
    "Key": "access_logs.s3.bucket"
  }
]
}

```

To enable access logs

This example enables access logs for the specified load balancer. Note that the S3 bucket must exist in the same region as the load balancer and must have a policy attached that grants access to the Elastic Load Balancing service.

Command:

```

aws elbv2 modify-load-balancer-attributes --load-balancer-arn
arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-
balancer/50dc6c495c0c9188 --attributes Key=access_logs.s3.enabled,Value=true
Key=access_logs.s3.bucket,Value=my-loadbalancer-logs
Key=access_logs.s3.prefix,Value=myapp

```

Output:

```

{
  "Attributes": [
    {
      "Value": "true",
      "Key": "access_logs.s3.enabled"
    },
    {
      "Value": "my-load-balancer-logs",
      "Key": "access_logs.s3.bucket"
    }
  ],
}

```



```

    {
      "Value": "myapp",
      "Key": "access_logs.s3.prefix"
    },
    {
      "Value": "60",
      "Key": "idle_timeout.timeout_seconds"
    },
    {
      "Value": "false",
      "Key": "deletion_protection.enabled"
    }
  ]
}

```

- For API details, see [ModifyLoadBalancerAttributes](#) in *AWS CLI Command Reference*.

modify-rule

The following code example shows how to use `modify-rule`.

AWS CLI

To modify a rule

The following `modify-rule` example updates the actions and conditions for the specified rule.

```

aws elbv2 modify-rule \
  --actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067 \
  --conditions Field=path-pattern,Values='/images/*'
  --rule-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:listener-rule/app/
my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2/9683b2d02a6cabee

```

Output:

```

{
  "Rules": [
    {
      "Priority": "10",
      "Conditions": [
        {
          "Field": "path-pattern",

```

```

        "Values": [
            "/images/*"
        ]
    },
    ],
    "RuleArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener-rule/app/my-load-balancer/50dc6c495c0c9188/
f2f7dc8efc522ab2/9683b2d02a6cabee",
    "IsDefault": false,
    "Actions": [
        {
            "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067",
            "Type": "forward"
        }
    ]
}
]
}

```

- For API details, see [ModifyRule](#) in *AWS CLI Command Reference*.

modify-target-group-attributes

The following code example shows how to use `modify-target-group-attributes`.

AWS CLI

To modify the deregistration delay timeout

This example sets the deregistration delay timeout to the specified value for the specified target group.

Command:

```

aws elbv2 modify-target-group-attributes --target-group-arn
arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-
targets/73e2d6bc24d8a067 --attributes
Key=deregistration_delay.timeout_seconds,Value=600

```

Output:

```
{
```

```
"Attributes": [  
  {  
    "Value": "false",  
    "Key": "stickiness.enabled"  
  },  
  {  
    "Value": "600",  
    "Key": "deregistration_delay.timeout_seconds"  
  },  
  {  
    "Value": "lb_cookie",  
    "Key": "stickiness.type"  
  },  
  {  
    "Value": "86400",  
    "Key": "stickiness.lb_cookie.duration_seconds"  
  }  
]
```

- For API details, see [ModifyTargetGroupAttributes](#) in *AWS CLI Command Reference*.

modify-target-group

The following code example shows how to use `modify-target-group`.

AWS CLI

To modify the health check configuration for a target group

The following `modify-target-group` example changes the configuration of the health checks used to evaluate the health of the targets for the specified target group. Note that due to the way the CLI parses commas, you must surround the range for the `--matcher` option with single quotes instead of double quotes.

```
aws elbv2 modify-target-group \  
  --target-group-arn arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-https-targets/2453ed029918f21f \  
  --health-check-protocol HTTPS \  
  --health-check-port 443 \  
  --matcher HttpCode='200,299'
```

Output:

```
{
  "TargetGroups": [
    {
      "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-https-targets/2453ed029918f21f",
      "TargetGroupName": "my-https-targets",
      "Protocol": "HTTPS",
      "Port": 443,
      "VpcId": "vpc-3ac0fb5f",
      "HealthCheckProtocol": "HTTPS",
      "HealthCheckPort": "443",
      "HealthCheckEnabled": true,
      "HealthCheckIntervalSeconds": 30,
      "HealthCheckTimeoutSeconds": 5,
      "HealthyThresholdCount": 5,
      "UnhealthyThresholdCount": 2,
      "Matcher": {
        "HttpCode": "200,299"
      },
      "LoadBalancerArns": [
        "arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/
app/my-load-balancer/50dc6c495c0c9188"
      ],
      "TargetType": "instance",
      "ProtocolVersion": "HTTP1",
      "IpAddressType": "ipv4"
    }
  ]
}
```

For more information, see [Target groups](#) in the *Application Load Balancers Guide*.

- For API details, see [ModifyTargetGroup](#) in *AWS CLI Command Reference*.

register-targets

The following code example shows how to use `register-targets`.

AWS CLI**Example 1: To register targets with a target group by instance ID**

The following `register-targets` example registers the specified instances with a target group. The target group must have a target type of `instance`.

```
aws elbv2 register-targets \  
  --target-group-arn arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067 \  
  --targets Id=i-1234567890abcdef0 Id=i-0abcdef1234567890
```

Example 2: To register targets with a target group using port overrides

The following `register-targets` example registers the specified instance with a target group using multiple ports. This enables you to register containers on the same instance as targets in the target group.

```
aws elbv2 register-targets \  
  --target-group-arn arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-internal-targets/3bb63f11dfb0faf9 \  
  --targets Id=i-0598c7d356eba48d7,Port=80 Id=i-0598c7d356eba48d7,Port=766
```

Example 3: To register targets with a target group by IP address

The following `register-targets` example registers the specified IP addresses with a target group. The target group must have a target type of `ip`.

```
aws elbv2 register-targets \  
  --target-group-arn arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-tcp-ip-targets/8518e899d173178f \  
  --targets Id=10.0.1.15 Id=10.0.1.23
```

Example 4: To register a Lambda function as a target

The following `register-targets` example registers the specified IP addresses with a target group. The target group must have a target type of `lambda`. You must grant Elastic Load Balancing permission to invoke the Lambda function.

```
aws elbv2 register-targets \  
  --target-group-arn arn:aws:elasticloadbalancing:us-  
west-2:123456789012:targetgroup/my-tcp-ip-targets/8518e899d173178f \  
  --targets Id=arn:aws:lambda:us-west-2:123456789012:function:my-function
```

- For API details, see [RegisterTargets](#) in *AWS CLI Command Reference*.

remove-listener-certificates

The following code example shows how to use `remove-listener-certificates`.

AWS CLI

To remove a certificate from a secure listener

This example removes the specified certificate from the specified secure listener.

Command:

```
aws elbv2 remove-listener-certificates --listener-arn
arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/
my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2 --certificates
CertificateArn=arn:aws:acm:us-west-2:123456789012:certificate/5cc54884-
f4a3-4072-80be-05b9ba72f705
```

- For API details, see [RemoveListenerCertificates](#) in *AWS CLI Command Reference*.

remove-tags

The following code example shows how to use `remove-tags`.

AWS CLI

To remove tags from a load balancer

The following `remove-tags` example removes the project and department tags from the specified load balancer.

```
aws elbv2 remove-tags \
--resource-arns arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188 \
--tag-keys project department
```

- For API details, see [RemoveTags](#) in *AWS CLI Command Reference*.

set-ip-address-type

The following code example shows how to use `set-ip-address-type`.

AWS CLI

To set the address type of a load balancer

This example sets the address type of the specified load balancer to `dualstack`. The load balancer subnets must have associated IPv6 CIDR blocks.

Command:

```
aws elbv2 set-ip-address-type --load-balancer-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188 --ip-address-type dualstack
```

Output:

```
{
  "IpAddressType": "dualstack"
}
```

- For API details, see [SetIpAddressType](#) in *AWS CLI Command Reference*.

set-rule-priorities

The following code example shows how to use `set-rule-priorities`.

AWS CLI

To set the rule priority

This example sets the priority of the specified rule.

Command:

```
aws elbv2 set-rule-priorities --rule-priorities
RuleArn=arn:aws:elasticloadbalancing:us-west-2:123456789012:listener-rule/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2/1291d13826f405c3,Priority=5
```

Output:

```
{
  "Rules": [
    {
```

```

    "Priority": "5",
    "Conditions": [
      {
        "Field": "path-pattern",
        "Values": [
          "/img/*"
        ]
      }
    ],
    "RuleArn": "arn:aws:elasticloadbalancing:us-west-2:123456789012:listener-
rule/app/my-load-balancer/50dc6c495c0c9188/f2f7dc8efc522ab2/1291d13826f405c3",
    "IsDefault": false,
    "Actions": [
      {
        "TargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067",
        "Type": "forward"
      }
    ]
  }
]
}

```

- For API details, see [SetRulePriorities](#) in *AWS CLI Command Reference*.

set-security-groups

The following code example shows how to use `set-security-groups`.

AWS CLI

To associate a security group with a load balancer

This example associates the specified security group with the specified load balancer.

Command:

```
aws elbv2 set-security-groups --load-balancer-arn arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188 --security-
groups sg-5943793c
```

Output:


```
{
  "SecurityGroupIds": [
    "sg-5943793c"
  ]
}
```

- For API details, see [SetSecurityGroups](#) in *AWS CLI Command Reference*.

set-subnets

The following code example shows how to use set-subnets.

AWS CLI

To enable Availability Zones for a load balancer

This example enables the Availability Zone for the specified subnet for the specified load balancer.

Command:

```
aws elbv2 set-subnets --load-balancer-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-load-balancer/50dc6c495c0c9188 --subnets subnet-8360a9e7 subnet-b7d581c0
```

Output:

```
{
  "AvailabilityZones": [
    {
      "SubnetId": "subnet-8360a9e7",
      "ZoneName": "us-west-2a"
    },
    {
      "SubnetId": "subnet-b7d581c0",
      "ZoneName": "us-west-2b"
    }
  ]
}
```

- For API details, see [SetSubnets](#) in *AWS CLI Command Reference*.

Elastic Transcoder examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Elastic Transcoder.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

cancel-job

The following code example shows how to use `cancel-job`.

AWS CLI

To cancel a job for ElasticTranscoder

This cancels the specified job for ElasticTranscoder.

Command:

```
aws elastictranscoder cancel-job --id 3333333333333-abcde3
```

- For API details, see [CancelJob](#) in *AWS CLI Command Reference*.

create-job

The following code example shows how to use `create-job`.

AWS CLI

To create a job for ElasticTranscoder

The following `create-job` example creates a job for ElasticTranscoder.

```
aws elastictranscoder create-job \  
  --pipeline-id 111111111111-abcde1 \  
  --inputs file://inputs.json \  
  --outputs file://outputs.json \  
  --output-key-prefix "recipes/" \  
  --user-metadata file://user-metadata.json
```

Contents of `inputs.json`:

```
[{  
  "Key": "ETS_example_file.mp4",  
  "FrameRate": "auto",  
  "Resolution": "auto",  
  "AspectRatio": "auto",  
  "Interlaced": "auto",  
  "Container": "mp4"  
}]
```

Contents of `outputs.json`:

```
[  
  {  
    "Key": "webm/ETS_example_file-kindlefirehd.webm",  
    "Rotate": "0",  
    "PresetId": "1351620000001-100250"  
  }  
]
```

Contents of `user-metadata.json`:

```
{  
  "Food type": "Italian",  
  "Cook book": "recipe notebook"  
}
```

Output:

```
{
  "Job": {
    "Status": "Submitted",
    "Inputs": [
      {
        "Container": "mp4",
        "FrameRate": "auto",
        "Key": "ETS_example_file.mp4",
        "AspectRatio": "auto",
        "Resolution": "auto",
        "Interlaced": "auto"
      }
    ],
    "Playlists": [],
    "Outputs": [
      {
        "Status": "Submitted",
        "Rotate": "0",
        "PresetId": "1351620000001-100250",
        "Watermarks": [],
        "Key": "webm/ETS_example_file-kindlefirehd.webm",
        "Id": "1"
      }
    ],
    "PipelineId": "3333333333333-abcde3",
    "OutputKeyPrefix": "recipes/",
    "UserMetadata": {
      "Cook book": "recipe notebook",
      "Food type": "Italian"
    },
    "Output": {
      "Status": "Submitted",
      "Rotate": "0",
      "PresetId": "1351620000001-100250",
      "Watermarks": [],
      "Key": "webm/ETS_example_file-kindlefirehd.webm",
      "Id": "1"
    },
    "Timing": {
      "SubmitTimeMillis": 1533838012298
    },
    "Input": {
```

```

        "Container": "mp4",
        "FrameRate": "auto",
        "Key": "ETS_example_file.mp4",
        "AspectRatio": "auto",
        "Resolution": "auto",
        "Interlaced": "auto"
    },
    "Id": "1533838012294-example",
    "Arn": "arn:aws:elastictranscoder:us-west-2:123456789012:job/1533838012294-
example"
    }
}

```

- For API details, see [CreateJob](#) in *AWS CLI Command Reference*.

create-pipeline

The following code example shows how to use create-pipeline.

AWS CLI

To create a pipeline for ElasticTranscoder

The following create-pipeline example creates a pipeline for ElasticTranscoder.

```

aws elastictranscoder create-pipeline \
  --name Default \
  --input-bucket salesoffice.example.com-source \
  --role arn:aws:iam::123456789012:role/Elastic_Transcoder_Default_Role \
  --notifications Progressing="",Completed="",Warning="",Error=arn:aws:sns:us-
east-1:111222333444:ETS_Errors \
  --content-config file://content-config.json \
  --thumbnail-config file://thumbnail-config.json

```

Contents of content-config.json:

```

{
  "Bucket":"salesoffice.example.com-public-promos",
  "Permissions":[
    {
      "GranteeType":"Email",
      "Grantee":"marketing-promos@example.com",

```

```

        "Access":[
            "FullControl"
        ]
    },
    "StorageClass":"Standard"
}

```

Contents of thumbnail-config.json:

```

{
  "Bucket":"salesoffice.example.com-public-promos-thumbnails",
  "Permissions":[
    {
      "GranteeType":"Email",
      "Grantee":"marketing-promos@example.com",
      "Access":[
        "FullControl"
      ]
    }
  ],
  "StorageClass":"ReducedRedundancy"
}

```

Output:

```

{
  "Pipeline": {
    "Status": "Active",
    "ContentConfig": {
      "Bucket": "salesoffice.example.com-public-promos",
      "StorageClass": "Standard",
      "Permissions": [
        {
          "Access": [
            "FullControl"
          ],
          "Grantee": "marketing-promos@example.com",
          "GranteeType": "Email"
        }
      ]
    }
  },
  "Name": "Default",
}

```

```

    "ThumbnailConfig": {
      "Bucket": "salesoffice.example.com-public-promos-thumbnails",
      "StorageClass": "ReducedRedundancy",
      "Permissions": [
        {
          "Access": [
            "FullControl"
          ],
          "Grantee": "marketing-promos@example.com",
          "GranteeType": "Email"
        }
      ]
    },
    "Notifications": {
      "Completed": "",
      "Warning": "",
      "Progressing": "",
      "Error": "arn:aws:sns:us-east-1:123456789012:ETS_Errors"
    },
    "Role": "arn:aws:iam::123456789012:role/Elastic_Transcoder_Default_Role",
    "InputBucket": "salesoffice.example.com-source",
    "Id": "1533765810590-example",
    "Arn": "arn:aws:elastictranscoder:us-
west-2:123456789012:pipeline/1533765810590-example"
  },
  "Warnings": [
    {
      "Message": "The SNS notification topic for Error events and the pipeline
are in different regions, which increases processing time for jobs in the pipeline
and can incur additional charges. To decrease processing time and prevent cross-
regional charges, use the same region for the SNS notification topic and the
pipeline.",
      "Code": "6006"
    }
  ]
}

```

- For API details, see [CreatePipeline](#) in *AWS CLI Command Reference*.

create-preset

The following code example shows how to use create-preset.

AWS CLI

To create a preset for ElasticTranscoder

The following create-preset example creates a preset for ElasticTranscoder.

```
aws elastictranscoder create-preset \  
  --name DefaultPreset \  
  --description "Use for published videos" \  
  --container mp4 \  
  --video file://video.json \  
  --audio file://audio.json \  
  --thumbnails file://thumbnails.json
```

Contents of video.json:

```
{  
  "Codec": "H.264",  
  "CodecOptions": {  
    "Profile": "main",  
    "Level": "2.2",  
    "MaxReferenceFrames": "3",  
    "MaxBitRate": "",  
    "BufferSize": "",  
    "InterlacedMode": "Progressive",  
    "ColorSpaceConversionMode": "None"  
  },  
  "KeyframesMaxDist": "240",  
  "FixedGOP": "false",  
  "BitRate": "1600",  
  "FrameRate": "auto",  
  "MaxFrameRate": "30",  
  "MaxWidth": "auto",  
  "MaxHeight": "auto",  
  "SizingPolicy": "Fit",  
  "PaddingPolicy": "Pad",  
  "DisplayAspectRatio": "auto",  
  "Watermarks": [  
    {  
      "Id": "company logo",  
      "MaxWidth": "20%",  
      "MaxHeight": "20%",  
      "SizingPolicy": "ShrinkToFit",  
    }  
  ]  
}
```



```
        "HorizontalAlign": "Right",
        "HorizontalOffset": "10px",
        "VerticalAlign": "Bottom",
        "VerticalOffset": "10px",
        "Opacity": "55.5",
        "Target": "Content"
    }
]
```

Contents of audio.json:

```
{
  "Codec": "AAC",
  "CodecOptions": {
    "Profile": "AAC-LC"
  },
  "SampleRate": "44100",
  "BitRate": "96",
  "Channels": "2"
}
```

Contents of thumbnails.json:

```
{
  "Format": "png",
  "Interval": "120",
  "MaxWidth": "auto",
  "MaxHeight": "auto",
  "SizingPolicy": "Fit",
  "PaddingPolicy": "Pad"
}
```

Output:

```
{
  "Preset": {
    "Thumbnails": {
      "SizingPolicy": "Fit",
      "MaxWidth": "auto",
      "Format": "png",
      "PaddingPolicy": "Pad",

```

```
    "Interval": "120",
    "MaxHeight": "auto"
  },
  "Container": "mp4",
  "Description": "Use for published videos",
  "Video": {
    "SizingPolicy": "Fit",
    "MaxWidth": "auto",
    "PaddingPolicy": "Pad",
    "MaxFrameRate": "30",
    "FrameRate": "auto",
    "MaxHeight": "auto",
    "KeyframesMaxDist": "240",
    "FixedGOP": "false",
    "Codec": "H.264",
    "Watermarks": [
      {
        "SizingPolicy": "ShrinkToFit",
        "VerticalOffset": "10px",
        "VerticalAlign": "Bottom",
        "Target": "Content",
        "MaxWidth": "20%",
        "MaxHeight": "20%",
        "HorizontalAlign": "Right",
        "HorizontalOffset": "10px",
        "Opacity": "55.5",
        "Id": "company logo"
      }
    ],
    "CodecOptions": {
      "Profile": "main",
      "MaxBitRate": "32",
      "InterlacedMode": "Progressive",
      "Level": "2.2",
      "ColorSpaceConversionMode": "None",
      "MaxReferenceFrames": "3",
      "BufferSize": "5"
    },
    "BitRate": "1600",
    "DisplayAspectRatio": "auto"
  },
  "Audio": {
    "Channels": "2",
    "CodecOptions": {
```

```
        "Profile": "AAC-LC"
      },
      "SampleRate": "44100",
      "Codec": "AAC",
      "BitRate": "96"
    },
    "Type": "Custom",
    "Id": "1533765290724-example"
    "Arn": "arn:aws:elastictranscoder:us-
west-2:123456789012:preset/1533765290724-example",
    "Name": "DefaultPreset"
  },
  "Warning": ""
}
```

- For API details, see [CreatePreset](#) in *AWS CLI Command Reference*.

delete-pipeline

The following code example shows how to use delete-pipeline.

AWS CLI

To delete the specified ElasticTranscoder pipeline

This deletes the specified ElasticTranscoder pipeline.

Command:

```
aws elastictranscoder delete-pipeline --id 111111111111-abcde1
```

Output:

```
{
  "Success": "true"
}
```

- For API details, see [DeletePipeline](#) in *AWS CLI Command Reference*.

delete-preset

The following code example shows how to use delete-preset.

AWS CLI

To delete the specified ElasticTranscoder preset

This deletes the specified ElasticTranscoder preset.

Command:

```
aws elastictranscoder delete-preset --id 555555555555-abcde5
```

- For API details, see [DeletePreset](#) in *AWS CLI Command Reference*.

list-jobs-by-pipeline

The following code example shows how to use `list-jobs-by-pipeline`.

AWS CLI

To retrieve a list of ElasticTranscoder jobs in the specified pipeline

This example retrieves a list of ElasticTranscoder jobs in the specified pipeline.

Command:

```
aws elastictranscoder list-jobs-by-pipeline --pipeline-id 111111111111-abcde1
```

Output:

```
{
  "Jobs": []
}
```

- For API details, see [ListJobsByPipeline](#) in *AWS CLI Command Reference*.

list-jobs-by-status

The following code example shows how to use `list-jobs-by-status`.

AWS CLI

To retrieve a list of ElasticTranscoder jobs with a status of Complete

This example retrieves a list of ElasticTranscoder jobs with a status of Complete.

Command:

```
aws elastictranscoder list-jobs-by-status --status Complete
```

Output:

```
{
  "Jobs": []
}
```

- For API details, see [ListJobsByStatus](#) in *AWS CLI Command Reference*.

list-pipelines

The following code example shows how to use `list-pipelines`.

AWS CLI

To retrieve a list of ElasticTranscoder pipelines

This example retrieves a list of ElasticTranscoder pipelines.

Command:

```
aws elastictranscoder list-pipelines
```

Output:

```
{
  "Pipelines": [
    {
      "Status": "Active",
      "ContentConfig": {
        "Bucket": "ets-example",
        "Permissions": []
      },
      "Name": "example-pipeline",
      "ThumbnailConfig": {
        "Bucket": "ets-example",
        "Permissions": []
      }
    }
  ]
}
```

```
    },
    "Notifications": {
      "Completed": "arn:aws:sns:us-west-2:123456789012:ets_example",
      "Warning": "",
      "Progressing": "",
      "Error": ""
    },
    },
    "Role": "arn:aws:iam::123456789012:role/Elastic_Transcoder_Default_Role",
    "InputBucket": "ets-example",
    "OutputBucket": "ets-example",
    "Id": "3333333333333-abcde3",
    "Arn": "arn:aws:elastictranscoder:us-
west-2:123456789012:pipeline/3333333333333-abcde3"
  },
  {
    "Status": "Paused",
    "ContentConfig": {
      "Bucket": "ets-example",
      "Permissions": []
    },
    },
    "Name": "example-php-test",
    "ThumbnailConfig": {
      "Bucket": "ets-example",
      "Permissions": []
    },
    },
    "Notifications": {
      "Completed": "",
      "Warning": "",
      "Progressing": "",
      "Error": ""
    },
    },
    "Role": "arn:aws:iam::123456789012:role/Elastic_Transcoder_Default_Role",
    "InputBucket": "ets-example",
    "OutputBucket": "ets-example",
    "Id": "3333333333333-abcde2",
    "Arn": "arn:aws:elastictranscoder:us-
west-2:123456789012:pipeline/3333333333333-abcde2"
  },
  {
    "Status": "Active",
    "ContentConfig": {
      "Bucket": "ets-west-output",
      "Permissions": []
    },
    },
  },
```

```

    "Name": "pipeline-west",
    "ThumbnailConfig": {
      "Bucket": "ets-west-output",
      "Permissions": []
    },
    "Notifications": {
      "Completed": "arn:aws:sns:us-west-2:123456789012:ets-notifications",
      "Warning": "",
      "Progressing": "",
      "Error": ""
    },
    "Role": "arn:aws:iam::123456789012:role/Elastic_Transcoder_Default_Role",
    "InputBucket": "ets-west-input",
    "OutputBucket": "ets-west-output",
    "Id": "3333333333333-abcde1",
    "Arn": "arn:aws:elastictranscoder:us-
west-2:123456789012:pipeline/3333333333333-abcde1"
  }
]
}

```

- For API details, see [ListPipelines](#) in *AWS CLI Command Reference*.

list-presets

The following code example shows how to use `list-presets`.

AWS CLI

To retrieve a list of ElasticTranscoder presets

This example retrieves a list of ElasticTranscoder presets.

Command:

```
aws elastictranscoder list-presets --max-items 2
```

Output:

```

{
  "Presets": [
    {
      "Container": "mp4",

```

```
"Name": "KindleFireHD-preset",
"Video": {
  "Resolution": "1280x720",
  "FrameRate": "30",
  "KeyframesMaxDist": "90",
  "FixedGOP": "false",
  "Codec": "H.264",
  "Watermarks": [],
  "CodecOptions": {
    "Profile": "main",
    "MaxReferenceFrames": "3",
    "ColorSpaceConversionMode": "None",
    "InterlacedMode": "Progressive",
    "Level": "4"
  },
  "AspectRatio": "16:9",
  "BitRate": "2200"
},
"Audio": {
  "Channels": "2",
  "CodecOptions": {
    "Profile": "AAC-LC"
  },
  "SampleRate": "48000",
  "Codec": "AAC",
  "BitRate": "160"
},
"Type": "Custom",
"Id": "333333333333-abcde2",
"Arn": "arn:aws:elastictranscoder:us-
west-2:123456789012:preset/333333333333-abcde2",
"Thumbnails": {
  "AspectRatio": "16:9",
  "Interval": "60",
  "Resolution": "192x108",
  "Format": "png"
}
},
{
  "Thumbnails": {
    "AspectRatio": "16:9",
    "Interval": "60",
    "Resolution": "192x108",
    "Format": "png"
  }
}
```



```

    },
    "Container": "mp4",
    "Description": "Custom preset for transcoding jobs",
    "Video": {
      "Resolution": "1280x720",
      "FrameRate": "30",
      "KeyframesMaxDist": "90",
      "FixedGOP": "false",
      "Codec": "H.264",
      "Watermarks": [],
      "CodecOptions": {
        "Profile": "main",
        "MaxReferenceFrames": "3",
        "ColorSpaceConversionMode": "None",
        "InterlacedMode": "Progressive",
        "Level": "3.1"
      },
      "AspectRatio": "16:9",
      "BitRate": "2200"
    },
    "Audio": {
      "Channels": "2",
      "CodecOptions": {
        "Profile": "AAC-LC"
      },
      "SampleRate": "44100",
      "Codec": "AAC",
      "BitRate": "160"
    },
    "Type": "Custom",
    "Id": "333333333333-abcde3",
    "Arn": "arn:aws:elastictranscoder:us-
west-2:123456789012:preset/333333333333-abcde3",
    "Name": "Roman's Preset"
  }
],
"NextToken": "eyJQYWdlVG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAyfQ=="
}

```

- For API details, see [ListPresets](#) in *AWS CLI Command Reference*.

read-job

The following code example shows how to use `read-job`.

AWS CLI

To retrieve an ElasticTranscoder job

This example retrieves the specified ElasticTranscoder job.

Command:

```
aws elastictranscoder read-job --id 1533838012294-example
```

Output:

```
{
  "Job": {
    "Status": "Progressing",
    "Inputs": [
      {
        "Container": "mp4",
        "FrameRate": "auto",
        "Key": "ETS_example_file.mp4",
        "AspectRatio": "auto",
        "Resolution": "auto",
        "Interlaced": "auto"
      }
    ],
    "Playlists": [],
    "Outputs": [
      {
        "Status": "Progressing",
        "Rotate": "0",
        "PresetId": "1351620000001-100250",
        "Watermarks": [],
        "Key": "webm/ETS_example_file-kindlefirehd.webm",
        "Id": "1"
      }
    ],
    "PipelineId": "3333333333333-abcde3",
    "OutputKeyPrefix": "recipes/",
    "UserMetadata": {
      "Cook book": "recipe notebook",
```

```

    "Food type": "Italian"
  },
  "Output": {
    "Status": "Progressing",
    "Rotate": "0",
    "PresetId": "1351620000001-100250",
    "Watermarks": [],
    "Key": "webm/ETS_example_file-kindlefirehd.webm",
    "Id": "1"
  },
  "Timing": {
    "SubmitTimeMillis": 1533838012298,
    "StartTimeMillis": 1533838013786
  },
  "Input": {
    "Container": "mp4",
    "FrameRate": "auto",
    "Key": "ETS_example_file.mp4",
    "AspectRatio": "auto",
    "Resolution": "auto",
    "Interlaced": "auto"
  },
  "Id": "1533838012294-example",
  "Arn": "arn:aws:elastictranscoder:us-west-2:123456789012:job/1533838012294-
example"
}
}

```

- For API details, see [ReadJob](#) in *AWS CLI Command Reference*.

read-pipeline

The following code example shows how to use read-pipeline.

AWS CLI

To retrieve an ElasticTranscoder pipeline

This example retrieves the specified ElasticTranscoder pipeline.

Command:

```
aws elastictranscoder read-pipeline --id 333333333333-abcde3
```

Output:

```
{
  "Pipeline": {
    "Status": "Active",
    "ContentConfig": {
      "Bucket": "ets-example",
      "StorageClass": "Standard",
      "Permissions": [
        {
          "Access": [
            "FullControl"
          ],
          "Grantee": "marketing-promos@example.com",
          "GranteeType": "Email"
        }
      ]
    },
    "Name": "Default",
    "ThumbnailConfig": {
      "Bucket": "ets-example",
      "StorageClass": "ReducedRedundancy",
      "Permissions": [
        {
          "Access": [
            "FullControl"
          ],
          "Grantee": "marketing-promos@example.com",
          "GranteeType": "Email"
        }
      ]
    },
    "Notifications": {
      "Completed": "",
      "Warning": "",
      "Progressing": "",
      "Error": "arn:aws:sns:us-east-1:123456789012:ETS_Errors"
    },
    "Role": "arn:aws:iam::123456789012:role/Elastic_Transcoder_Default_Role",
    "InputBucket": "ets-example",
    "Id": "3333333333333-abcde3",
    "Arn": "arn:aws:elastictranscoder:us-west-2:123456789012:pipeline/3333333333333-abcde3"
  },
}
```

```
"Warnings": [  
  {  
    "Message": "The SNS notification topic for Error events and the pipeline  
are in different regions, which increases processing time for jobs in the pipeline  
and can incur additional charges. To decrease processing time and prevent cross-  
regional charges, use the same region for the SNS notification topic and the  
pipeline.",  
    "Code": "6006"  
  }  
]
```

- For API details, see [ReadPipeline](#) in *AWS CLI Command Reference*.

read-preset

The following code example shows how to use read-preset.

AWS CLI

To retrieve an ElasticTranscoder preset

This example retrieves the specified ElasticTranscoder preset.

Command:

```
aws elastictranscoder read-preset --id 1351620000001-500020
```

Output:

```
{  
  "Preset": {  
    "Thumbnails": {  
      "SizingPolicy": "ShrinkToFit",  
      "MaxWidth": "192",  
      "Format": "png",  
      "PaddingPolicy": "NoPad",  
      "Interval": "300",  
      "MaxHeight": "108"  
    },  
    "Container": "fmp4",  
    "Description": "System preset: MPEG-Dash Video - 4.8M",  
    "Video": {
```

```
"SizingPolicy": "ShrinkToFit",
"MaxWidth": "1280",
"PaddingPolicy": "NoPad",
"FrameRate": "30",
"MaxHeight": "720",
"KeyframesMaxDist": "60",
"FixedGOP": "true",
"Codec": "H.264",
"Watermarks": [
  {
    "SizingPolicy": "ShrinkToFit",
    "VerticalOffset": "10%",
    "VerticalAlign": "Top",
    "Target": "Content",
    "MaxWidth": "10%",
    "MaxHeight": "10%",
    "HorizontalAlign": "Left",
    "HorizontalOffset": "10%",
    "Opacity": "100",
    "Id": "TopLeft"
  },
  {
    "SizingPolicy": "ShrinkToFit",
    "VerticalOffset": "10%",
    "VerticalAlign": "Top",
    "Target": "Content",
    "MaxWidth": "10%",
    "MaxHeight": "10%",
    "HorizontalAlign": "Right",
    "HorizontalOffset": "10%",
    "Opacity": "100",
    "Id": "TopRight"
  },
  {
    "SizingPolicy": "ShrinkToFit",
    "VerticalOffset": "10%",
    "VerticalAlign": "Bottom",
    "Target": "Content",
    "MaxWidth": "10%",
    "MaxHeight": "10%",
    "HorizontalAlign": "Left",
    "HorizontalOffset": "10%",
    "Opacity": "100",
    "Id": "BottomLeft"
  }
]
```

```

    },
    {
      "SizingPolicy": "ShrinkToFit",
      "VerticalOffset": "10%",
      "VerticalAlign": "Bottom",
      "Target": "Content",
      "MaxWidth": "10%",
      "MaxHeight": "10%",
      "HorizontalAlign": "Right",
      "HorizontalOffset": "10%",
      "Opacity": "100",
      "Id": "BottomRight"
    }
  ],
  "CodecOptions": {
    "Profile": "main",
    "MaxBitRate": "4800",
    "InterlacedMode": "Progressive",
    "Level": "3.1",
    "ColorSpaceConversionMode": "None",
    "MaxReferenceFrames": "3",
    "BufferSize": "9600"
  },
  "BitRate": "4800",
  "DisplayAspectRatio": "auto"
},
"Type": "System",
"Id": "1351620000001-500020",
"Arn": "arn:aws:elastictranscoder:us-
west-2:123456789012:preset/1351620000001-500020",
"Name": "System preset: MPEG-Dash Video - 4.8M"
}
}

```

- For API details, see [ReadPreset](#) in *AWS CLI Command Reference*.

update-pipeline-notifications

The following code example shows how to use update-pipeline-notifications.

AWS CLI

To update the notifications of an ElasticTranscoder pipeline

This example updates the notifications of the specified ElasticTranscoder pipeline.

Command:

```
aws elastictranscoder update-pipeline-notifications --id 111111111111-
abcde1 --notifications Progressing=arn:aws:sns:us-west-2:0123456789012:my-
topic,Completed=arn:aws:sns:us-west-2:0123456789012:my-topic,Warning=arn:aws:sns:us-
west-2:0123456789012:my-topic,Error=arn:aws:sns:us-east-1:111222333444:ETS_Errors
```

Output:

```
{
  "Pipeline": {
    "Status": "Active",
    "ContentConfig": {
      "Bucket": "ets-example",
      "StorageClass": "Standard",
      "Permissions": [
        {
          "Access": [
            "FullControl"
          ],
          "Grantee": "marketing-promos@example.com",
          "GranteeType": "Email"
        }
      ]
    },
    "Name": "Default",
    "ThumbnailConfig": {
      "Bucket": "ets-example",
      "StorageClass": "ReducedRedundancy",
      "Permissions": [
        {
          "Access": [
            "FullControl"
          ],
          "Grantee": "marketing-promos@example.com",
          "GranteeType": "Email"
        }
      ]
    }
  },
  "Notifications": {
    "Completed": "arn:aws:sns:us-west-2:0123456789012:my-topic",
```



```

        "Warning": "arn:aws:sns:us-west-2:0123456789012:my-topic",
        "Progressing": "arn:aws:sns:us-west-2:0123456789012:my-topic",
        "Error": "arn:aws:sns:us-east-1:111222333444:ETS_Errors"
    },
    "Role": "arn:aws:iam::123456789012:role/Elastic_Transcoder_Default_Role",
    "InputBucket": "ets-example",
    "Id": "111111111111-abcde1",
    "Arn": "arn:aws:elastictranscoder:us-
west-2:123456789012:pipeline/111111111111-abcde1"
    }
}

```

- For API details, see [UpdatePipelineNotifications](#) in *AWS CLI Command Reference*.

update-pipeline-status

The following code example shows how to use `update-pipeline-status`.

AWS CLI

To update the status of an ElasticTranscoder pipeline

This example updates the status of the specified ElasticTranscoder pipeline.

Command:

```
aws elastictranscoder update-pipeline-status --id 111111111111-abcde1 --status Paused
```

Output:

```

{
  "Pipeline": {
    "Status": "Paused",
    "ContentConfig": {
      "Bucket": "ets-example",
      "StorageClass": "Standard",
      "Permissions": [
        {
          "Access": [
            "FullControl"
          ]
        }
      ]
    }
  }
}

```

```

        ],
        "Grantee": "marketing-promos@example.com",
        "GranteeType": "Email"
    }
]
},
"Name": "Default",
"ThumbnailConfig": {
    "Bucket": "ets-example",
    "StorageClass": "ReducedRedundancy",
    "Permissions": [
        {
            "Access": [
                "FullControl"
            ],
            "Grantee": "marketing-promos@example.com",
            "GranteeType": "Email"
        }
    ]
},
"Notifications": {
    "Completed": "",
    "Warning": "",
    "Progressing": "",
    "Error": "arn:aws:sns:us-east-1:803981987763:ETS_Errors"
},
"Role": "arn:aws:iam::123456789012:role/Elastic_Transcoder_Default_Role",
"InputBucket": "ets-example",
"Id": "111111111111-abcde1",
"Arn": "arn:aws:elastictranscoder:us-
west-2:123456789012:pipeline/111111111111-abcde1"
}
}

```

- For API details, see [UpdatePipelineStatus](#) in *AWS CLI Command Reference*.

update-pipeline

The following code example shows how to use update-pipeline.

AWS CLI

To update an ElasticTranscoder pipeline

The following `update-pipeline` example updates the specified ElasticTranscoder pipeline.

```
aws elastictranscoder update-pipeline \  
  --id 111111111111-abcde1 \  
  --name DefaultExample \  
  --input-bucket salesoffice.example.com-source \  
  --role arn:aws:iam::123456789012:role/Elastic_Transcoder_Default_Role \  
  --notifications Progressing="",Completed="",Warning="",Error=arn:aws:sns:us-  
east-1:111222333444:ETS_Errors \  
  --content-config file://content-config.json \  
  --thumbnail-config file://thumbnail-config.json
```

Contents of `content-config.json`:

```
{  
  "Bucket": "salesoffice.example.com-public-promos",  
  "Permissions": [  
    {  
      "GranteeType": "Email",  
      "Grantee": "marketing-promos@example.com",  
      "Access": [  
        "FullControl"  
      ]  
    }  
  ],  
  "StorageClass": "Standard"  
}
```

Contents of `thumbnail-config.json`:

```
{  
  "Bucket": "salesoffice.example.com-public-promos-thumbnails",  
  "Permissions": [  
    {  
      "GranteeType": "Email",  
      "Grantee": "marketing-promos@example.com",  
      "Access": [  
        "FullControl"  
      ]  
    }  
  ],  
  "StorageClass": "ReducedRedundancy"
```

```
}
```

Output:

```
{
  "Pipeline": {
    "Status": "Active",
    "ContentConfig": {
      "Bucket": "ets-example",
      "StorageClass": "Standard",
      "Permissions": [
        {
          "Access": [
            "FullControl"
          ],
          "Grantee": "marketing-promos@example.com",
          "GranteeType": "Email"
        }
      ]
    },
    "Name": "DefaultExample",
    "ThumbnailConfig": {
      "Bucket": "ets-example",
      "StorageClass": "ReducedRedundancy",
      "Permissions": [
        {
          "Access": [
            "FullControl"
          ],
          "Grantee": "marketing-promos@example.com",
          "GranteeType": "Email"
        }
      ]
    },
    "Notifications": {
      "Completed": "",
      "Warning": "",
      "Progressing": "",
      "Error": "arn:aws:sns:us-east-1:1112223333444:ETS_Errors"
    },
    "Role": "arn:aws:iam::123456789012:role/Elastic_Transcoder_Default_Role",
    "InputBucket": "ets-example",
    "Id": "33333333333333-abcde3",
  }
}
```

```
    "Arn": "arn:aws:elastictranscoder:us-  
west-2:123456789012:pipeline/333333333333-abcde3"  
  },  
  "Warnings": [  
    {  
      "Message": "The SNS notification topic for Error events and the pipeline  
are in different regions, which increases processing time for jobs in the pipeline  
and can incur additional charges. To decrease processing time and prevent cross-  
regional charges, use the same region for the SNS notification topic and the  
pipeline.",  
      "Code": "6006"  
    }  
  ]  
}
```

- For API details, see [UpdatePipeline](#) in *AWS CLI Command Reference*.

ElastiCache examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with ElastiCache.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-tags-to-resource

The following code example shows how to use `add-tags-to-resource`.

AWS CLI

To add tags to a resource

The following `add-tags-to-resource` example adds up to 10 tags, key-value pairs, to a cluster or snapshot resource.

```
aws elasticache add-tags-to-resource \  
  --resource-name "arn:aws:elasticache:us-east-1:1234567890:cluster:my-mem-  
cluster" \  
  --tags '{"20150202":15, "ElastiCache":"Service"}'
```

Output:

```
{  
  "TagList": [  
    {  
      "Value": "20150202",  
      "Key": "APIVersion"  
    },  
    {  
      "Value": "ElastiCache",  
      "Key": "Service"  
    }  
  ]  
}
```

For more information, see [Monitoring Costs with Cost Allocation Tags](#) in the *ElastiCache User Guide*.

- For API details, see [AddTagsToResource](#) in *AWS CLI Command Reference*.

authorize-cache-security-group-ingress

The following code example shows how to use `authorize-cache-security-group-ingress`.

AWS CLI

To authorize cache security group for ingress

The following `authorize-cache-security-group-ingress` example allows network ingress to a cache security group.

```
aws elasticache authorize-cache-security-group-ingress \  
  --cache-security-group-name "my-sec-grp" \  
  --ec2-security-group-name "my-ec2-sec-grp" \  
  --ec2-security-group-owner-id "1234567890"
```

The command produces no output.

For more information, see [Self-Service Updates in Amazon ElastiCache](#) in the *Elasticache User Guide*.

- For API details, see [AuthorizeCacheSecurityGroupIngress](#) in *AWS CLI Command Reference*.

batch-apply-update-action

The following code example shows how to use batch-apply-update-action.

AWS CLI

To apply a service update

The following batch-apply-update-action example applies a service update to a Redis cluster.

```
aws elasticache batch-apply-update-action \  
  --service-update-name elc-xxxxx406-xxx \  
  --replication-group-ids test-cluster
```

Output:

```
{  
  "ProcessedUpdateActions": [  
    {  
      "ReplicationGroupId": "pat-cluster",  
      "ServiceUpdateName": "elc-xxxxx406-xxx",  
      "UpdateActionStatus": "waiting-to-start"  
    }  
  ],  
  "UnprocessedUpdateActions": []  
}
```

For more information, see [Self-Service Updates in Amazon ElastiCache](#) in the *Elasticache User Guide*.

- For API details, see [BatchApplyUpdateAction](#) in *AWS CLI Command Reference*.

batch-stop-update-action

The following code example shows how to use batch-stop-update-action.

AWS CLI

To stop a service update

The following batch-stop-update-action example applies a service update to a Redis cluster.

```
aws elasticache batch-stop-update-action \  
  --service-update-name elc-xxxxx406-xxx \  
  --replication-group-ids test-cluster
```

Output:

```
{  
  "ProcessedUpdateActions": [  
    {  
      "ReplicationGroupId": "pat-cluster",  
      "ServiceUpdateName": "elc-xxxxx406-xxx",  
      "UpdateActionStatus": "stopping"  
    }  
  ],  
  "UnprocessedUpdateActions": []  
}
```

For more information, see [Self-Service Updates in Amazon ElastiCache](#) in the *Elasticache User Guide*.

- For API details, see [BatchStopUpdateAction](#) in *AWS CLI Command Reference*.

copy-snapshot

The following code example shows how to use copy-snapshot.

AWS CLI

To copy a snapshot

The following `copy-snapshot` example makes a copy of an existing snapshot.

```
aws elasticache copy-snapshot \  
  --source-snapshot-name "my-snapshot" \  
  --target-snapshot-name "my-snapshot-copy"
```

Output:

```
{  
  "Snapshot":{  
    "Engine": "redis",  
    "CacheParameterGroupName": "default.redis3.2",  
    "VpcId": "vpc-3820329f3",  
    "CacheClusterId": "my-redis4",  
    "SnapshotRetentionLimit": 7,  
    "NumCacheNodes": 1,  
    "SnapshotName": "my-snapshot-copy",  
    "CacheClusterCreateTime": "2016-12-21T22:24:04.955Z",  
    "AutoMinorVersionUpgrade": true,  
    "PreferredAvailabilityZone": "us-east-1c",  
    "SnapshotStatus": "creating",  
    "SnapshotSource": "manual",  
    "SnapshotWindow": "07:00-08:00",  
    "EngineVersion": "3.2.4",  
    "NodeSnapshots": [  
      {  
        "CacheSize": "3 MB",  
        "SnapshotCreateTime": "2016-12-28T07:00:52Z",  
        "CacheNodeId": "0001",  
        "CacheNodeCreateTime": "2016-12-21T22:24:04.955Z"  
      }  
    ],  
    "CacheSubnetGroupName": "default",  
    "Port": 6379,  
    "PreferredMaintenanceWindow": "tue:09:30-tue:10:30",  
    "CacheNodeType": "cache.m3.large"  
  }  
}
```

For more information, see [Exporting a Backup](#) in the *Elasticache User Guide*.

- For API details, see [CopySnapshot](#) in *AWS CLI Command Reference*.

create-cache-cluster

The following code example shows how to use `create-cache-cluster`.

AWS CLI

To create a cache cluster

The following `create-cache-cluster` example creates a cache cluster using the Redis engine.

```
aws elasticache create-cache-cluster \  
  --cache-cluster-id "cluster-test" \  
  --engine redis \  
  --cache-node-type cache.m5.large \  
  --num-cache-nodes 1
```

Output:

```
{  
  "CacheCluster": {  
    "CacheClusterId": "cluster-test",  
    "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/  
home#client-download:",  
    "CacheNodeType": "cache.m5.large",  
    "Engine": "redis",  
    "EngineVersion": "5.0.5",  
    "CacheClusterStatus": "creating",  
    "NumCacheNodes": 1,  
    "PreferredMaintenanceWindow": "sat:13:00-sat:14:00",  
    "PendingModifiedValues": {},  
    "CacheSecurityGroups": [],  
    "CacheParameterGroup": {  
      "CacheParameterGroupName": "default.redis5.0",  
      "ParameterApplyStatus": "in-sync",  
      "CacheNodeIdsToReboot": []  
    },  
    "CacheSubnetGroupName": "default",  
    "AutoMinorVersionUpgrade": true,  
    "SnapshotRetentionLimit": 0,  
    "SnapshotWindow": "06:30-07:30",  
    "TransitEncryptionEnabled": false,  
    "AtRestEncryptionEnabled": false
```

```
}  
}
```

For more information, see [Creating a Cluster](#) in the *ElastiCache User Guide*.

- For API details, see [CreateCacheCluster](#) in *AWS CLI Command Reference*.

create-cache-parameter-group

The following code example shows how to use `create-cache-parameter-group`.

AWS CLI

To create a cache parameter group

The following `create-cache-parameter-group` example creates a new Amazon ElastiCache cache parameter group.

```
aws elasticache create-cache-parameter-group \  
  --cache-parameter-group-family "redis5.0" \  
  --cache-parameter-group-name "mygroup" \  
  --description "mygroup"
```

Output:

```
{  
  "CacheParameterGroup": {  
    "CacheParameterGroupName": "mygroup",  
    "CacheParameterGroupFamily": "redis5.0",  
    "Description": "my group"  
  }  
}
```

For more information, see [Creating a Parameter Group](#) in the *ElastiCache User Guide*.

- For API details, see [CreateCacheParameterGroup](#) in *AWS CLI Command Reference*.

create-cache-subnet-group

The following code example shows how to use `create-cache-subnet-group`.

AWS CLI

To create a cache subnet group

The following `create-cache-subnet-group` example creates a new cache subnet group.

```
aws elasticache create-cache-subnet-group \  
  --cache-subnet-group-name "mygroup" \  
  --cache-subnet-group-description "my subnet group" \  
  --subnet-ids "subnet-xxxxec4f"
```

Output:

```
{  
  "CacheSubnetGroup": {  
    "CacheSubnetGroupName": "mygroup",  
    "CacheSubnetGroupDescription": "my subnet group",  
    "VpcId": "vpc-a3e97cdb",  
    "Subnets": [  
      {  
        "SubnetIdentifier": "subnet-xxxxec4f",  
        "SubnetAvailabilityZone": {  
          "Name": "us-west-2d"  
        }  
      }  
    ]  
  }  
}
```

For more information, see [Creating a Cache Subnet Group](#) in the *Elasticache User Guide*.

- For API details, see [CreateCacheSubnetGroup](#) in *AWS CLI Command Reference*.

create-global-replication-group

The following code example shows how to use `create-global-replication-group`.

AWS CLI

To create a global replication group

The following `create-global-replication-group` example creates a new global replication group.

```
aws elasticache create-global-replication-group \  
  --global-replication-group-id-suffix my-global-replication-group \  
  --primary-replication-group-id my-primary-cluster
```

Output:

```
{  
  "GlobalReplicationGroup": {  
    "GlobalReplicationGroupId": "sgaui-my-global-replication-group",  
    "GlobalReplicationGroupDescription": " ",  
    "Status": "creating",  
    "CacheNodeType": "cache.r5.large",  
    "Engine": "redis",  
    "EngineVersion": "5.0.6",  
    "Members": [  
      {  
        "ReplicationGroupId": "my-primary-cluster",  
        "ReplicationGroupRegion": "us-west-2",  
        "Role": "PRIMARY",  
        "AutomaticFailover": "enabled",  
        "Status": "associating"  
      }  
    ],  
    "ClusterEnabled": true,  
    "GlobalNodeGroups": [  
      {  
        "GlobalNodeGroupId": "sgaui-my-global-replication-group-0001",  
        "Slots": "0-16383"  
      }  
    ],  
    "AuthTokenEnabled": false,  
    "TransitEncryptionEnabled": false,  
    "AtRestEncryptionEnabled": false  
  }  
}
```

For more information, see [Replication Across AWS Regions Using Global Datastore](#) in the *Elasticache User Guide*.

- For API details, see [CreateGlobalReplicationGroup](#) in *AWS CLI Command Reference*.

create-replication-group

The following code example shows how to use create-replication-group.

AWS CLI

To create a replication group

The following create-replication-group example creates a Redis (cluster mode disabled) or a Redis (cluster mode enabled) replication group. This operation is valid for Redis only.

```
aws elasticache create-replication-group \  
  --replication-group-id "mygroup" \  
  --replication-group-description "my group" \  
  --engine "redis" \  
  --cache-node-type "cache.m5.large"
```

Output:

```
{  
  "ReplicationGroup": {  
    "ReplicationGroupId": "mygroup",  
    "Description": "my group",  
    "Status": "creating",  
    "PendingModifiedValues": {},  
    "MemberClusters": [  
      "mygroup-001"  
    ],  
    "AutomaticFailover": "disabled",  
    "SnapshotRetentionLimit": 0,  
    "SnapshotWindow": "06:00-07:00",  
    "ClusterEnabled": false,  
    "CacheNodeType": "cache.m5.large",  
    "TransitEncryptionEnabled": false,  
    "AtRestEncryptionEnabled": false  
  }  
}
```

For more information, see [Creating a Redis Replication Group](#) in the *Elasticache User Guide*.

- For API details, see [CreateReplicationGroup](#) in *AWS CLI Command Reference*.

create-snapshot

The following code example shows how to use create-snapshot.

AWS CLI

To create a snapshot

The following create-snapshot example creates a snapshot using the Redis engine.

```
aws elasticache create-snapshot \  
  --snapshot-name mysnapshot \  
  --cache-cluster-id cluster-test
```

Output:

```
{  
  "Snapshot": {  
    "SnapshotName": "mysnapshot",  
    "CacheClusterId": "cluster-test",  
    "SnapshotStatus": "creating",  
    "SnapshotSource": "manual",  
    "CacheNodeType": "cache.m5.large",  
    "Engine": "redis",  
    "EngineVersion": "5.0.5",  
    "NumCacheNodes": 1,  
    "PreferredAvailabilityZone": "us-west-2b",  
    "CacheClusterCreateTime": "2020-03-19T03:12:01.483Z",  
    "PreferredMaintenanceWindow": "sat:13:00-sat:14:00",  
    "Port": 6379,  
    "CacheParameterGroupName": "default.redis5.0",  
    "CacheSubnetGroupName": "default",  
    "VpcId": "vpc-a3e97cdb",  
    "AutoMinorVersionUpgrade": true,  
    "SnapshotRetentionLimit": 0,  
    "SnapshotWindow": "06:30-07:30",  
    "NodeSnapshots": [  
      {  
        "CacheNodeId": "0001",  
        "CacheSize": "",  
        "CacheNodeCreateTime": "2020-03-19T03:12:01.483Z"  
      }  
    ]  
  }  
}
```

```
}  
}
```

For more information, see [Backup and Restore for ElastiCache for Redis](#) in the *ElastiCache User Guide*.

- For API details, see [CreateSnapshot](#) in *AWS CLI Command Reference*.

create-user-group

The following code example shows how to use `create-user-group`.

AWS CLI

To create a user group

The following `create-user-group` example creates a new user group.

```
aws elasticache create-user-group \  
  --user-group-id myusergroup \  
  --engine redis \  
  --user-ids default
```

Output:

```
{  
  "UserGroupId": "myusergroup",  
  "Status": "creating",  
  "Engine": "redis",  
  "UserIds": [  
    "default"  
  ],  
  "ReplicationGroups": [],  
  "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxx52:usergroup:myusergroup"  
}
```

For more information, see [Authenticating Users with Role-Based Access Control \(RBAC\)](#) in the *ElastiCache User Guide*.

- For API details, see [CreateUserGroup](#) in *AWS CLI Command Reference*.

create-user

The following code example shows how to use create-user.

AWS CLI

To create a user

The following create-user example creates a new user.

```
aws elasticache create-user \  
  --user-id user1 \  
  --user-name myUser \  
  --passwords mYnuUzrpAxXw2rdzx \  
  --engine redis \  
  --access-string "on ~app:* -@all +@read"
```

Output:

```
{  
  "UserId": "user2",  
  "UserName": "myUser",  
  "Status": "active",  
  "Engine": "redis",  
  "AccessString": "on ~app:* -@all +@read +@hash +@bitmap +@geo -setbit -bitfield  
-hset -hsetnx -hmset -hincrby -hincrbyfloat -hdel -bitop -geoadd -georadius -  
georadiusbymember",  
  "UserGroupIds": [],  
  "Authentication": {  
    "Type": "password",  
    "PasswordCount": 1  
  },  
  "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxx52:user:user2"  
}
```

For more information, see [Authenticating Users with Role-Based Access Control \(RBAC\)](#) in the *Elasticache User Guide*.

- For API details, see [CreateUser](#) in *AWS CLI Command Reference*.

decrease-node-groups-in-global-replication-group

The following code example shows how to use `decrease-node-groups-in-global-replication-group`.

AWS CLI

To decrease the number of node groups in a global replication group

The following `decrease-node-groups-in-global-replication-group` decreases the node group count using the Redis engine.

```
aws elasticache decrease-node-groups-in-global-replication-group \  
  --global-replication-group-id sgaii-test \  
  --node-group-count 1 \  
  --apply-immediately \  
  --global-node-groups-to-retain sgaii-test-0003
```

Output:

```
{  
  "GlobalReplicationGroup":  
  {  
    "GlobalReplicationGroupId": "sgaii-test",  
    "GlobalReplicationGroupDescription": "test",  
    "Status": "modifying",  
    "CacheNodeType": "cache.r5.large",  
    "Engine": "redis",  
    "EngineVersion": "5.0.6",  
    "Members": [  
      {  
        "ReplicationGroupId": "test-2",  
        "ReplicationGroupRegion": "us-east-1",  
        "Role": "SECONDARY",  
        "AutomaticFailover": "enabled",  
        "Status": "associated"  
      },  
      {  
        "ReplicationGroupId": "test-1",  
        "ReplicationGroupRegion": "us-west-2",  
        "Role": "PRIMARY",  
        "AutomaticFailover": "enabled",  
        "Status": "associated"  
      }  
    ]  
  }  
}
```

```

    }
  ],
  "ClusterEnabled": true,
  "GlobalNodeGroups": [
    {
      "GlobalNodeId": "sgaui-test-0001",
      "Slots": "0-449,1816-5461"
    },
    {
      "GlobalNodeId": "sgaui-test-0002",
      "Slots": "6827-10922"
    },
    {
      "GlobalNodeId": "sgaui-test-0003",
      "Slots": "10923-14052,15418-16383"
    },
    {
      "GlobalNodeId": "sgaui-test-0004",
      "Slots": "450-1815,5462-6826,14053-15417"
    }
  ],
  "AuthTokenEnabled": false,
  "TransitEncryptionEnabled": false,
  "AtRestEncryptionEnabled": false
}
}

```

For more information, see [Replication Across AWS Regions Using Global Datastore](#) in the *Elasticache User Guide*.

- For API details, see [DecreaseNodeGroupsInGlobalReplicationGroup](#) in *AWS CLI Command Reference*.

decrease-replica-count

The following code example shows how to use `decrease-replica-count`.

AWS CLI

To decrease replica count

The following `decrease-replica-count` example dynamically decreases the number of replicas in a Redis (cluster mode disabled) replication group or the number of replica nodes

in one or more node groups (shards) of a Redis (cluster mode enabled) replication group. This operation is performed with no cluster downtime.

```
aws elasticache decrease-replica-count \  
  --replication-group-id my-cluster \  
  --apply-immediately \  
  --new-replica-count 2
```

Output:

```
{  
  "ReplicationGroup": {  
    "ReplicationGroupId": "my-cluster",  
    "Description": " ",  
    "Status": "modifying",  
    "PendingModifiedValues": {},  
    "MemberClusters": [  
      "myrepliac",  
      "my-cluster-001",  
      "my-cluster-002",  
      "my-cluster-003"  
    ],  
    "NodeGroups": [  
      {  
        "NodeGroupId": "0001",  
        "Status": "modifying",  
        "PrimaryEndpoint": {  
          "Address": "my-cluster.xxxxx.ng.0001.usw2.cache.amazonaws.com",  
          "Port": 6379  
        },  
        "ReaderEndpoint": {  
          "Address": "my-cluster-  
ro.xxxxx.ng.0001.usw2.cache.amazonaws.com",  
          "Port": 6379  
        },  
        "NodeGroupMembers": [  
          {  
            "CacheClusterId": "myrepliac",  
            "CacheNodeId": "0001",  
            "ReadEndpoint": {  
              "Address":  
"myrepliac.xxxxx.0001.usw2.cache.amazonaws.com",  
              "Port": 6379  
            }  
          }  
        ]  
      }  
    ]  
  }  
}
```

```
    },
    "PreferredAvailabilityZone": "us-west-2a",
    "CurrentRole": "replica"
  },
  {
    "CacheClusterId": "my-cluster-001",
    "CacheNodeId": "0001",
    "ReadEndpoint": {
      "Address": "my-
cluster-001.xxxxx.0001.usw2.cache.amazonaws.com",
      "Port": 6379
    },
    "PreferredAvailabilityZone": "us-west-2a",
    "CurrentRole": "primary"
  },
  {
    "CacheClusterId": "my-cluster-002",
    "CacheNodeId": "0001",
    "ReadEndpoint": {
      "Address": "my-
cluster-002.xxxxx.0001.usw2.cache.amazonaws.com",
      "Port": 6379
    },
    "PreferredAvailabilityZone": "us-west-2a",
    "CurrentRole": "replica"
  },
  {
    "CacheClusterId": "my-cluster-003",
    "CacheNodeId": "0001",
    "ReadEndpoint": {
      "Address": "my-
cluster-003.xxxxx.0001.usw2.cache.amazonaws.com",
      "Port": 6379
    },
    "PreferredAvailabilityZone": "us-west-2a",
    "CurrentRole": "replica"
  }
]
}
],
"AutomaticFailover": "disabled",
"SnapshotRetentionLimit": 0,
"SnapshotWindow": "07:30-08:30",
"ClusterEnabled": false,
```

```
    "CacheNodeType": "cache.r5.xlarge",
    "TransitEncryptionEnabled": false,
    "AtRestEncryptionEnabled": false
  }
}
```

For more information, see [Changing the Number of Replicas](#) in the *Elasticache User Guide*.

- For API details, see [DecreaseReplicaCount](#) in *AWS CLI Command Reference*.

delete-cache-cluster

The following code example shows how to use `delete-cache-cluster`.

AWS CLI

To delete a cache cluster

The following `delete-cache-cluster` example deletes the specified previously provisioned cluster. The command deletes all associated cache nodes, node endpoints, and the cluster itself. When you receive a successful response from this operation, Amazon ElastiCache immediately begins deleting the cluster; you can't cancel or revert this operation.

This operation is not valid for the following:

Redis (cluster mode enabled) clusters
A cluster that is the last read replica of a replication group
A node group (shard) that has Multi-AZ mode enabled
A cluster from a Redis (cluster mode enabled) replication group
A cluster that is not in the available state

```
aws elasticache delete-cache-cluster \
  --cache-cluster-id "my-cluster-002"
```

Output:

```
{
  "CacheCluster": {
    "CacheClusterId": "my-cluster-002",
    "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/home#client-download:",
    "CacheNodeType": "cache.r5.xlarge",
    "Engine": "redis",
    "EngineVersion": "5.0.5",
```

```

    "CacheClusterStatus": "deleting",
    "NumCacheNodes": 1,
    "PreferredAvailabilityZone": "us-west-2a",
    "CacheClusterCreateTime": "2019-11-26T03:35:04.546Z",
    "PreferredMaintenanceWindow": "mon:04:05-mon:05:05",
    "PendingModifiedValues": {},
    "NotificationConfiguration": {
        "TopicArn": "arn:aws:sns:us-west-x:xxxxxxx4152:My_Topic",
        "TopicStatus": "active"
    },
    "CacheSecurityGroups": [],
    "CacheParameterGroup": {
        "CacheParameterGroupName": "mygroup",
        "ParameterApplyStatus": "in-sync",
        "CacheNodeIdsToReboot": []
    },
    "CacheSubnetGroupName": "kxkxk",
    "AutoMinorVersionUpgrade": true,
    "SecurityGroups": [
        {
            "SecurityGroupId": "sg-xxxxxxxxxx9836",
            "Status": "active"
        },
        {
            "SecurityGroupId": "sg-xxxxxxxxxxx7b",
            "Status": "active"
        }
    ],
    "ReplicationGroupId": "my-cluster",
    "SnapshotRetentionLimit": 0,
    "SnapshotWindow": "07:30-08:30",
    "TransitEncryptionEnabled": false,
    "AtRestEncryptionEnabled": false
}
}

```

For more information, see [Deleting a Cluster](#) in the *ElastiCache User Guide*.

- For API details, see [DeleteCacheCluster](#) in *AWS CLI Command Reference*.

delete-cache-parameter-group

The following code example shows how to use `delete-cache-parameter-group`.

AWS CLI

To delete a cache parameter group

The following `delete-cache-parameter-group` example deletes the specified cache parameter group. You can't delete a cache parameter group if it's associated with any cache clusters.

```
aws elasticache delete-cache-parameter-group \  
  --cache-parameter-group-name myparamgroup
```

This command produces no output.

For more information, see [Deleting a Parameter Group](#) in the *Elasticache User Guide*.

- For API details, see [DeleteCacheParameterGroup](#) in *AWS CLI Command Reference*.

delete-cache-subnet-group

The following code example shows how to use `delete-cache-subnet-group`.

AWS CLI

To delete a cache subnet group

The following `delete-cache-subnet-group` example deletes the specified cache subnet group. You can't delete a cache subnet group if it's associated with any clusters.

```
aws elasticache delete-cache-subnet-group \  
  --cache-subnet-group-name "mygroup"
```

This command produces no output.

For more information, see [Deleting a Subnet Group](#) in the *Elasticache User Guide*.

- For API details, see [DeleteCacheSubnetGroup](#) in *AWS CLI Command Reference*.

delete-global-replication-group

The following code example shows how to use `delete-global-replication-group`.

AWS CLI

To delete a global replication group

The following `delete-global-replication-group` example deletes a new global replication group.

```
aws elasticache delete-global-replication-group \  
  --global-replication-group-id my-global-replication-group \  
  --retain-primary-replication-group
```

Output:

```
{  
  "GlobalReplicationGroup": {  
    "GlobalReplicationGroupId": "sgaui-my-grg",  
    "GlobalReplicationGroupDescription": "my-grg",  
    "Status": "deleting",  
    "CacheNodeType": "cache.r5.large",  
    "Engine": "redis",  
    "EngineVersion": "5.0.6",  
    "Members": [  
      {  
        "ReplicationGroupId": "my-cluster-grg",  
        "ReplicationGroupRegion": "us-west-2",  
        "Role": "PRIMARY",  
        "AutomaticFailover": "enabled",  
        "Status": "associated"  
      }  
    ],  
    "ClusterEnabled": false,  
    "AuthTokenEnabled": false,  
    "TransitEncryptionEnabled": false,  
    "AtRestEncryptionEnabled": false  
  }  
}
```

For more information, see [Replication Across AWS Regions Using Global Datastore](#) in the *Elasticache User Guide*.

- For API details, see [DeleteGlobalReplicationGroup](#) in *AWS CLI Command Reference*.

delete-replication-group

The following code example shows how to use delete-replication-group.

AWS CLI

To delete a replication group

The following delete-replication-group example deletes an existing replication group. By default, this operation deletes the entire replication group, including the primary/primaries and all of the read replicas. If the replication group has only one primary, you can optionally delete only the read replicas, while retaining the primary by setting RetainPrimaryCluster=true .

When you receive a successful response from this operation, Amazon ElastiCache immediately begins deleting the selected resources; you cannot cancel or revert this operation. Valid for Redis only.

```
aws elasticache delete-replication-group \  
  --replication-group-id "mygroup"
```

Output:

```
{  
  "ReplicationGroup": {  
    "ReplicationGroupId": "mygroup",  
    "Description": "my group",  
    "Status": "deleting",  
    "PendingModifiedValues": {},  
    "AutomaticFailover": "disabled",  
    "SnapshotRetentionLimit": 0,  
    "SnapshotWindow": "06:00-07:00",  
    "TransitEncryptionEnabled": false,  
    "AtRestEncryptionEnabled": false  
  }  
}
```

- For API details, see [DeleteReplicationGroup](#) in *AWS CLI Command Reference*.

delete-snapshot

The following code example shows how to use delete-snapshot.

AWS CLI

To delete a snapshot

The following `delete-snapshot` example deleted a snapshot using the Redis engine.

```
aws elasticache delete-snapshot \  
  --snapshot-name mysnapshot
```

Output:

```
{  
  "Snapshot": {  
    "SnapshotName": "my-cluster-snapshot",  
    "ReplicationGroupId": "mycluster",  
    "ReplicationGroupDescription": "mycluster",  
    "SnapshotStatus": "deleting",  
    "SnapshotSource": "manual",  
    "CacheNodeType": "cache.r5.xlarge",  
    "Engine": "redis",  
    "EngineVersion": "5.0.5",  
    "PreferredMaintenanceWindow": "thu:12:00-thu:13:00",  
    "TopicArn": "arn:aws:sns:us-west-2:xxxxxxxxxxxxx152:My_Topic",  
    "Port": 6379,  
    "CacheParameterGroupName": "default.redis5.0.cluster.on",  
    "CacheSubnetGroupName": "default",  
    "VpcId": "vpc-a3e97cdb",  
    "AutoMinorVersionUpgrade": true,  
    "SnapshotRetentionLimit": 1,  
    "SnapshotWindow": "13:00-14:00",  
    "NumNodeGroups": 4,  
    "AutomaticFailover": "enabled",  
    "NodeSnapshots": [  
      {  
        "CacheClusterId": "mycluster-0002-003",  
        "NodeGroupId": "0002",  
        "CacheNodeId": "0001",  
        "CacheSize": "6 MB",  
        "CacheNodeCreateTime": "2020-06-18T00:05:44.719000+00:00",  
        "SnapshotCreateTime": "2020-06-25T20:34:30+00:00"  
      },  
      {  
        "CacheClusterId": "mycluster-0003-003",
```

```

        "NodeGroupId": "0003",
        "CacheNodeId": "0001",
        "CacheSize": "6 MB",
        "CacheNodeCreateTime": "2019-12-05T19:13:15.912000+00:00",
        "SnapshotCreateTime": "2020-06-25T20:34:30+00:00"
    },
    {
        "CacheClusterId": "mycluster-0004-002",
        "NodeGroupId": "0004",
        "CacheNodeId": "0001",
        "CacheSize": "6 MB",
        "CacheNodeCreateTime": "2019-12-09T19:44:34.324000+00:00",
        "SnapshotCreateTime": "2020-06-25T20:34:30+00:00"
    },
    {
        "CacheClusterId": "mycluster-0005-003",
        "NodeGroupId": "0005",
        "CacheNodeId": "0001",
        "CacheSize": "6 MB",
        "CacheNodeCreateTime": "2020-06-18T00:05:44.775000+00:00",
        "SnapshotCreateTime": "2020-06-25T20:34:30+00:00"
    }
]
}
}

```

For more information, see [Backup and Restore for ElastiCache for Redis](#) in the *ElastiCache User Guide*.

- For API details, see [DeleteSnapshot](#) in *AWS CLI Command Reference*.

delete-user-group

The following code example shows how to use `delete-user-group`.

AWS CLI

To delete a user group

The following `delete-user-group` example deletes a user group.

```
aws elasticache delete-user-group \
  --user-group-id myusergroup
```

Output:

```
{
  "UserGroupId": "myusergroup",
  "Status": "deleting",
  "Engine": "redis",
  "UserIds": [
    "default"
  ],
  "ReplicationGroups": [],
  "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxx52:usergroup:myusergroup"
}
```

For more information, see [Authenticating Users with Role-Based Access Control \(RBAC\)](#) in the *Elasticache User Guide*.

- For API details, see [DeleteUserGroup](#) in *AWS CLI Command Reference*.

delete-user

The following code example shows how to use `delete-user`.

AWS CLI**To delete a user**

The following `delete-user` example deletes a user.

```
aws elasticache delete-user \  
  --user-id user2
```

Output:

```
{
  "UserId": "user1",
  "UserName": "myUser",
  "Status": "deleting",
  "Engine": "redis",
  "AccessString": "on ~* +@all",
  "UserGroupIds": [
    "myusergroup"
  ],
  "Authentication": {
```

```
    "Type": "password",
    "PasswordCount": 1
  },
  "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxx52:user:user1"
}
```

For more information, see [Authenticating Users with Role-Based Access Control \(RBAC\)](#) in the *Elasticache User Guide*.

- For API details, see [DeleteUser](#) in *AWS CLI Command Reference*.

describe-cache-clusters

The following code example shows how to use `describe-cache-clusters`.

AWS CLI

To describe a cache cluster

The following `describe-cache-clusters` example describes a cache cluster.

```
aws elasticache describe-cache-clusters
```

Output:

```
{
  "CacheClusters": [
    {
      "CacheClusterId": "my-cluster-003",
      "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/home#client-download:",
      "CacheNodeType": "cache.r5.large",
      "Engine": "redis",
      "EngineVersion": "5.0.5",
      "CacheClusterStatus": "available",
      "NumCacheNodes": 1,
      "PreferredAvailabilityZone": "us-west-2a",
      "CacheClusterCreateTime": "2019-11-26T01:22:52.396Z",
      "PreferredMaintenanceWindow": "mon:17:30-mon:18:30",
      "PendingModifiedValues": {},
      "NotificationConfiguration": {
        "TopicArn": "arn:aws:sns:us-west-2:xxxxxxxxxxxx152:My_Topic",
        "TopicStatus": "active"
      }
    }
  ]
}
```

```

    },
    "CacheSecurityGroups": [],
    "CacheParameterGroup": {
        "CacheParameterGroupName": "default.redis5.0",
        "ParameterApplyStatus": "in-sync",
        "CacheNodeIdsToReboot": []
    },
    "CacheSubnetGroupName": "kxkxk",
    "AutoMinorVersionUpgrade": true,
    "SecurityGroups": [
        {
            "SecurityGroupId": "sg-xxxxxd7b",
            "Status": "active"
        }
    ],
    "ReplicationGroupId": "my-cluster",
    "SnapshotRetentionLimit": 0,
    "SnapshotWindow": "06:30-07:30",
    "AuthTokenEnabled": false,
    "TransitEncryptionEnabled": false,
    "AtRestEncryptionEnabled": false,
    "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxxx152:cluster:my-cache-
cluster",
    "ReplicationGroupLogDeliveryEnabled": false,
    "LogDeliveryConfigurations": [
        {
            "LogType": "slow-log",
            "DestinationType": "cloudwatch-logs",
            "DestinationDetails": {
                "CloudWatchLogsDetails": {
                    "LogGroup": "test-log"
                }
            },
            "LogFormat": "text",
            "Status": "active"
        }
    ]
}
]
}
}

```

For more information, see [Managing Clusters](#) in the *Elasticache User Guide*.

- For API details, see [DescribeCacheClusters](#) in *AWS CLI Command Reference*.

describe-cache-engine-versions

The following code example shows how to use describe-cache-engine-versions.

AWS CLI

To describe a cache engine version

The following describe-cache-engine-versions example returns a list of the available cache engines and their versions.

```
aws elasticache describe-cache-engine-versions \  
  --engine "Redis"
```

Output:

```
{  
  "CacheEngineVersions": [  
    {  
      "Engine": "redis",  
      "EngineVersion": "2.6.13",  
      "CacheParameterGroupFamily": "redis2.6",  
      "CacheEngineDescription": "Redis",  
      "CacheEngineVersionDescription": "redis version 2.6.13"  
    },  
    {  
      "Engine": "redis",  
      "EngineVersion": "2.8.19",  
      "CacheParameterGroupFamily": "redis2.8",  
      "CacheEngineDescription": "Redis",  
      "CacheEngineVersionDescription": "redis version 2.8.19"  
    },  
    {  
      "Engine": "redis",  
      "EngineVersion": "2.8.21",  
      "CacheParameterGroupFamily": "redis2.8",  
      "CacheEngineDescription": "Redis",  
      "CacheEngineVersionDescription": "redis version 2.8.21"  
    },  
    {  
      "Engine": "redis",  
      "EngineVersion": "2.8.22",  
      "CacheParameterGroupFamily": "redis2.8",
```



```
    "CacheEngineDescription": "Redis",
    "CacheEngineVersionDescription": "redis version 2.8.22"
  },
  {
    "Engine": "redis",
    "EngineVersion": "2.8.23",
    "CacheParameterGroupFamily": "redis2.8",
    "CacheEngineDescription": "Redis",
    "CacheEngineVersionDescription": "redis version 2.8.23"
  },
  {
    "Engine": "redis",
    "EngineVersion": "2.8.24",
    "CacheParameterGroupFamily": "redis2.8",
    "CacheEngineDescription": "Redis",
    "CacheEngineVersionDescription": "redis version 2.8.24"
  },
  {
    "Engine": "redis",
    "EngineVersion": "2.8.6",
    "CacheParameterGroupFamily": "redis2.8",
    "CacheEngineDescription": "Redis",
    "CacheEngineVersionDescription": "redis version 2.8.6"
  },
  {
    "Engine": "redis",
    "EngineVersion": "3.2.10",
    "CacheParameterGroupFamily": "redis3.2",
    "CacheEngineDescription": "Redis",
    "CacheEngineVersionDescription": "redis version 3.2.10"
  },
  {
    "Engine": "redis",
    "EngineVersion": "3.2.4",
    "CacheParameterGroupFamily": "redis3.2",
    "CacheEngineDescription": "Redis",
    "CacheEngineVersionDescription": "redis version 3.2.4"
  },
  {
    "Engine": "redis",
    "EngineVersion": "3.2.6",
    "CacheParameterGroupFamily": "redis3.2",
    "CacheEngineDescription": "Redis",
    "CacheEngineVersionDescription": "redis version 3.2.6"
  }
```

```
    },
    {
      "Engine": "redis",
      "EngineVersion": "4.0.10",
      "CacheParameterGroupFamily": "redis4.0",
      "CacheEngineDescription": "Redis",
      "CacheEngineVersionDescription": "redis version 4.0.10"
    },
    {
      "Engine": "redis",
      "EngineVersion": "5.0.0",
      "CacheParameterGroupFamily": "redis5.0",
      "CacheEngineDescription": "Redis",
      "CacheEngineVersionDescription": "redis version 5.0.0"
    },
    {
      "Engine": "redis",
      "EngineVersion": "5.0.3",
      "CacheParameterGroupFamily": "redis5.0",
      "CacheEngineDescription": "Redis",
      "CacheEngineVersionDescription": "redis version 5.0.3"
    },
    {
      "Engine": "redis",
      "EngineVersion": "5.0.4",
      "CacheParameterGroupFamily": "redis5.0",
      "CacheEngineDescription": "Redis",
      "CacheEngineVersionDescription": "redis version 5.0.4"
    },
    {
      "Engine": "redis",
      "EngineVersion": "5.0.5",
      "CacheParameterGroupFamily": "redis5.0",
      "CacheEngineDescription": "Redis",
      "CacheEngineVersionDescription": "redis version 5.0.5"
    }
  ]
}
```

- For API details, see [DescribeCacheEngineVersions](#) in *AWS CLI Command Reference*.

describe-cache-parameter-groups

The following code example shows how to use `describe-cache-parameter-groups`.

AWS CLI

To describe a cache parameter group

The following `describe-cache-parameter-groups` example returns a list of cache parameter group descriptions.

```
aws elasticache describe-cache-parameter-groups \  
  --cache-parameter-group-name "mygroup"
```

Output:

```
{  
  "CacheParameterGroups": [  
    {  
      "CacheParameterGroupName": "mygroup",  
      "CacheParameterGroupFamily": "redis5.0",  
      "Description": " "  
    }  
  ]  
}
```

For more information, see [Configuring Engine Parameters Using Parameter Groups](#) in the *Elasticache User Guide*.

- For API details, see [DescribeCacheParameterGroups](#) in *AWS CLI Command Reference*.

describe-cache-parameters

The following code example shows how to use `describe-cache-parameters`.

AWS CLI

To describe cache parameters

The following `"describe-cache-parameters"` example returns the detailed parameter list for the specified cache parameter group.

```
aws elasticache describe-cache-parameters \  
  --cache-parameter-group-name "myparamgroup"
```

Output:

```
{  
  "Parameters": [  
    {  
      "ParameterName": "activedefrag",  
      "ParameterValue": "yes",  
      "Description": "Enabled active memory defragmentation",  
      "Source": "user",  
      "DataType": "string",  
      "AllowedValues": "yes,no",  
      "IsModifiable": true,  
      "MinimumEngineVersion": "5.0.0",  
      "ChangeType": "immediate"  
    },  
    {  
      "ParameterName": "active-defrag-cycle-max",  
      "ParameterValue": "75",  
      "Description": "Maximal effort for defrag in CPU percentage",  
      "Source": "user",  
      "DataType": "integer",  
      "AllowedValues": "1-75",  
      "IsModifiable": true,  
      "MinimumEngineVersion": "5.0.0",  
      "ChangeType": "immediate"  
    },  
    {  
      "ParameterName": "active-defrag-cycle-min",  
      "ParameterValue": "5",  
      "Description": "Minimal effort for defrag in CPU percentage",  
      "Source": "user",  
      "DataType": "integer",  
      "AllowedValues": "1-75",  
      "IsModifiable": true,  
      "MinimumEngineVersion": "5.0.0",  
      "ChangeType": "immediate"  
    },  
    {  
      "ParameterName": "active-defrag-ignore-bytes",  
      "ParameterValue": "104857600",  
      "Description": "Minimal effort for defrag in CPU percentage",  
      "Source": "user",  
      "DataType": "integer",  
      "AllowedValues": "1-75",  
      "IsModifiable": true,  
      "MinimumEngineVersion": "5.0.0",  
      "ChangeType": "immediate"  
    }  
  ]  
}
```

```

    "Description": "Minimum amount of fragmentation waste to start active
defrag",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "1048576-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "active-defrag-max-scan-fields",
    "ParameterValue": "1000",
    "Description": "Maximum number of set/hash/zset/list fields that will be
processed from the main dictionary scan",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "1-1000000",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "active-defrag-threshold-lower",
    "ParameterValue": "10",
    "Description": "Minimum percentage of fragmentation to start active
defrag",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "1-100",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "active-defrag-threshold-upper",
    "ParameterValue": "100",
    "Description": "Maximum percentage of fragmentation at which we use
maximum effort",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "1-100",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  }
}

```

```
  },
  {
    "ParameterName": "activeresharding",
    "ParameterValue": "yes",
    "Description": "Apply rehashing or not.",
    "Source": "user",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "requires-reboot"
  },
  {
    "ParameterName": "appendfsync",
    "ParameterValue": "everysec",
    "Description": "fsync policy for AOF persistence",
    "Source": "system",
    "DataType": "string",
    "AllowedValues": "always,everysec,no",
    "IsModifiable": false,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "appendonly",
    "ParameterValue": "no",
    "Description": "Enable Redis persistence.",
    "Source": "system",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "IsModifiable": false,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "client-output-buffer-limit-normal-hard-limit",
    "ParameterValue": "0",
    "Description": "Normal client output buffer hard limit in bytes.",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  }
}
```

```
  },
  {
    "ParameterName": "client-output-buffer-limit-normal-soft-limit",
    "ParameterValue": "0",
    "Description": "Normal client output buffer soft limit in bytes.",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "client-output-buffer-limit-normal-soft-seconds",
    "ParameterValue": "0",
    "Description": "Normal client output buffer soft limit in seconds.",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "client-output-buffer-limit-pubsub-hard-limit",
    "ParameterValue": "33554432",
    "Description": "Pubsub client output buffer hard limit in bytes.",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "client-output-buffer-limit-pubsub-soft-limit",
    "ParameterValue": "8388608",
    "Description": "Pubsub client output buffer soft limit in bytes.",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  }
}
```

```
  },
  {
    "ParameterName": "client-output-buffer-limit-pubsub-soft-seconds",
    "ParameterValue": "60",
    "Description": "Pubsub client output buffer soft limit in seconds.",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "client-output-buffer-limit-replica-soft-seconds",
    "ParameterValue": "60",
    "Description": "Replica client output buffer soft limit in seconds.",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": false,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "client-query-buffer-limit",
    "ParameterValue": "1073741824",
    "Description": "Max size of a single client query buffer",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "1048576-1073741824",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "close-on-replica-write",
    "ParameterValue": "yes",
    "Description": "If enabled, clients who attempt to write to a read-only replica will be disconnected. Applicable to 2.8.23 and higher.",
    "Source": "user",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
```



```
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "cluster-enabled",
    "ParameterValue": "no",
    "Description": "Enable cluster mode",
    "Source": "user",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "requires-reboot"
  },
  {
    "ParameterName": "cluster-require-full-coverage",
    "ParameterValue": "no",
    "Description": "Whether cluster becomes unavailable if one or more slots
are not covered",
    "Source": "user",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "databases",
    "ParameterValue": "16",
    "Description": "Set the number of databases.",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "1-1200000",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "requires-reboot"
  },
  {
    "ParameterName": "hash-max-ziplist-entries",
    "ParameterValue": "512",
    "Description": "The maximum number of hash entries in order for the
dataset to be compressed.",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "0-",
```

```
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "hash-max-ziplist-value",
    "ParameterValue": "64",
    "Description": "The threshold of biggest hash entries in order for the
dataset to be compressed.",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "hll-sparse-max-bytes",
    "ParameterValue": "3000",
    "Description": "HyperLogLog sparse representation bytes limit",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "1-16000",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "lazyfree-lazy-eviction",
    "ParameterValue": "no",
    "Description": "Perform an asynchronous delete on evictions",
    "Source": "user",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "lazyfree-lazy-expire",
    "ParameterValue": "no",
    "Description": "Perform an asynchronous delete on expired keys",
    "Source": "user",
    "DataType": "string",
```

```
    "AllowedValues": "yes,no",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "lazyfree-lazy-server-del",
    "ParameterValue": "no",
    "Description": "Perform an asynchronous delete on key updates",
    "Source": "user",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "lfu-decay-time",
    "ParameterValue": "1",
    "Description": "The amount of time in minutes to decrement the key
counter for LFU eviction policy",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "lfu-log-factor",
    "ParameterValue": "10",
    "Description": "The log factor for incrementing key counter for LFU
eviction policy",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "1-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "list-compress-depth",
    "ParameterValue": "0",
```

```

    "Description": "Number of quicklist ziplist nodes from each side of
the list to exclude from compression. The head and tail of the list are always
uncompressed for fast push/pop operations",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "list-max-ziplist-size",
    "ParameterValue": "-2",
    "Description": "The number of entries allowed per internal list node can
be specified as a fixed maximum size or a maximum number of elements",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "-5,-4,-3,-2,-1,1-",
    "IsModifiable": false,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "lua-replicate-commands",
    "ParameterValue": "yes",
    "Description": "Always enable Lua effect replication or not",
    "Source": "user",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "lua-time-limit",
    "ParameterValue": "5000",
    "Description": "Max execution time of a Lua script in milliseconds. 0
for unlimited execution without warnings.",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "5000",
    "IsModifiable": false,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  }

```

```

    },
    {
      "ParameterName": "maxclients",
      "ParameterValue": "65000",
      "Description": "The maximum number of Redis clients.",
      "Source": "system",
      "DataType": "integer",
      "AllowedValues": "1-65000",
      "IsModifiable": false,
      "MinimumEngineVersion": "5.0.0",
      "ChangeType": "requires-reboot"
    },
    {
      "ParameterName": "maxmemory-policy",
      "ParameterValue": "volatile-lru",
      "Description": "Max memory policy.",
      "Source": "user",
      "DataType": "string",
      "AllowedValues": "volatile-lru,allkeys-lru,volatile-lfu,allkeys-lfu,volatile-random,allkeys-random,volatile-ttl,noeviction",
      "IsModifiable": true,
      "MinimumEngineVersion": "5.0.0",
      "ChangeType": "immediate"
    },
    {
      "ParameterName": "maxmemory-samples",
      "ParameterValue": "3",
      "Description": "Max memory samples.",
      "Source": "user",
      "DataType": "integer",
      "AllowedValues": "1-",
      "IsModifiable": true,
      "MinimumEngineVersion": "5.0.0",
      "ChangeType": "immediate"
    },
    {
      "ParameterName": "min-replicas-max-lag",
      "ParameterValue": "10",
      "Description": "The maximum amount of replica lag in seconds beyond which the master would stop taking writes. A value of 0 means the master always takes writes.",
      "Source": "user",
      "DataType": "integer",
      "AllowedValues": "0-",

```

```
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "min-replicas-to-write",
    "ParameterValue": "0",
    "Description": "The minimum number of replicas that must be present with
lag no greater than min-replicas-max-lag for master to take writes. Setting this to
0 means the master always takes writes.",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "notify-keyspace-events",
    "Description": "The keyspace events for Redis to notify Pub/Sub clients
about. By default all notifications are disabled",
    "Source": "user",
    "DataType": "string",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "proto-max-bulk-len",
    "ParameterValue": "536870912",
    "Description": "Max size of a single element request",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "1048576-536870912",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "rename-commands",
    "ParameterValue": "",
    "Description": "Redis commands that can be dynamically renamed by the
customer",
    "Source": "user",
```

```

        "DataType": "string",
        "AllowedValues":
"APPEND,BITCOUNT,BITFIELD,BITOP,BITPOS,BLPOP,BRPOP,BRPOPLPUSH,BZPOPMIN,BZPOPMAX,CLIENT,COMM
        "IsModifiable": true,
        "MinimumEngineVersion": "5.0.3",
        "ChangeType": "immediate"
    },
    {
        "ParameterName": "repl-backlog-size",
        "ParameterValue": "1048576",
        "Description": "The replication backlog size in bytes for PSYNC. This is
the size of the buffer which accumulates slave data when slave is disconnected for
some time, so that when slave reconnects again, only transfer the portion of data
which the slave missed. Minimum value is 16K.",
        "Source": "user",
        "DataType": "integer",
        "AllowedValues": "16384-",
        "IsModifiable": true,
        "MinimumEngineVersion": "5.0.0",
        "ChangeType": "immediate"
    },
    {
        "ParameterName": "repl-backlog-ttl",
        "ParameterValue": "3600",
        "Description": "The amount of time in seconds after the master no longer
have any slaves connected for the master to free the replication backlog. A value
of 0 means to never release the backlog.",
        "Source": "user",
        "DataType": "integer",
        "AllowedValues": "0-",
        "IsModifiable": true,
        "MinimumEngineVersion": "5.0.0",
        "ChangeType": "immediate"
    },
    {
        "ParameterName": "replica-allow-chaining",
        "ParameterValue": "no",
        "Description": "Configures if chaining of replicas is allowed",
        "Source": "system",
        "DataType": "string",
        "AllowedValues": "yes,no",
        "IsModifiable": false,
        "MinimumEngineVersion": "5.0.0",
        "ChangeType": "immediate"
    }

```

```
    },
    {
      "ParameterName": "replica-ignore-maxmemory",
      "ParameterValue": "yes",
      "Description": "Determines if replica ignores maxmemory setting by not
evicting items independent from the master",
      "Source": "system",
      "DataType": "string",
      "AllowedValues": "yes,no",
      "IsModifiable": false,
      "MinimumEngineVersion": "5.0.0",
      "ChangeType": "immediate"
    },
    {
      "ParameterName": "replica-lazy-flush",
      "ParameterValue": "no",
      "Description": "Perform an asynchronous flushDB during replica sync",
      "Source": "system",
      "DataType": "string",
      "AllowedValues": "yes,no",
      "IsModifiable": false,
      "MinimumEngineVersion": "5.0.0",
      "ChangeType": "immediate"
    },
    {
      "ParameterName": "reserved-memory-percent",
      "ParameterValue": "25",
      "Description": "The percent of memory reserved for non-cache memory
usage. You may want to increase this parameter for nodes with read replicas, AOF
enabled, etc, to reduce swap usage.",
      "Source": "user",
      "DataType": "integer",
      "AllowedValues": "0-100",
      "IsModifiable": true,
      "MinimumEngineVersion": "5.0.0",
      "ChangeType": "immediate"
    },
    {
      "ParameterName": "set-max-intset-entries",
      "ParameterValue": "512",
      "Description": "The limit in the size of the set in order for the
dataset to be compressed.",
      "Source": "user",
      "DataType": "integer",
```



```
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "slowlog-log-slower-than",
    "ParameterValue": "10000",
    "Description": "The execution time, in microseconds, to exceed in order
for the command to get logged. Note that a negative number disables the slow log,
while a value of zero forces the logging of every command.",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "slowlog-max-len",
    "ParameterValue": "128",
    "Description": "The length of the slow log. There is no limit to this
length. Just be aware that it will consume memory. You can reclaim memory used by
the slow log with SLOWLOG RESET.",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "stream-node-max-bytes",
    "ParameterValue": "4096",
    "Description": "The maximum size of a single node in a stream in bytes",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "stream-node-max-entries",
```

```

    "ParameterValue": "100",
    "Description": "The maximum number of items a single node in a stream
can contain",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "tcp-keepalive",
    "ParameterValue": "300",
    "Description": "If non-zero, send ACKs every given number of seconds.",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "timeout",
    "ParameterValue": "0",
    "Description": "Close connection if client is idle for a given number of
seconds, or never if 0.",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "0,20-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "zset-max-ziplist-entries",
    "ParameterValue": "128",
    "Description": "The maximum number of sorted set entries in order for
the dataset to be compressed.",
    "Source": "user",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  }

```

```

    },
    {
      "ParameterName": "zset-max-ziplist-value",
      "ParameterValue": "64",
      "Description": "The threshold of biggest sorted set entries in order for
the dataset to be compressed.",
      "Source": "user",
      "DataType": "integer",
      "AllowedValues": "0-",
      "IsModifiable": true,
      "MinimumEngineVersion": "5.0.0",
      "ChangeType": "immediate"
    }
  ]
}

```

For more information, see [Parameter Management](#) in the *Elasticache User Guide*.

- For API details, see [DescribeCacheParameters](#) in *AWS CLI Command Reference*.

describe-cache-subnet-groups

The following code example shows how to use `describe-cache-subnet-groups`.

AWS CLI

To describe cache subnet groups

The following `describe-cache-subnet-groups` example returns a list of subnet groups.

```
aws elasticache describe-cache-subnet-groups
```

Output:

```

{
  "CacheSubnetGroups": [
    {
      "CacheSubnetGroupName": "default",
      "CacheSubnetGroupDescription": "Default CacheSubnetGroup",
      "VpcId": "vpc-a3e97cdb",
      "Subnets": [
        {
          "SubnetIdentifier": "subnet-8d4bacf5",

```

```
        "SubnetAvailabilityZone": {
            "Name": "us-west-2b"
        }
    },
    {
        "SubnetIdentifier": "subnet-dde21380",
        "SubnetAvailabilityZone": {
            "Name": "us-west-2c"
        }
    },
    {
        "SubnetIdentifier": "subnet-6485ec4f",
        "SubnetAvailabilityZone": {
            "Name": "us-west-2d"
        }
    },
    {
        "SubnetIdentifier": "subnet-b4ebebff",
        "SubnetAvailabilityZone": {
            "Name": "us-west-2a"
        }
    }
]
},
{
    "CacheSubnetGroupName": "kxxkxk",
    "CacheSubnetGroupDescription": "mygroup",
    "VpcId": "vpc-a3e97cdb",
    "Subnets": [
        {
            "SubnetIdentifier": "subnet-b4ebebff",
            "SubnetAvailabilityZone": {
                "Name": "us-west-2a"
            }
        }
    ]
},
{
    "CacheSubnetGroupName": "test",
    "CacheSubnetGroupDescription": "test",
    "VpcId": "vpc-a3e97cdb",
    "Subnets": [
        {
            "SubnetIdentifier": "subnet-b4ebebff",
```

```

        "SubnetAvailabilityZone": {
            "Name": "us-west-2a"
        }
    ]
}

```

For more information, see [Subnets and Subnet Groups](#) in the *Elasticache User Guide* or [Subnets and Subnet Groups](#) in the *ElastiCache for Memcached User Guide*.

- For API details, see [DescribeCacheSubnetGroups](#) in *AWS CLI Command Reference*.

describe-engine-default-parameters

The following code example shows how to use `describe-engine-default-parameters`.

AWS CLI

To describe engine default parameters

The following `describe-engine-default-parameters` example returns the default engine and system parameter information for the specified cache engine.

```

aws elasticache describe-engine-default-parameters \
  --cache-parameter-group-family "redis5.0"

```

Output:

```

{
  "EngineDefaults": {
    "Parameters": [
      {
        "ParameterName": "activedefrag",
        "ParameterValue": "no",
        "Description": "Enabled active memory defragmentation",
        "Source": "system",
        "DataType": "string",
        "AllowedValues": "yes,no",
        "IsModifiable": true,
        "MinimumEngineVersion": "5.0.0",
        "ChangeType": "immediate"
      }
    ]
  }
}

```

```
    },
    {
      "ParameterName": "active-defrag-cycle-max",
      "ParameterValue": "75",
      "Description": "Maximal effort for defrag in CPU percentage",
      "Source": "system",
      "DataType": "integer",
      "AllowedValues": "1-75",
      "IsModifiable": true,
      "MinimumEngineVersion": "5.0.0",
      "ChangeType": "immediate"
    },
    {
      "ParameterName": "active-defrag-cycle-min",
      "ParameterValue": "5",
      "Description": "Minimal effort for defrag in CPU percentage",
      "Source": "system",
      "DataType": "integer",
      "AllowedValues": "1-75",
      "IsModifiable": true,
      "MinimumEngineVersion": "5.0.0",
      "ChangeType": "immediate"
    },
    {
      "ParameterName": "active-defrag-ignore-bytes",
      "ParameterValue": "104857600",
      "Description": "Minimum amount of fragmentation waste to start
active defrag",
      "Source": "system",
      "DataType": "integer",
      "AllowedValues": "1048576-",
      "IsModifiable": true,
      "MinimumEngineVersion": "5.0.0",
      "ChangeType": "immediate"
    },
    {
      "ParameterName": "active-defrag-max-scan-fields",
      "ParameterValue": "1000",
      "Description": "Maximum number of set/hash/zset/list fields that
will be processed from the main dictionary scan",
      "Source": "system",
      "DataType": "integer",
      "AllowedValues": "1-1000000",
      "IsModifiable": true,
```

```

    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "active-defrag-threshold-lower",
    "ParameterValue": "10",
    "Description": "Minimum percentage of fragmentation to start active
defrag",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "1-100",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "active-defrag-threshold-upper",
    "ParameterValue": "100",
    "Description": "Maximum percentage of fragmentation at which we use
maximum effort",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "1-100",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "activeresharding",
    "ParameterValue": "yes",
    "Description": "Apply rehashing or not.",
    "Source": "system",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "IsModifiable": false,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "requires-reboot"
  },
  {
    "ParameterName": "appendfsync",
    "ParameterValue": "everysec",
    "Description": "fsync policy for AOF persistence",
    "Source": "system",
    "DataType": "string",

```

```
    "AllowedValues": "always, everysec, no",
    "IsModifiable": false,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "appendonly",
    "ParameterValue": "no",
    "Description": "Enable Redis persistence.",
    "Source": "system",
    "DataType": "string",
    "AllowedValues": "yes, no",
    "IsModifiable": false,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "client-output-buffer-limit-normal-hard-limit",
    "ParameterValue": "0",
    "Description": "Normal client output buffer hard limit in bytes.",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "client-output-buffer-limit-normal-soft-limit",
    "ParameterValue": "0",
    "Description": "Normal client output buffer soft limit in bytes.",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "client-output-buffer-limit-normal-soft-seconds",
    "ParameterValue": "0",
    "Description": "Normal client output buffer soft limit in seconds.",
    "Source": "system",
    "DataType": "integer",
```



```
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "client-output-buffer-limit-pubsub-hard-limit",
    "ParameterValue": "33554432",
    "Description": "Pubsub client output buffer hard limit in bytes.",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "client-output-buffer-limit-pubsub-soft-limit",
    "ParameterValue": "8388608",
    "Description": "Pubsub client output buffer soft limit in bytes.",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "client-output-buffer-limit-pubsub-soft-seconds",
    "ParameterValue": "60",
    "Description": "Pubsub client output buffer soft limit in seconds.",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "client-output-buffer-limit-replica-soft-seconds",
    "ParameterValue": "60",
    "Description": "Replica client output buffer soft limit in
seconds.",
    "Source": "system",
```

```
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": false,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "client-query-buffer-limit",
    "ParameterValue": "1073741824",
    "Description": "Max size of a single client query buffer",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "1048576-1073741824",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "close-on-replica-write",
    "ParameterValue": "yes",
    "Description": "If enabled, clients who attempt to write to a read-
only replica will be disconnected. Applicable to 2.8.23 and higher.",
    "Source": "system",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "cluster-enabled",
    "ParameterValue": "no",
    "Description": "Enable cluster mode",
    "Source": "system",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "IsModifiable": false,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "requires-reboot"
  },
  {
    "ParameterName": "cluster-require-full-coverage",
    "ParameterValue": "no",
```

```

        "Description": "Whether cluster becomes unavailable if one or more
slots are not covered",
        "Source": "system",
        "DataType": "string",
        "AllowedValues": "yes,no",
        "IsModifiable": true,
        "MinimumEngineVersion": "5.0.0",
        "ChangeType": "immediate"
    },
    {
        "ParameterName": "databases",
        "ParameterValue": "16",
        "Description": "Set the number of databases.",
        "Source": "system",
        "DataType": "integer",
        "AllowedValues": "1-1200000",
        "IsModifiable": false,
        "MinimumEngineVersion": "5.0.0",
        "ChangeType": "requires-reboot"
    },
    {
        "ParameterName": "hash-max-ziplist-entries",
        "ParameterValue": "512",
        "Description": "The maximum number of hash entries in order for the
dataset to be compressed.",
        "Source": "system",
        "DataType": "integer",
        "AllowedValues": "0-",
        "IsModifiable": true,
        "MinimumEngineVersion": "5.0.0",
        "ChangeType": "immediate"
    },
    {
        "ParameterName": "hash-max-ziplist-value",
        "ParameterValue": "64",
        "Description": "The threshold of biggest hash entries in order for
the dataset to be compressed.",
        "Source": "system",
        "DataType": "integer",
        "AllowedValues": "0-",
        "IsModifiable": true,
        "MinimumEngineVersion": "5.0.0",
        "ChangeType": "immediate"
    },
    },

```

```
{
  "ParameterName": "hll-sparse-max-bytes",
  "ParameterValue": "3000",
  "Description": "HyperLogLog sparse representation bytes limit",
  "Source": "system",
  "DataType": "integer",
  "AllowedValues": "1-16000",
  "IsModifiable": true,
  "MinimumEngineVersion": "5.0.0",
  "ChangeType": "immediate"
},
{
  "ParameterName": "lazyfree-lazy-eviction",
  "ParameterValue": "no",
  "Description": "Perform an asynchronous delete on evictions",
  "Source": "system",
  "DataType": "string",
  "AllowedValues": "yes,no",
  "IsModifiable": true,
  "MinimumEngineVersion": "5.0.0",
  "ChangeType": "immediate"
},
{
  "ParameterName": "lazyfree-lazy-expire",
  "ParameterValue": "no",
  "Description": "Perform an asynchronous delete on expired keys",
  "Source": "system",
  "DataType": "string",
  "AllowedValues": "yes,no",
  "IsModifiable": true,
  "MinimumEngineVersion": "5.0.0",
  "ChangeType": "immediate"
},
{
  "ParameterName": "lazyfree-lazy-server-del",
  "ParameterValue": "no",
  "Description": "Perform an asynchronous delete on key updates",
  "Source": "system",
  "DataType": "string",
  "AllowedValues": "yes,no",
  "IsModifiable": true,
  "MinimumEngineVersion": "5.0.0",
  "ChangeType": "immediate"
},
```

```
{
  "ParameterName": "lfu-decay-time",
  "ParameterValue": "1",
  "Description": "The amount of time in minutes to decrement the key
counter for LFU eviction policy",
  "Source": "system",
  "DataType": "integer",
  "AllowedValues": "0-",
  "IsModifiable": true,
  "MinimumEngineVersion": "5.0.0",
  "ChangeType": "immediate"
},
{
  "ParameterName": "lfu-log-factor",
  "ParameterValue": "10",
  "Description": "The log factor for incrementing key counter for LFU
eviction policy",
  "Source": "system",
  "DataType": "integer",
  "AllowedValues": "1-",
  "IsModifiable": true,
  "MinimumEngineVersion": "5.0.0",
  "ChangeType": "immediate"
},
{
  "ParameterName": "list-compress-depth",
  "ParameterValue": "0",
  "Description": "Number of quicklist ziplist nodes from each side
of the list to exclude from compression. The head and tail of the list are always
uncompressed for fast push/pop operations",
  "Source": "system",
  "DataType": "integer",
  "AllowedValues": "0-",
  "IsModifiable": true,
  "MinimumEngineVersion": "5.0.0",
  "ChangeType": "immediate"
},
{
  "ParameterName": "list-max-ziplist-size",
  "ParameterValue": "-2",
  "Description": "The number of entries allowed per internal list node
can be specified as a fixed maximum size or a maximum number of elements",
  "Source": "system",
  "DataType": "integer",
```

```
    "AllowedValues": "-5,-4,-3,-2,-1,1-",
    "IsModifiable": false,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "lua-replicate-commands",
    "ParameterValue": "yes",
    "Description": "Always enable Lua effect replication or not",
    "Source": "system",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "lua-time-limit",
    "ParameterValue": "5000",
    "Description": "Max execution time of a Lua script in milliseconds.
0 for unlimited execution without warnings.",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "5000",
    "IsModifiable": false,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "maxclients",
    "ParameterValue": "65000",
    "Description": "The maximum number of Redis clients.",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "1-65000",
    "IsModifiable": false,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "requires-reboot"
  },
  {
    "ParameterName": "maxmemory-policy",
    "ParameterValue": "volatile-lru",
    "Description": "Max memory policy.",
    "Source": "system",
```

```

        "DataType": "string",
        "AllowedValues": "volatile-lru,allkeys-lru,volatile-lfu,allkeys-
lfu,volatile-random,allkeys-random,volatile-ttl,noeviction",
        "IsModifiable": true,
        "MinimumEngineVersion": "5.0.0",
        "ChangeType": "immediate"
    },
    {
        "ParameterName": "maxmemory-samples",
        "ParameterValue": "3",
        "Description": "Max memory samples.",
        "Source": "system",
        "DataType": "integer",
        "AllowedValues": "1-",
        "IsModifiable": true,
        "MinimumEngineVersion": "5.0.0",
        "ChangeType": "immediate"
    },
    {
        "ParameterName": "min-replicas-max-lag",
        "ParameterValue": "10",
        "Description": "The maximum amount of replica lag in seconds beyond
which the master would stop taking writes. A value of 0 means the master always
takes writes.",
        "Source": "system",
        "DataType": "integer",
        "AllowedValues": "0-",
        "IsModifiable": true,
        "MinimumEngineVersion": "5.0.0",
        "ChangeType": "immediate"
    },
    {
        "ParameterName": "min-replicas-to-write",
        "ParameterValue": "0",
        "Description": "The minimum number of replicas that must be present
with lag no greater than min-replicas-max-lag for master to take writes. Setting
this to 0 means the master always takes writes.",
        "Source": "system",
        "DataType": "integer",
        "AllowedValues": "0-",
        "IsModifiable": true,
        "MinimumEngineVersion": "5.0.0",
        "ChangeType": "immediate"
    },

```

```

    {
      "ParameterName": "notify-keyspace-events",
      "Description": "The keyspace events for Redis to notify Pub/Sub
clients about. By default all notifications are disabled",
      "Source": "system",
      "DataType": "string",
      "IsModifiable": true,
      "MinimumEngineVersion": "5.0.0",
      "ChangeType": "immediate"
    },
    {
      "ParameterName": "proto-max-bulk-len",
      "ParameterValue": "536870912",
      "Description": "Max size of a single element request",
      "Source": "system",
      "DataType": "integer",
      "AllowedValues": "1048576-536870912",
      "IsModifiable": true,
      "MinimumEngineVersion": "5.0.0",
      "ChangeType": "immediate"
    },
    {
      "ParameterName": "rename-commands",
      "ParameterValue": "",
      "Description": "Redis commands that can be dynamically renamed by
the customer",
      "Source": "system",
      "DataType": "string",
      "AllowedValues":
"APPEND,BITCOUNT,BITFIELD,BITOP,BITPOS,BLPOP,BRPOP,BRPOPLUSH,BZPOPMIN,BZPOPMAX,CLIENT,COMM
      "IsModifiable": true,
      "MinimumEngineVersion": "5.0.3",
      "ChangeType": "immediate"
    },
    {
      "ParameterName": "repl-backlog-size",
      "ParameterValue": "1048576",
      "Description": "The replication backlog size in bytes for PSYNC.
This is the size of the buffer which accumulates slave data when slave is
disconnected for some time, so that when slave reconnects again, only transfer the
portion of data which the slave missed. Minimum value is 16K.",
      "Source": "system",
      "DataType": "integer",
      "AllowedValues": "16384-",

```



```
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "repl-backlog-ttl",
    "ParameterValue": "3600",
    "Description": "The amount of time in seconds after the master no
longer have any slaves connected for the master to free the replication backlog. A
value of 0 means to never release the backlog.",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "replica-allow-chaining",
    "ParameterValue": "no",
    "Description": "Configures if chaining of replicas is allowed",
    "Source": "system",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "IsModifiable": false,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "replica-ignore-maxmemory",
    "ParameterValue": "yes",
    "Description": "Determines if replica ignores maxmemory setting by
not evicting items independent from the master",
    "Source": "system",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "IsModifiable": false,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "replica-lazy-flush",
    "ParameterValue": "no",
```

```

sync",
    "Description": "Perform an asynchronous flushDB during replica
",
    "Source": "system",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "IsModifiable": false,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
},
{
    "ParameterName": "reserved-memory-percent",
    "ParameterValue": "25",
    "Description": "The percent of memory reserved for non-cache memory
usage. You may want to increase this parameter for nodes with read replicas, AOF
enabled, etc, to reduce swap usage.",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "0-100",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
},
{
    "ParameterName": "set-max-intset-entries",
    "ParameterValue": "512",
    "Description": "The limit in the size of the set in order for the
dataset to be compressed.",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
},
{
    "ParameterName": "slowlog-log-slower-than",
    "ParameterValue": "10000",
    "Description": "The execution time, in microseconds, to exceed in
order for the command to get logged. Note that a negative number disables the slow
log, while a value of zero forces the logging of every command.",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "-",
    "IsModifiable": true,

```

```
        "MinimumEngineVersion": "5.0.0",
        "ChangeType": "immediate"
    },
    {
        "ParameterName": "slowlog-max-len",
        "ParameterValue": "128",
        "Description": "The length of the slow log. There is no limit to
this length. Just be aware that it will consume memory. You can reclaim memory used
by the slow log with SLOWLOG RESET.",
        "Source": "system",
        "DataType": "integer",
        "AllowedValues": "0-",
        "IsModifiable": true,
        "MinimumEngineVersion": "5.0.0",
        "ChangeType": "immediate"
    },
    {
        "ParameterName": "stream-node-max-bytes",
        "ParameterValue": "4096",
        "Description": "The maximum size of a single node in a stream in
bytes",
        "Source": "system",
        "DataType": "integer",
        "AllowedValues": "0-",
        "IsModifiable": true,
        "MinimumEngineVersion": "5.0.0",
        "ChangeType": "immediate"
    },
    {
        "ParameterName": "stream-node-max-entries",
        "ParameterValue": "100",
        "Description": "The maximum number of items a single node in a
stream can contain",
        "Source": "system",
        "DataType": "integer",
        "AllowedValues": "0-",
        "IsModifiable": true,
        "MinimumEngineVersion": "5.0.0",
        "ChangeType": "immediate"
    },
    {
        "ParameterName": "tcp-keepalive",
        "ParameterValue": "300",
```

```

    "Description": "If non-zero, send ACKs every given number of
seconds.",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "timeout",
    "ParameterValue": "0",
    "Description": "Close connection if client is idle for a given
number of seconds, or never if 0.",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "0,20-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "zset-max-ziplist-entries",
    "ParameterValue": "128",
    "Description": "The maximum number of sorted set entries in order
for the dataset to be compressed.",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  },
  {
    "ParameterName": "zset-max-ziplist-value",
    "ParameterValue": "64",
    "Description": "The threshold of biggest sorted set entries in order
for the dataset to be compressed.",
    "Source": "system",
    "DataType": "integer",
    "AllowedValues": "0-",
    "IsModifiable": true,
    "MinimumEngineVersion": "5.0.0",
    "ChangeType": "immediate"
  }

```

```

    }
  ]
}

```

- For API details, see [DescribeEngineDefaultParameters](#) in *AWS CLI Command Reference*.

describe-events

The following code example shows how to use describe-events.

AWS CLI

To describe events of a replication group

The following describe-events example returns a list of events for a replication group.

```

aws elasticache describe-events \
  --source-identifier test-cluster \
  --source-type replication-group

```

Output:

```

{
  "Events": [
    {
      "SourceIdentifier": "test-cluster",
      "SourceType": "replication-group",
      "Message": "Automatic failover has been turned on for replication group
test-cluster",
      "Date": "2020-03-18T23:51:34.457Z"
    },
    {
      "SourceIdentifier": "test-cluster",
      "SourceType": "replication-group",
      "Message": "Replication group test-cluster created",
      "Date": "2020-03-18T23:50:31.378Z"
    }
  ]
}

```

For more information, see [Monitoring Events](#) in the *Elasticache User Guide*.

- For API details, see [DescribeEvents](#) in *AWS CLI Command Reference*.

describe-global-replication-groups

The following code example shows how to use `describe-global-replication-groups`.

AWS CLI

To describe global replication groups

The following `describe-global-replication-groups` example returns details of a Global datastore.

```
aws elasticache describe-global-replication-groups \  
  --global-replication-group-id my-grg
```

Output:

```
{  
  "GlobalReplicationGroups": [  
    {  
      "GlobalReplicationGroupId": "my-grg",  
      "GlobalReplicationGroupDescription": "my-grg",  
      "Status": "creating",  
      "CacheNodeType": "cache.r5.large",  
      "Engine": "redis",  
      "EngineVersion": "5.0.6",  
      "ClusterEnabled": false,  
      "AuthTokenEnabled": false,  
      "TransitEncryptionEnabled": false,  
      "AtRestEncryptionEnabled": false  
    }  
  ]  
}
```

For more information, see [Replication Across AWS Regions Using Global Datastore](#) in the *Elasticache User Guide*.

- For API details, see [DescribeGlobalReplicationGroups](#) in *AWS CLI Command Reference*.

describe-replication-groups

The following code example shows how to use describe-replication-groups.

AWS CLI

To return a list of replication group details

The following describe-replication-groups example returns the replication groups.

```
aws elasticache describe-replication-groups
```

Output:

```
{
  "ReplicationGroups": [
    {
      "ReplicationGroupId": "my-cluster",
      "Description": "mycluster",
      "Status": "available",
      "PendingModifiedValues": {},
      "MemberClusters": [
        "pat-cluster-001",
        "pat-cluster-002",
        "pat-cluster-003",
        "pat-cluster-004"
      ],
      "NodeGroups": [
        {
          "NodeGroupId": "0001",
          "Status": "available",
          "PrimaryEndpoint": {
            "Address": "my-
cluster.xxxxih.ng.0001.usw2.cache.amazonaws.com",
            "Port": 6379
          },
          "ReaderEndpoint": {
            "Address": "my-cluster-
ro.xxxxih.ng.0001.usw2.cache.amazonaws.com",
            "Port": 6379
          },
          "NodeGroupMembers": [
            {
              "CacheClusterId": "my-cluster-001",
```

```
        "CacheNodeId": "0001",
        "ReadEndpoint": {
            "Address": "pat-
cluster-001.xxxih.0001.usw2.cache.amazonaws.com",
            "Port": 6379
        },
        "PreferredAvailabilityZone": "us-west-2a",
        "CurrentRole": "primary"
    },
    {
        "CacheClusterId": "my-cluster-002",
        "CacheNodeId": "0001",
        "ReadEndpoint": {
            "Address": "pat-
cluster-002.xxxxih.0001.usw2.cache.amazonaws.com",
            "Port": 6379
        },
        "PreferredAvailabilityZone": "us-west-2a",
        "CurrentRole": "replica"
    },
    {
        "CacheClusterId": "my-cluster-003",
        "CacheNodeId": "0001",
        "ReadEndpoint": {
            "Address": "pat-
cluster-003.xxxxih.0001.usw2.cache.amazonaws.com",
            "Port": 6379
        },
        "PreferredAvailabilityZone": "us-west-2a",
        "CurrentRole": "replica"
    },
    {
        "CacheClusterId": "my-cluster-004",
        "CacheNodeId": "0001",
        "ReadEndpoint": {
            "Address": "pat-
cluster-004.xxxih.0001.usw2.cache.amazonaws.com",
            "Port": 6379
        },
        "PreferredAvailabilityZone": "us-west-2a",
        "CurrentRole": "replica"
    }
]
}
```



```

    ],
    "AutomaticFailover": "disabled",
    "SnapshotRetentionLimit": 0,
    "SnapshotWindow": "07:30-08:30",
    "ClusterEnabled": false,
    "CacheNodeType": "cache.r5.xlarge",
    "AuthTokenEnabled": false,
    "TransitEncryptionEnabled": false,
    "AtRestEncryptionEnabled": false,
    "ARN": "arn:aws:elasticache:us-
west-2:xxxxxxxxxxx152:replicationgroup:my-cluster",
    "LogDeliveryConfigurations": [
      {
        "LogType": "slow-log",
        "DestinationType": "cloudwatch-logs",
        "DestinationDetails": {
          "CloudWatchLogsDetails": {
            "LogGroup": "test-log"
          }
        },
        "LogFormat": "json",
        "Status": "active"
      }
    ]
  }
]
}

```

For more information, see [Managing Clusters](#) in the *Elasticache User Guide*.

- For API details, see [DescribeReplicationGroups](#) in *AWS CLI Command Reference*.

describe-reserved-cache-nodes-offerings

The following code example shows how to use `describe-reserved-cache-nodes-offerings`.

AWS CLI

To describe reserved-cache-nodes-offerings

The following `describe-reserved-cache-nodes-offerings` example returns details of a reserved-cache-node options.

```
aws elasticache describe-reserved-cache-nodes-offerings
```

Output:

```
{
  "ReservedCacheNodesOfferings": [
    {
      "ReservedCacheNodesOfferingId": "01ce0a19-a476-41cb-8aee-48eacbcd8e5",
      "CacheNodeType": "cache.t3.small",
      "Duration": 31536000,
      "FixedPrice": 97.0,
      "UsagePrice": 0.0,
      "ProductDescription": "memcached",
      "OfferingType": "Partial Upfront",
      "RecurringCharges": [
        {
          "RecurringChargeAmount": 0.011,
          "RecurringChargeFrequency": "Hourly"
        }
      ]
    },
    {
      "ReservedCacheNodesOfferingId": "0443a27b-4da5-4b90-b92d-929fbd7abed2",
      "CacheNodeType": "cache.m3.2xlarge",
      "Duration": 31536000,
      "FixedPrice": 1772.0,
      "UsagePrice": 0.0,
      "ProductDescription": "redis",
      "OfferingType": "Heavy Utilization",
      "RecurringCharges": [
        {
          "RecurringChargeAmount": 0.25,
          "RecurringChargeFrequency": "Hourly"
        }
      ]
    },
    ...
  ]
}
```

For more information, see [Getting Info About Reserved Node Offerings](#) in the *Elasticache Redis User Guide* or [Getting Info About Reserved Node Offerings](#) in the *Elasticache Memcached User Guide*.

- For API details, see [DescribeReservedCacheNodesOfferings](#) in *AWS CLI Command Reference*.

describe-reserved-cache-nodes

The following code example shows how to use `describe-reserved-cache-nodes`.

AWS CLI

To describe reserved cache nodes

The following `describe-reserved-cache-nodes` example returns information about reserved cache nodes for this account, or about the specified reserved cache node.

```
aws elasticache describe-reserved-cache-nodes
```

Output:

```
{
  "ReservedCacheNodes": [
    {
      "ReservedCacheNodeId": "mynode",
      "ReservedCacheNodesOfferingId": "xxxxxxxxx-xxxxx-xxxxx-xxxx-xxxxxxxxx71",
      "CacheNodeType": "cache.t3.small",
      "StartTime": "2019-12-06T02:50:44.003Z",
      "Duration": 31536000,
      "FixedPrice": 0.0,
      "UsagePrice": 0.0,
      "CacheNodeCount": 1,
      "ProductDescription": "redis",
      "OfferingType": "No Upfront",
      "State": "payment-pending",
      "RecurringCharges": [
        {
          "RecurringChargeAmount": 0.023,
          "RecurringChargeFrequency": "Hourly"
        }
      ],
      "ReservationARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxxxxx52:reserved-instance:mynode"
    }
  ]
}
```

```

    }
  ]
}

```

For more information, see [Managing Costs with Reserved Nodes](#) in the *Elasticache User Guide*.

- For API details, see [DescribeReservedCacheNodes](#) in *AWS CLI Command Reference*.

describe-service-updates

The following code example shows how to use describe-service-updates.

AWS CLI

To describe service updates

The following describe-service-updates example returns details about service updates.

```
aws elasticache describe-service-updates
```

Output:

```

{
  "ServiceUpdates": [
    {
      "ServiceUpdateName": "elc-xxxxxxx7-001",
      "ServiceUpdateReleaseDate": "2019-10-09T16:00:00Z",
      "ServiceUpdateEndDate": "2020-02-09T15:59:59Z",
      "ServiceUpdateSeverity": "important",
      "ServiceUpdateRecommendedApplyByDate": "2019-11-08T15:59:59Z",
      "ServiceUpdateStatus": "available",
      "ServiceUpdateDescription": "Upgrades to improve the security,
reliability, and operational performance of your ElastiCache nodes",
      "ServiceUpdateType": "security-update",
      "Engine": "redis, memcached",
      "EngineVersion": "redis 2.6.13 and onwards, memcached 1.4.5 and
onwards",
      "AutoUpdateAfterRecommendedApplyByDate": false,
      "EstimatedUpdateTime": "30 minutes per node"
    },
    {
      "ServiceUpdateName": "elc-xxxxxxx4-001",
      "ServiceUpdateReleaseDate": "2019-06-11T15:00:00Z",

```

```

    "ServiceUpdateEndDate": "2019-10-01T09:24:00Z",
    "ServiceUpdateSeverity": "important",
    "ServiceUpdateRecommendedApplyByDate": "2019-07-11T14:59:59Z",
    "ServiceUpdateStatus": "expired",
    "ServiceUpdateDescription": "Upgrades to improve the security,
reliability, and operational performance of your ElastiCache nodes",
    "ServiceUpdateType": "security-update",
    "Engine": "redis",
    "EngineVersion": "redis 3.2.6, redis 4.0 and onwards",
    "AutoUpdateAfterRecommendedApplyByDate": false,
    "EstimatedUpdateTime": "30 minutes per node"
  }
]
}

```

- For API details, see [DescribeServiceUpdates](#) in *AWS CLI Command Reference*.

describe-snapshots

The following code example shows how to use describe-snapshots.

AWS CLI

To describe snapshots

The following "describe-snapshots" example returns information about your cluster or replication group snapshots.

```
aws elasticache describe-snapshots
```

Output:

```

{
  "Snapshots": [
    {
      "SnapshotName": "automatic.my-cluster2-002-2019-12-05-06-38",
      "CacheClusterId": "my-cluster2-002",
      "SnapshotStatus": "available",
      "SnapshotSource": "automated",
      "CacheNodeType": "cache.r5.large",
      "Engine": "redis",
      "EngineVersion": "5.0.5",
    }
  ]
}

```

```

    "NumCacheNodes": 1,
    "PreferredAvailabilityZone": "us-west-2a",
    "CacheClusterCreateTime": "2019-11-26T01:22:52.396Z",
    "PreferredMaintenanceWindow": "mon:17:30-mon:18:30",
    "TopicArn": "arn:aws:sns:us-west-2:xxxxxxxxx52:My_Topic",
    "Port": 6379,
    "CacheParameterGroupName": "default.redis5.0",
    "CacheSubnetGroupName": "kxkxk",
    "VpcId": "vpc-a3e97cdb",
    "AutoMinorVersionUpgrade": true,
    "SnapshotRetentionLimit": 1,
    "SnapshotWindow": "06:30-07:30",
    "NodeSnapshots": [
      {
        "CacheNodeId": "0001",
        "CacheSize": "5 MB",
        "CacheNodeCreateTime": "2019-11-26T01:22:52.396Z",
        "SnapshotCreateTime": "2019-12-05T06:38:23Z"
      }
    ]
  },
  {
    "SnapshotName": "myreplica-backup",
    "CacheClusterId": "myreplica",
    "SnapshotStatus": "available",
    "SnapshotSource": "manual",
    "CacheNodeType": "cache.r5.large",
    "Engine": "redis",
    "EngineVersion": "5.0.5",
    "NumCacheNodes": 1,
    "PreferredAvailabilityZone": "us-west-2a",
    "CacheClusterCreateTime": "2019-11-26T00:14:52.439Z",
    "PreferredMaintenanceWindow": "sat:10:00-sat:11:00",
    "TopicArn": "arn:aws:sns:us-west-2:xxxxxxxxx152:My_Topic",
    "Port": 6379,
    "CacheParameterGroupName": "default.redis5.0",
    "CacheSubnetGroupName": "kxkxk",
    "VpcId": "vpc-a3e97cdb",
    "AutoMinorVersionUpgrade": true,
    "SnapshotRetentionLimit": 0,
    "SnapshotWindow": "09:00-10:00",
    "NodeSnapshots": [
      {
        "CacheNodeId": "0001",

```

```

        "CacheSize": "5 MB",
        "CacheNodeCreateTime": "2019-11-26T00:14:52.439Z",
        "SnapshotCreateTime": "2019-11-26T00:25:01Z"
    }
]
},
{
    "SnapshotName": "my-cluster",
    "CacheClusterId": "my-cluster-003",
    "SnapshotStatus": "available",
    "SnapshotSource": "manual",
    "CacheNodeType": "cache.r5.large",
    "Engine": "redis",
    "EngineVersion": "5.0.5",
    "NumCacheNodes": 1,
    "PreferredAvailabilityZone": "us-west-2a",
    "CacheClusterCreateTime": "2019-11-25T23:56:17.186Z",
    "PreferredMaintenanceWindow": "sat:10:00-sat:11:00",
    "TopicArn": "arn:aws:sns:us-west-2:xxxxxxxxxx152:My_Topic",
    "Port": 6379,
    "CacheParameterGroupName": "default.redis5.0",
    "CacheSubnetGroupName": "kxxkxk",
    "VpcId": "vpc-a3e97cdb",
    "AutoMinorVersionUpgrade": true,
    "SnapshotRetentionLimit": 0,
    "SnapshotWindow": "09:00-10:00",
    "NodeSnapshots": [
        {
            "CacheNodeId": "0001",
            "CacheSize": "5 MB",
            "CacheNodeCreateTime": "2019-11-25T23:56:17.186Z",
            "SnapshotCreateTime": "2019-11-26T03:08:33Z"
        }
    ]
}
]
}
}

```

For more information, see [Backup and Restore for ElastiCache for Redis](#) in the *ElastiCache User Guide*.

- For API details, see [DescribeSnapshots](#) in *AWS CLI Command Reference*.

describe-update-actions

The following code example shows how to use describe-update-actions.

AWS CLI

To describe update actions

The following describe-update-actions example returns details of update actions.

```
aws elasticache describe-update-actions
```

Output:

```
{
  "UpdateActions": [
    {
      "ReplicationGroupId": "mycluster",
      "ServiceUpdateName": "elc-20191007-001",
      "ServiceUpdateReleaseDate": "2019-10-09T16:00:00Z",
      "ServiceUpdateSeverity": "important",
      "ServiceUpdateStatus": "available",
      "ServiceUpdateRecommendedApplyByDate": "2019-11-08T15:59:59Z",
      "ServiceUpdateType": "security-update",
      "UpdateActionAvailableDate": "2019-12-05T19:15:19.995Z",
      "UpdateActionStatus": "complete",
      "NodesUpdated": "9/9",
      "UpdateActionStatusModifiedDate": "2019-12-05T19:15:20.461Z",
      "SlaMet": "n/a",
      "Engine": "redis"
    },
    {
      "CacheClusterId": "my-memcached-cluster",
      "ServiceUpdateName": "elc-20191007-001",
      "ServiceUpdateReleaseDate": "2019-10-09T16:00:00Z",
      "ServiceUpdateSeverity": "important",
      "ServiceUpdateStatus": "available",
      "ServiceUpdateRecommendedApplyByDate": "2019-11-08T15:59:59Z",
      "ServiceUpdateType": "security-update",
      "UpdateActionAvailableDate": "2019-12-04T18:26:05.349Z",
      "UpdateActionStatus": "complete",
      "NodesUpdated": "1/1",
      "UpdateActionStatusModifiedDate": "2019-12-04T18:26:05.352Z",
      "SlaMet": "n/a",

```



```

    "Engine": "redis"
  },
  {
    "ReplicationGroupId": "my-cluster",
    "ServiceUpdateName": "elc-20191007-001",
    "ServiceUpdateReleaseDate": "2019-10-09T16:00:00Z",
    "ServiceUpdateSeverity": "important",
    "ServiceUpdateStatus": "available",
    "ServiceUpdateRecommendedApplyByDate": "2019-11-08T15:59:59Z",
    "ServiceUpdateType": "security-update",
    "UpdateActionAvailableDate": "2019-11-26T03:36:26.320Z",
    "UpdateActionStatus": "complete",
    "NodesUpdated": "4/4",
    "UpdateActionStatusModifiedDate": "2019-12-04T22:11:12.664Z",
    "SlaMet": "n/a",
    "Engine": "redis"
  },
  {
    "ReplicationGroupId": "my-cluster2",
    "ServiceUpdateName": "elc-20191007-001",
    "ServiceUpdateReleaseDate": "2019-10-09T16:00:00Z",
    "ServiceUpdateSeverity": "important",
    "ServiceUpdateStatus": "available",
    "ServiceUpdateRecommendedApplyByDate": "2019-11-08T15:59:59Z",
    "ServiceUpdateType": "security-update",
    "UpdateActionAvailableDate": "2019-11-26T01:26:01.617Z",
    "UpdateActionStatus": "complete",
    "NodesUpdated": "3/3",
    "UpdateActionStatusModifiedDate": "2019-11-26T01:26:01.753Z",
    "SlaMet": "n/a",
    "Engine": "redis"
  }
]
}

```

For more information, see [Self-Service Updates in Amazon ElastiCache](#) in the *ElastiCache User Guide*.

- For API details, see [DescribeUpdateActions](#) in *AWS CLI Command Reference*.

describe-user-groups

The following code example shows how to use describe-user-groups.

AWS CLI

To describe user-groups

The following describe-user-groups example returns a list of user groups.

```
aws elasticache describe-user-groups
```

Output:

```
{
  "UserGroups": [
    {
      "UserGroupId": "myusergroup",
      "Status": "active",
      "Engine": "redis",
      "UserIds": [
        "default"
      ],
      "ReplicationGroups": [],
      "ARN": "arn:aws:elasticache:us-
west-2:xxxxxxxxxx52:usergroup:myusergroup"
    }
  ]
}
```

For more information, see [Authenticating Users with Role-Based Access Control \(RBAC\)](#) in the *Elasticache User Guide*.

- For API details, see [DescribeUserGroups](#) in *AWS CLI Command Reference*.

describe-users

The following code example shows how to use describe-users.

AWS CLI

To describe users

The following describe-users example returns a list of users.

```
aws elasticache describe-users
```

Output:

```
{
  "Users": [
    {
      "UserId": "default",
      "UserName": "default",
      "Status": "active",
      "Engine": "redis",
      "AccessString": "on ~* +@all",
      "UserGroupIds": [
        "myusergroup"
      ],
      "Authentication": {
        "Type": "no-password"
      },
      "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxx52:user:default"
    },
    {
      "UserId": "user1",
      "UserName": "myUser",
      "Status": "active",
      "Engine": "redis",
      "AccessString": "on ~* +@all",
      "UserGroupIds": [],
      "Authentication": {
        "Type": "password",
        "PasswordCount": 1
      },
      "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxx52:user:user1"
    },
    {
      "UserId": "user2",
      "UserName": "myUser",
      "Status": "active",
      "Engine": "redis",
      "AccessString": "on ~app:* -@all +@read +@hash +@bitmap +@geo -setbit -
bitfield -hset -hsetnx -hmset -hincrby -hincrbyfloat -hdel -bitop -geoadd -georadius
-georadiusbymember",
      "UserGroupIds": [],
      "Authentication": {
        "Type": "password",
        "PasswordCount": 1
      },
    },
  ],
}
```

```

        "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxx52:user:user2"
    }
]
}

```

For more information, see [Authenticating Users with Role-Based Access Control \(RBAC\)](#) in the *Elasticache User Guide*.

- For API details, see [DescribeUsers](#) in *AWS CLI Command Reference*.

disassociate-global-replication-group

The following code example shows how to use `disassociate-global-replication-group`.

AWS CLI

To diassociate a secondary cluster from a global replication group

The following `disassociate-global-replication-group` example removes a secondary cluster from a Global datastore

```

aws elasticache disassociate-global-replication-group \
  --global-replication-group-id my-grg \
  --replication-group-id my-cluster-grg-secondary \
  --replication-group-region us-east-1

```

Output:

```

{
  "GlobalReplicationGroup": {
    "GlobalReplicationGroupId": "my-grg",
    "GlobalReplicationGroupDescription": "my-grg",
    "Status": "modifying",
    "CacheNodeType": "cache.r5.large",
    "Engine": "redis",
    "EngineVersion": "5.0.6",
    "Members": [
      {
        "ReplicationGroupId": "my-cluster-grg-secondary",
        "ReplicationGroupRegion": "us-east-1",
        "Role": "SECONDARY",
        "AutomaticFailover": "enabled",
        "Status": "associated"
      }
    ]
  }
}

```

```

    },
    {
      "ReplicationGroupId": "my-cluster-grg",
      "ReplicationGroupRegion": "us-west-2",
      "Role": "PRIMARY",
      "AutomaticFailover": "enabled",
      "Status": "associated"
    }
  ],
  "ClusterEnabled": false,
  "AuthTokenEnabled": false,
  "TransitEncryptionEnabled": false,
  "AtRestEncryptionEnabled": false
}
}

```

For more information, see [Replication Across AWS Regions Using Global Datastore](#) in the *Elasticache User Guide*.

- For API details, see [DisassociateGlobalReplicationGroup](#) in *AWS CLI Command Reference*.

increase-node-groups-in-global-replication-group

The following code example shows how to use `increase-node-groups-in-global-replication-group`.

AWS CLI

To increase the number of node groups in a global replication group

The following `increase-node-groups-in-global-replication-group` increases the node group count using the Redis engine.

```

aws elasticache increase-node-groups-in-global-replication-group \
  --global-replication-group-id sgauipat-test-4 \
  --node-group-count 6 \
  --apply-immediately

```

Output:

```

{
  "GlobalReplicationGroup": {

```

```
"GlobalReplicationGroupId": "sgaui-test-4",
"GlobalReplicationGroupDescription": "test-4",
"Status": "modifying",
"CacheNodeType": "cache.r5.large",
"Engine": "redis",
"EngineVersion": "5.0.6",
"Members": [
  {
    "ReplicationGroupId": "my-cluster-b",
    "ReplicationGroupRegion": "us-east-1",
    "Role": "SECONDARY",
    "AutomaticFailover": "enabled",
    "Status": "associated"
  },
  {
    "ReplicationGroupId": "my-cluster-a",
    "ReplicationGroupRegion": "us-west-2",
    "Role": "PRIMARY",
    "AutomaticFailover": "enabled",
    "Status": "associated"
  }
],
"ClusterEnabled": true,
"GlobalNodeGroups": [
  {
    "GlobalNodeGroupId": "sgaui-test-4-0001",
    "Slots": "0-234,2420-5461"
  },
  {
    "GlobalNodeGroupId": "sgaui-test-4-0002",
    "Slots": "5462-5904,6997-9830"
  },
  {
    "GlobalNodeGroupId": "sgaui-test-4-0003",
    "Slots": "10923-11190,13375-16383"
  },
  {
    "GlobalNodeGroupId": "sgaui-test-4-0004",
    "Slots": "235-2419,5905-6996"
  },
  {
    "GlobalNodeGroupId": "sgaui-test-4-0005",
    "Slots": "9831-10922,11191-13374"
  }
]
```

```
    ],
    "AuthTokenEnabled": false,
    "TransitEncryptionEnabled": false,
    "AtRestEncryptionEnabled": false
  }
}
```

For more information, see [Replication Across AWS Regions Using Global Datastore](#) in the *Elasticache User Guide*.

- For API details, see [IncreaseNodeGroupsInGlobalReplicationGroup](#) in *AWS CLI Command Reference*.

increase-replica-count

The following code example shows how to use `increase-replica-count`.

AWS CLI

To increase replica count

The following `increase-replica-count` example does one of two things. It can dynamically increase the number of replicas in a Redis (cluster mode disabled) replication group. Or it can dynamically increase the number of replica nodes in one or more node groups (shards) of a Redis (cluster mode enabled) replication group. This operation is performed with no cluster downtime.

```
aws elasticache increase-replica-count \
  --replication-group-id "my-cluster" \
  --apply-immediately \
  --new-replica-count 3
```

Output:

```
{
  "ReplicationGroup": {
    "ReplicationGroupId": "my-cluster",
    "Description": " ",
    "Status": "modifying",
    "PendingModifiedValues": {},
    "MemberClusters": [
      "my-cluster-001",
```

```
        "my-cluster-002",
        "my-cluster-003",
        "my-cluster-004"
    ],
    "NodeGroups": [
        {
            "NodeGroupId": "0001",
            "Status": "modifying",
            "PrimaryEndpoint": {
                "Address": "my-
cluster.xxxxxih.ng.0001.usw2.cache.amazonaws.com",
                "Port": 6379
            },
            "ReaderEndpoint": {
                "Address": "my-cluster-
ro.xxxxxxih.ng.0001.usw2.cache.amazonaws.com",
                "Port": 6379
            },
            "NodeGroupMembers": [
                {
                    "CacheClusterId": "my-cluster-001",
                    "CacheNodeId": "0001",
                    "ReadEndpoint": {
                        "Address": "my-
cluster-001.xxxxxih.0001.usw2.cache.amazonaws.com",
                        "Port": 6379
                    },
                    "PreferredAvailabilityZone": "us-west-2a",
                    "CurrentRole": "primary"
                },
                {
                    "CacheClusterId": "my-cluster-003",
                    "CacheNodeId": "0001",
                    "ReadEndpoint": {
                        "Address": "my-
cluster-003.xxxxxih.0001.usw2.cache.amazonaws.com",
                        "Port": 6379
                    },
                    "PreferredAvailabilityZone": "us-west-2a",
                    "CurrentRole": "replica"
                }
            ]
        }
    ],
    ],
```



```
    "AutomaticFailover": "disabled",
    "SnapshotRetentionLimit": 0,
    "SnapshotWindow": "07:30-08:30",
    "ClusterEnabled": false,
    "CacheNodeType": "cache.r5.xlarge",
    "TransitEncryptionEnabled": false,
    "AtRestEncryptionEnabled": false
  }
}
```

For more information, see [Increasing the Number of Replicas in a Shard](#) in the *Elasticache User Guide*.

- For API details, see [IncreaseReplicaCount](#) in *AWS CLI Command Reference*.

list-allowed-node-modifications

The following code example shows how to use `list-allowed-node-modifications`.

AWS CLI

To list the allowed node modifications

The following `list-allowed-node-type-modifications` example lists all the available node types that you can scale your Redis cluster's or replication group's current node type to.

```
aws elasticache list-allowed-node-type-modifications \
  --replication-group-id "my-replication-group"
```

Output:

```
{
  "ScaleUpModifications": [
    "cache.m5.12xlarge",
    "cache.m5.24xlarge",
    "cache.m5.4xlarge",
    "cache.r5.12xlarge",
    "cache.r5.24xlarge",
    "cache.r5.2xlarge",
    "cache.r5.4xlarge"
  ],
  "ScaleDownModifications": [
```

```
    "cache.m3.large",
    "cache.m3.medium",
    "cache.m3.xlarge",
    "cache.m4.large",
    "cache.m4.xlarge",
    "cache.m5.2xlarge",
    "cache.m5.large",
    "cache.m5.xlarge",
    "cache.r3.large",
    "cache.r4.large",
    "cache.r4.xlarge",
    "cache.r5.large",
    "cache.t2.medium",
    "cache.t2.micro",
    "cache.t2.small",
    "cache.t3.medium",
    "cache.t3.micro",
    "cache.t3.small"
  ]
}
```

For more information, see [Scaling ElastiCache for Redis Clusters](#) in the *ElastiCache User Guide*.

- For API details, see [ListAllowedNodeTypeModifications](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags for a resource

The following `list-tags-for-resource` example lists tags for a resource.

```
aws elasticache list-tags-for-resource \
  --resource-name "arn:aws:elasticache:us-east-1:123456789012:cluster:my-cluster"
```

Output:

```
{
  "TagList": [
    {
```

```
        "Key": "Project",
        "Value": "querySpeedUp"
    },
    {
        "Key": "Environment",
        "Value": "PROD"
    }
]
}
```

For more information, see [Listing Tags Using the AWS CLI](#) in the *ElastiCache User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

modify-cache-cluster

The following code example shows how to use `modify-cache-cluster`.

AWS CLI

To modify cache clusters

The following `modify-cache-cluster` example modifies the settings for the specified cluster.

```
aws elasticache modify-cache-cluster \
  --cache-cluster-id "my-cluster" \
  --num-cache-nodes 1
```

Output:

```
{
  "CacheCluster": {
    "CacheClusterId": "my-cluster",
    "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
    "CacheNodeType": "cache.m5.large",
    "Engine": "redis",
    "EngineVersion": "5.0.5",
    "CacheClusterStatus": "available",
    "NumCacheNodes": 1,
    "PreferredAvailabilityZone": "us-west-2c",
    "CacheClusterCreateTime": "2019-12-04T18:24:56.652Z",
    "PreferredMaintenanceWindow": "sat:10:00-sat:11:00",
```

```

    "PendingModifiedValues": {},
    "CacheSecurityGroups": [],
    "CacheParameterGroup": {
        "CacheParameterGroupName": "default.redis5.0",
        "ParameterApplyStatus": "in-sync",
        "CacheNodeIdsToReboot": []
    },
    "CacheSubnetGroupName": "default",
    "AutoMinorVersionUpgrade": true,
    "SnapshotRetentionLimit": 0,
    "SnapshotWindow": "07:00-08:00",
    "TransitEncryptionEnabled": false,
    "AtRestEncryptionEnabled": false
}
}

```

For more information, see [Modifying an ElastiCache Cluster](#) in the *ElastiCache User Guide*.

- For API details, see [ModifyCacheCluster](#) in *AWS CLI Command Reference*.

modify-cache-parameter-group

The following code example shows how to use `modify-cache-parameter-group`.

AWS CLI

To modify a cache parameter group

The following `modify-cache-parameter-group` example modifies the parameters of the specified cache parameter group.

```

aws elasticache modify-cache-parameter-group \
  --cache-parameter-group-name "mygroup" \
  --parameter-name-values "ParameterName=activedefrag, ParameterValue=no"

```

Output:

```

{
  "CacheParameterGroupName": "mygroup"
}

```

For more information, see [Modifying a Parameter Group](#) in the *ElastiCache User Guide*.

- For API details, see [ModifyCacheParameterGroup](#) in *AWS CLI Command Reference*.

modify-cache-subnet-group

The following code example shows how to use `modify-cache-subnet-group`.

AWS CLI

To modify a cache subnet group

The following `modify-cache-subnet-group` example modifies the specified cache subnet group.

```
aws elasticache modify-cache-subnet-group \  
  --cache-subnet-group-name kxkxk \  
  --cache-subnet-group-description "mygroup"
```

Output:

```
{  
  "CacheSubnetGroup": {  
    "CacheSubnetGroupName": "kxkxk",  
    "CacheSubnetGroupDescription": "mygroup",  
    "VpcId": "vpc-xxxxcdb",  
    "Subnets": [  
      {  
        "SubnetIdentifier": "subnet-xxxxbff",  
        "SubnetAvailabilityZone": {  
          "Name": "us-west-2a"  
        }  
      }  
    ]  
  }  
}
```

For more information, see [Modifying a Subnet Group](#) in the *Elasticache User Guide*.

- For API details, see [ModifyCacheSubnetGroup](#) in *AWS CLI Command Reference*.

modify-global-replication-group

The following code example shows how to use `modify-global-replication-group`.

AWS CLI

To modify a global replication group

The following `modify-global-replication-group` modifies the properties of a global replication group, in this case disabling automatic failover, using the Redis engine.

```
aws elasticache modify-global-replication-group \  
  --global-replication-group-id sgau1-pat-group \  
  --apply-immediately \  
  --no-automatic-failover-enabled
```

Output

```
{  
  "GlobalReplicationGroup": {  
    "GlobalReplicationGroupId": "sgau1-test-group",  
    "GlobalReplicationGroupDescription": " ",  
    "Status": "modifying",  
    "CacheNodeType": "cache.r5.large",  
    "Engine": "redis",  
    "EngineVersion": "5.0.6",  
    "ClusterEnabled": false,  
    "AuthTokenEnabled": false,  
    "TransitEncryptionEnabled": false,  
    "AtRestEncryptionEnabled": false  
  }  
}
```

For more information, see [Replication Across AWS Regions Using Global Datastore](#) in the *Elasticache User Guide*.

- For API details, see [ModifyGlobalReplicationGroup](#) in *AWS CLI Command Reference*.

modify-replication-group-shard-configuration

The following code example shows how to use `modify-replication-group-shard-configuration`.

AWS CLI

To modify a replication group shard configuration

The following `modify-replication-group-shard-configuration` decreases the node group count using the Redis engine.

```
aws elasticache modify-replication-group-shard-configuration \  
  --replication-group-id mycluster \  
  --node-group-count 3 \  
  --apply-immediately \  
  --node-groups-to-remove 0002
```

Output

```
{  
  "ReplicationGroup": {  
    "ReplicationGroupId": "mycluster",  
    "Description": "mycluster",  
    "GlobalReplicationGroupInfo": {},  
    "Status": "modifying",  
    "PendingModifiedValues": {},  
    "MemberClusters": [  
      "mycluster-0002-001",  
      "mycluster-0002-002",  
      "mycluster-0002-003",  
      "mycluster-0003-001",  
      "mycluster-0003-002",  
      "mycluster-0003-003",  
      "mycluster-0003-004",  
      "mycluster-0004-001",  
      "mycluster-0004-002",  
      "mycluster-0004-003",  
      "mycluster-0005-001",  
      "mycluster-0005-002",  
      "mycluster-0005-003"  
    ],  
    "NodeGroups": [  
      {  
        "NodeGroupId": "0002",  
        "Status": "modifying",  
        "Slots": "894-1767,3134-4443,5149-5461,6827-7332,12570-13662",  
        "NodeGroupMembers": [  
          {  
            "CacheClusterId": "mycluster-0002-001",  
            "CacheNodeId": "0001",  
            "PreferredAvailabilityZone": "us-west-2c"          }  
        ]  
      }  
    ]  
  }  
}
```

```
    },
    {
      "CacheClusterId": "mycluster-0002-002",
      "CacheNodeId": "0001",
      "PreferredAvailabilityZone": "us-west-2a"
    },
    {
      "CacheClusterId": "mycluster-0002-003",
      "CacheNodeId": "0001",
      "PreferredAvailabilityZone": "us-west-2b"
    }
  ]
},
{
  "NodeGroupId": "0003",
  "Status": "modifying",
  "Slots":
"0-324,5462-5692,6784-6826,7698-8191,10923-11075,12441-12569,13663-16383",
  "NodeGroupMembers": [
    {
      "CacheClusterId": "mycluster-0003-001",
      "CacheNodeId": "0001",
      "PreferredAvailabilityZone": "us-west-2c"
    },
    {
      "CacheClusterId": "mycluster-0003-002",
      "CacheNodeId": "0001",
      "PreferredAvailabilityZone": "us-west-2b"
    },
    {
      "CacheClusterId": "mycluster-0003-003",
      "CacheNodeId": "0001",
      "PreferredAvailabilityZone": "us-west-2a"
    },
    {
      "CacheClusterId": "mycluster-0003-004",
      "CacheNodeId": "0001",
      "PreferredAvailabilityZone": "us-west-2c"
    }
  ]
},
{
  "NodeGroupId": "0004",
  "Status": "modifying",
```



```
"Slots": "325-336,4706-5148,7333-7697,9012-10922,11076-12440",
"NodeGroupMembers": [
  {
    "CacheClusterId": "mycluster-0004-001",
    "CacheNodeId": "0001",
    "PreferredAvailabilityZone": "us-west-2b"
  },
  {
    "CacheClusterId": "mycluster-0004-002",
    "CacheNodeId": "0001",
    "PreferredAvailabilityZone": "us-west-2a"
  },
  {
    "CacheClusterId": "mycluster-0004-003",
    "CacheNodeId": "0001",
    "PreferredAvailabilityZone": "us-west-2c"
  }
]
},
{
  "NodeGroupId": "0005",
  "Status": "modifying",
  "Slots": "337-893,1768-3133,4444-4705,5693-6783,8192-9011",
  "NodeGroupMembers": [
    {
      "CacheClusterId": "mycluster-0005-001",
      "CacheNodeId": "0001",
      "PreferredAvailabilityZone": "us-west-2a"
    },
    {
      "CacheClusterId": "mycluster-0005-002",
      "CacheNodeId": "0001",
      "PreferredAvailabilityZone": "us-west-2c"
    },
    {
      "CacheClusterId": "mycluster-0005-003",
      "CacheNodeId": "0001",
      "PreferredAvailabilityZone": "us-west-2b"
    }
  ]
}
],
"AutomaticFailover": "enabled",
"MultiAZ": "enabled",
```

```

    "ConfigurationEndpoint": {
      "Address": "mycluster.g2xbih.clustercfg.usw2.cache.amazonaws.com",
      "Port": 6379
    },
    "SnapshotRetentionLimit": 1,
    "SnapshotWindow": "13:00-14:00",
    "ClusterEnabled": true,
    "CacheNodeType": "cache.r5.xlarge",
    "TransitEncryptionEnabled": false,
    "AtRestEncryptionEnabled": false
  }
}

```

For more information, see [Scaling ElastiCache for Redis Clusters](#) in the *ElastiCache User Guide*.

- For API details, see [ModifyReplicationGroupShardConfiguration](#) in *AWS CLI Command Reference*.

modify-replication-group

The following code example shows how to use `modify-replication-group`.

AWS CLI

To modify a replication group

The following `modify-replication-group` disables Multi-AZ using the Redis engine.

```

aws elasticache modify-replication-group \
  --replication-group-id test-cluster \
  --no-multi-az-enabled \
  --apply-immediately

```

Output

```

{
  "ReplicationGroup": {
    "ReplicationGroupId": "test-cluster",
    "Description": "test-cluster",
    "GlobalReplicationGroupInfo": {
      "GlobalReplicationGroupId": "sgaui-pat-group",
      "GlobalReplicationGroupMemberRole": "PRIMARY"
    }
  },
}

```

```
"Status": "available",
"PendingModifiedValues": {},
"MemberClusters": [
  "test-cluster-001",
  "test-cluster-002",
  "test-cluster-003"
],
"NodeGroups": [
  {
    "NodeGroupId": "0001",
    "Status": "available",
    "PrimaryEndpoint": {
      "Address": "test-
cluster.g2xbih.ng.0001.usw2.cache.amazonaws.com",
      "Port": 6379
    },
    "ReaderEndpoint": {
      "Address": "test-cluster-
ro.g2xbih.ng.0001.usw2.cache.amazonaws.com",
      "Port": 6379
    },
    "NodeGroupMembers": [
      {
        "CacheClusterId": "test-cluster-001",
        "CacheNodeId": "0001",
        "ReadEndpoint": {
          "Address": "test-
cluster-001.g2xbih.0001.usw2.cache.amazonaws.com",
          "Port": 6379
        },
        "PreferredAvailabilityZone": "us-west-2c",
        "CurrentRole": "primary"
      },
      {
        "CacheClusterId": "test-cluster-002",
        "CacheNodeId": "0001",
        "ReadEndpoint": {
          "Address": "test-
cluster-002.g2xbih.0001.usw2.cache.amazonaws.com",
          "Port": 6379
        },
        "PreferredAvailabilityZone": "us-west-2b",
        "CurrentRole": "replica"
      }
    ]
  }
]
```

```

        {
            "CacheClusterId": "test-cluster-003",
            "CacheNodeId": "0001",
            "ReadEndpoint": {
                "Address": "test-
cluster-003.g2xbih.0001.usw2.cache.amazonaws.com",
                "Port": 6379
            },
            "PreferredAvailabilityZone": "us-west-2a",
            "CurrentRole": "replica"
        }
    ]
}
],
"SnapshottingClusterId": "test-cluster-002",
"AutomaticFailover": "enabled",
"MultiAZ": "disabled",
"SnapshotRetentionLimit": 1,
"SnapshotWindow": "08:00-09:00",
"ClusterEnabled": false,
"CacheNodeType": "cache.r5.large",
"TransitEncryptionEnabled": false,
"AtRestEncryptionEnabled": false
}
}

```

For more information, see [Modifying a Replication Group](#) in the *Elasticache User Guide*.

- For API details, see [ModifyReplicationGroup](#) in *AWS CLI Command Reference*.

modify-user-group

The following code example shows how to use `modify-user-group`.

AWS CLI

To modify a user group

The following `modify-user-group` example adds a user to the user group.

```

aws elasticache modify-user-group \
  --user-group-id myusergroup \
  --user-ids-to-add user1

```

Output:

```
{
  "UserGroupId": "myusergroup",
  "Status": "modifying",
  "Engine": "redis",
  "UserIds": [
    "default"
  ],
  "PendingChanges": {
    "UserIdsToAdd": [
      "user1"
    ]
  },
  "ReplicationGroups": [],
  "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxx52:usergroup:myusergroup"
}
```

For more information, see [Authenticating Users with Role-Based Access Control \(RBAC\)](#) in the *Elasticache User Guide*.

- For API details, see [ModifyUserGroup](#) in *AWS CLI Command Reference*.

modify-user

The following code example shows how to use `modify-user`.

AWS CLI**To modify a user**

The following `modify-user` example modifies a user's access string.

```
aws elasticache modify-user \  
  --user-id user2 \  
  --append-access-string "on ~* +@all"
```

Output:

```
{
  "UserId": "user2",
  "UserName": "myUser",
```

```

    "Status": "modifying",
    "Engine": "redis",
    "AccessString": "on ~* +@all",
    "UserGroupIds": [],
    "Authentication": {
        "Type": "password",
        "PasswordCount": 1
    },
    "ARN": "arn:aws:elasticache:us-west-2:xxxxxxxxxx52:user:user2"
}

```

For more information, see [Authenticating Users with Role-Based Access Control \(RBAC\)](#) in the *Elasticache User Guide*.

- For API details, see [ModifyUser](#) in *AWS CLI Command Reference*.

purchase-reserved-cache-nodes-offering

The following code example shows how to use `purchase-reserved-cache-nodes-offering`.

AWS CLI

To purchase a reserved-cache-node-offering

The following `purchase-reserved-cache-nodes-offering` example allows you to purchase a reserved cache node offering.

```

aws elasticache purchase-reserved-cache-nodes-offering \
  --reserved-cache-nodes-offering-id xxxxxxxx-4da5-4b90-b92d-929fbd7abed2

```

Output

```

{
  "ReservedCacheNode": {
    "ReservedCacheNodeId": "ri-2020-06-30-17-59-40-474",
    "ReservedCacheNodesOfferingId": "xxxxxxx-4da5-4b90-b92d-929fbd7abed2",
    "CacheNodeType": "cache.m3.2xlarge",
    "StartTime": "2020-06-30T17:59:40.474000+00:00",
    "Duration": 31536000,
    "FixedPrice": 1772.0,
    "UsagePrice": 0.0,
    "CacheNodeCount": 1,
  }
}

```

```

    "ProductDescription": "redis",
    "OfferingType": "Heavy Utilization",
    "State": "payment-pending",
    "RecurringCharges": [
      {
        "RecurringChargeAmount": 0.25,
        "RecurringChargeFrequency": "Hourly"
      }
    ]
  }
}

```

For more information, see [Getting Info About Reserved Node Offerings](#) in the *Elasticache Redis User Guide* or [Getting Info About Reserved Node Offerings](#) in the *Elasticache Memcached User Guide*.

- For API details, see [PurchaseReservedCacheNodesOffering](#) in *AWS CLI Command Reference*.

reboot-cache-cluster

The following code example shows how to use `reboot-cache-cluster`.

AWS CLI

To reboot a cache cluster

The following `reboot-cache-cluster` example reboots some, or all, of the cache nodes within a provisioned cluster. This operation applies any modified cache parameter groups to the cluster. The reboot operation takes place as soon as possible, and results in a momentary outage to the cluster. During the reboot, the cluster status is set to REBOOTING.

```

aws elasticache reboot-cache-cluster \
  --cache-cluster-id "my-cluster-001" \
  --cache-node-ids-to-reboot "0001"

```

Output:

```

{
  "CacheCluster": {
    "CacheClusterId": "my-cluster-001",
    "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/home#client-download:",
  }
}

```

```
"CacheNodeType": "cache.r5.xlarge",
"Engine": "redis",
"EngineVersion": "5.0.5",
"CacheClusterStatus": "rebooting cache cluster nodes",
"NumCacheNodes": 1,
"PreferredAvailabilityZone": "us-west-2a",
"CacheClusterCreateTime": "2019-11-26T03:35:04.546Z",
"PreferredMaintenanceWindow": "mon:04:05-mon:05:05",
"PendingModifiedValues": {},
"NotificationConfiguration": {
  "TopicArn": "arn:aws:sns:us-west-2:xxxxxxxxxxx152:My_Topic",
  "TopicStatus": "active"
},
"CacheSecurityGroups": [],
"CacheParameterGroup": {
  "CacheParameterGroupName": "mygroup",
  "ParameterApplyStatus": "in-sync",
  "CacheNodeIdsToReboot": []
},
"CacheSubnetGroupName": "kxkxk",
"AutoMinorVersionUpgrade": true,
"SecurityGroups": [
  {
    "SecurityGroupId": "sg-xxxxxxxxxxxx836",
    "Status": "active"
  },
  {
    "SecurityGroupId": "sg-xxxxxxx7b",
    "Status": "active"
  }
],
"ReplicationGroupId": "my-cluster",
"SnapshotRetentionLimit": 0,
"SnapshotWindow": "07:30-08:30",
"TransitEncryptionEnabled": false,
"AtRestEncryptionEnabled": false
}
}
```

For more information, see [Rebooting a Cluster <https://docs.aws.amazon.com/AmazonElastiCache/latest/red-ug/Clusters.Rebooting.html>](https://docs.aws.amazon.com/AmazonElastiCache/latest/red-ug/Clusters.Rebooting.html) in the *ElastiCache User Guide*.

- For API details, see [RebootCacheCluster](#) in *AWS CLI Command Reference*.

reset-cache-parameter-group

The following code example shows how to use `reset-cache-parameter-group`.

AWS CLI

To reset a cache parameter group

The following `reset-cache-parameter-group` example modifies the parameters of a cache parameter group to the engine or system default value. You can reset specific parameters by submitting a list of parameter names. To reset the entire cache parameter group, specify the `--reset-all-parameters` and `--cache-parameter-group-name` parameters.

```
aws elasticache reset-cache-parameter-group \  
  --cache-parameter-group-name "mygroup" \  
  --reset-all-parameters
```

Output:

```
{  
  "CacheParameterGroupName": "mygroup"  
}
```

- For API details, see [ResetCacheParameterGroup](#) in *AWS CLI Command Reference*.

start-migration

The following code example shows how to use `start-migration`.

AWS CLI

To start a migration

The following `start-migration` migrates your data from self-hosted Redis on Amazon EC2 to Amazon ElastiCache, using the Redis engine.

```
aws elasticache start-migration \  
  --replication-group-id test \  
  --customer-node-endpoint-list  
  "Address='test.g2xbih.ng.0001.usw2.cache.amazonaws.com',Port=6379"
```

Output

```
{
  "ReplicationGroup": {
    "ReplicationGroupId": "test",
    "Description": "test",
    "GlobalReplicationGroupInfo": {},
    "Status": "modifying",
    "PendingModifiedValues": {},
    "MemberClusters": [
      "test-001",
      "test-002",
      "test-003"
    ],
    "NodeGroups": [
      {
        "NodeGroupId": "0001",
        "Status": "available",
        "PrimaryEndpoint": {
          "Address": "test.g2xbih.ng.0001.usw2.cache.amazonaws.com",
          "Port": 6379
        },
        "ReaderEndpoint": {
          "Address": "test-ro.g2xbih.ng.0001.usw2.cache.amazonaws.com",
          "Port": 6379
        },
        "NodeGroupMembers": [
          {
            "CacheClusterId": "test-001",
            "CacheNodeId": "0001",
            "ReadEndpoint": {
              "Address":
"test-001.g2xbih.0001.usw2.cache.amazonaws.com",
              "Port": 6379
            },
            "PreferredAvailabilityZone": "us-west-2a",
            "CurrentRole": "primary"
          },
          {
            "CacheClusterId": "test-002",
            "CacheNodeId": "0001",
            "ReadEndpoint": {
              "Address":
"test-002.g2xbih.0001.usw2.cache.amazonaws.com",
```

```

        "Port": 6379
      },
      "PreferredAvailabilityZone": "us-west-2c",
      "CurrentRole": "replica"
    },
    {
      "CacheClusterId": "test-003",
      "CacheNodeId": "0001",
      "ReadEndpoint": {
        "Address":
"test-003.g2xbih.0001.usw2.cache.amazonaws.com",
        "Port": 6379
      },
      "PreferredAvailabilityZone": "us-west-2b",
      "CurrentRole": "replica"
    }
  ]
}
],
"SnapshottingClusterId": "test-002",
"AutomaticFailover": "enabled",
"MultiAZ": "enabled",
"SnapshotRetentionLimit": 1,
"SnapshotWindow": "07:30-08:30",
"ClusterEnabled": false,
"CacheNodeType": "cache.r5.large",
"TransitEncryptionEnabled": false,
"AtRestEncryptionEnabled": false
}
}

```

For more information, see [Online Migration to ElastiCache](#) in the *ElastiCache User Guide*.

- For API details, see [StartMigration](#) in *AWS CLI Command Reference*.

test-failover

The following code example shows how to use test-failover.

AWS CLI

To test failover of a node group

The following test-failover example tests automatic failover on the specified node group (called a shard in the console) in a replication group (called a cluster in the console).

```
aws elasticache test-failover /
  --replication-group-id "mycluster" /
  --node-group-id "0001"
```

Output:

```
{
  "ReplicationGroup": {
    "ReplicationGroupId": "mycluster",
    "Description": "My Cluster",
    "Status": "available",
    "PendingModifiedValues": {},
    "MemberClusters": [
      "mycluster-0001-001",
      "mycluster-0001-002",
      "mycluster-0001-003",
      "mycluster-0002-001",
      "mycluster-0002-002",
      "mycluster-0002-003",
      "mycluster-0003-001",
      "mycluster-0003-002",
      "mycluster-0003-003"
    ],
    "NodeGroups": [
      {
        "NodeGroupId": "0001",
        "Status": "available",
        "Slots": "0-5461",
        "NodeGroupMembers": [
          {
            "CacheClusterId": "mycluster-0001-001",
            "CacheNodeId": "0001",
            "PreferredAvailabilityZone": "us-west-2b"
          },
          {
            "CacheClusterId": "mycluster-0001-002",
            "CacheNodeId": "0001",
            "PreferredAvailabilityZone": "us-west-2a"
          },
          {
```

```
        "CacheClusterId": "mycluster-0001-003",
        "CacheNodeId": "0001",
        "PreferredAvailabilityZone": "us-west-2c"
    }
]
},
{
    "NodeGroupId": "0002",
    "Status": "available",
    "Slots": "5462-10922",
    "NodeGroupMembers": [
        {
            "CacheClusterId": "mycluster-0002-001",
            "CacheNodeId": "0001",
            "PreferredAvailabilityZone": "us-west-2a"
        },
        {
            "CacheClusterId": "mycluster-0002-002",
            "CacheNodeId": "0001",
            "PreferredAvailabilityZone": "us-west-2b"
        },
        {
            "CacheClusterId": "mycluster-0002-003",
            "CacheNodeId": "0001",
            "PreferredAvailabilityZone": "us-west-2c"
        }
    ]
},
{
    "NodeGroupId": "0003",
    "Status": "available",
    "Slots": "10923-16383",
    "NodeGroupMembers": [
        {
            "CacheClusterId": "mycluster-0003-001",
            "CacheNodeId": "0001",
            "PreferredAvailabilityZone": "us-west-2c"
        },
        {
            "CacheClusterId": "mycluster-0003-002",
            "CacheNodeId": "0001",
            "PreferredAvailabilityZone": "us-west-2b"
        }
    ]
}
```

```
        "CacheClusterId": "mycluster-0003-003",
        "CacheNodeId": "0001",
        "PreferredAvailabilityZone": "us-west-2a"
    }
]
},
"AutomaticFailover": "enabled",
"ConfigurationEndpoint": {
    "Address": "mycluster.xxxxih.clustercfg.usw2.cache.amazonaws.com",
    "Port": 6379
},
"SnapshotRetentionLimit": 1,
"SnapshotWindow": "13:00-14:00",
"ClusterEnabled": true,
"CacheNodeType": "cache.r5.large",
"TransitEncryptionEnabled": false,
"AtRestEncryptionEnabled": false
}
}
```

- For API details, see [TestFailover](#) in *AWS CLI Command Reference*.

MediaStore examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with MediaStore.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-container

The following code example shows how to use create-container.

AWS CLI

To create a container

The following create-container example creates a new, empty container.

```
aws mediastore create-container --container-name ExampleContainer
```

Output:

```
{
  "Container": {
    "AccessLoggingEnabled": false,
    "CreationTime": 1563557265,
    "Name": "ExampleContainer",
    "Status": "CREATING",
    "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/
ExampleContainer"
  }
}
```

For more information, see [Creating a Container](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [CreateContainer](#) in *AWS CLI Command Reference*.

delete-container-policy

The following code example shows how to use delete-container-policy.

AWS CLI

To delete a container policy

The following delete-container-policy example deletes the policy that is assigned to the specified container. When the policy is deleted, AWS Elemental MediaStore automatically assigns the default policy to the container.

```
aws mediastore delete-container-policy \  
  --container-name LiveEvents
```

This command produces no output.

For more information, see [DeleteContainerPolicy](#) in the *AWS Elemental MediaStore API reference*.

- For API details, see [DeleteContainerPolicy](#) in *AWS CLI Command Reference*.

delete-container

The following code example shows how to use delete-container.

AWS CLI

To delete a container

The following delete-container example deletes the specified container. You can delete a container only if it has no objects.

```
aws mediastore delete-container \  
  --container-name=ExampleLiveDemo
```

This command produces no output.

For more information, see [Deleting a Container](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [DeleteContainer](#) in *AWS CLI Command Reference*.

delete-cors-policy

The following code example shows how to use delete-cors-policy.

AWS CLI

To delete a CORS policy

The following delete-cors-policy example deletes the cross-origin resource sharing (CORS) policy that is assigned to the specified container.

```
aws mediastore delete-cors-policy \  
  --container-name ExampleLiveDemo
```



```
--container-name ExampleContainer
```

This command produces no output.

For more information, see [Deleting a CORS Policy](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [DeleteCorsPolicy](#) in *AWS CLI Command Reference*.

delete-lifecycle-policy

The following code example shows how to use `delete-lifecycle-policy`.

AWS CLI

To delete an object lifecycle policy

The following `delete-lifecycle-policy` example deletes the object lifecycle policy attached to the specified container. This change can take up to 20 minutes to take effect.

```
aws mediastore delete-lifecycle-policy \  
  --container-name LiveEvents
```

This command produces no output.

For more information, see [Deleting an Object Lifecycle Policy](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [DeleteLifecyclePolicy](#) in *AWS CLI Command Reference*.

describe-container

The following code example shows how to use `describe-container`.

AWS CLI

To view the details of a container

The following `describe-container` example displays the details of the specified container.

```
aws mediastore describe-container \  
  --container-name ExampleContainer
```

Output:

```
{
  "Container": {
    "CreationTime": 1563558086,
    "AccessLoggingEnabled": false,
    "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/
ExampleContainer",
    "Status": "ACTIVE",
    "Name": "ExampleContainer",
    "Endpoint": "https://aaabbbcccddee.data.mediastore.us-west-2.amazonaws.com"
  }
}
```

For more information, see [Viewing the Details for a Container](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [DescribeContainer](#) in *AWS CLI Command Reference*.

describe-object

The following code example shows how to use `describe-object`.

AWS CLI**To view a list of objects and folders in a specific container**

The following `describe-object` example displays items (objects and folders) stored in a specific container.

```
aws mediastore-data describe-object \
  --endpoint https://aaabbbcccddee.data.mediastore.us-west-2.amazonaws.com \
  --path /folder_name/file1234.jpg
```

Output:

```
{
  "ContentType": "image/jpeg",
  "LastModified": "Fri, 19 Jul 2019 21:32:20 GMT",
  "ContentLength": "2307346",
  "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9e99999994dd89ff7f5555555555555555da6d3"
}
```

For more information, see [Viewing the Details of an Object](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [DescribeObject](#) in *AWS CLI Command Reference*.

get-container-policy

The following code example shows how to use `get-container-policy`.

AWS CLI

To view a container policy

The following `get-container-policy` example displays the resource-based policy of the specified container.

```
aws mediastore get-container-policy \  
  --container-name ExampleLiveDemo
```

Output:

```
{  
  "Policy": {  
    "Version": "2012-10-17",  
    "Statement": [  
      {  
        "Sid": "PublicReadOverHttps",  
        "Effect": "Allow",  
        "Principal": {  
          "AWS": "arn:aws:iam::111122223333:root"  
        },  
        "Action": [  
          "mediastore:GetObject",  
          "mediastore:DescribeObject"  
        ],  
        "Resource": "arn:aws:mediastore:us-west-2:111122223333:container/  
ExampleLiveDemo/",  
        "Condition": {  
          "Bool": {  
            "aws:SecureTransport": "true"  
          }  
        }  
      }  
    ]  
  }  
}
```

```
    ]
  }
}
```

For more information, see [Viewing a Container Policy](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [GetContainerPolicy](#) in *AWS CLI Command Reference*.

get-cors-policy

The following code example shows how to use `get-cors-policy`.

AWS CLI

To view a CORS policy

The following `get-cors-policy` example displays the cross-origin resource sharing (CORS) policy that is assigned to the specified container.

```
aws mediastore get-cors-policy \
  --container-name ExampleContainer \
  --region us-west-2
```

Output:

```
{
  "CorsPolicy": [
    {
      "AllowedMethods": [
        "GET",
        "HEAD"
      ],
      "MaxAgeSeconds": 3000,
      "AllowedOrigins": [
        ""
      ],
      "AllowedHeaders": [
        ""
      ]
    }
  ]
}
```

```
}
```

For more information, see [Viewing a CORS Policy](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [GetCorsPolicy](#) in *AWS CLI Command Reference*.

get-lifecycle-policy

The following code example shows how to use `get-lifecycle-policy`.

AWS CLI

To view an object lifecycle policy

The following `get-lifecycle-policy` example displays the object lifecycle policy attached to the specified container.

```
aws mediastore get-lifecycle-policy \  
  --container-name LiveEvents
```

Output:

```
{  
  "LifecyclePolicy": {  
    "rules": [  
      {  
        "definition": {  
          "path": [  
            {  
              "prefix": "Football/"  
            },  
            {  
              "prefix": "Baseball/"  
            }  
          ],  
          "days_since_create": [  
            {  
              "numeric": [  
                ">",  
                28  
              ]  
            }  
          ]  
        }  
      ]  
    }  
  }
```

```

    ]
  },
  "action": "EXPIRE"
}
]
}
}

```

For more information, see [Viewing an Object Lifecycle Policy](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [GetLifecyclePolicy](#) in *AWS CLI Command Reference*.

get-object

The following code example shows how to use `get-object`.

AWS CLI

To download an object

The following `get-object` example download an object to the specified endpoint.

```

aws mediastore-data get-object \
  --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com \
  --path=/folder_name/README.md README.md

```

Output:

```

{
  "ContentLength": "2307346",
  "ContentType": "image/jpeg",
  "LastModified": "Fri, 19 Jul 2019 21:32:20 GMT",
  "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9e4dd89ff7f5555555555555555da6d3",
  "StatusCode": 200
}

```

To download part of an object

The following `get-object` example downloads a portion an object to the specified endpoint.

```

aws mediastore-data get-object \

```

```
--endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com \  
--path /folder_name/README.md \  
--range="bytes=0-100" README2.md
```

Output:

```
{  
  "StatusCode": 206,  
  "ContentRange": "bytes 0-100/2307346",  
  "ContentLength": "101",  
  "LastModified": "Fri, 19 Jul 2019 21:32:20 GMT",  
  "ContentType": "image/jpeg",  
  "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9e4dd89ff7f5555555555555555da6d3"  
}
```

For more information, see [Downloading an Object](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [GetObject](#) in *AWS CLI Command Reference*.

list-containers

The following code example shows how to use `list-containers`.

AWS CLI

To view a list of containers

The following `list-containers` example displays a list of all containers that are associated with your account.

```
aws mediastore list-containers
```

Output:

```
{  
  "Containers": [  
    {  
      "CreationTime": 1505317931,  
      "Endpoint": "https://aaabbbcccdddee.data.mediastore.us-  
west-2.amazonaws.com",  
      "Status": "ACTIVE",
```

```

        "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/
ExampleLiveDemo",
        "AccessLoggingEnabled": false,
        "Name": "ExampleLiveDemo"
    },
    {
        "CreationTime": 1506528818,
        "Endpoint": "https://ffffggghhhiiijj.data.mediastore.us-
west-2.amazonaws.com",
        "Status": "ACTIVE",
        "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/
ExampleContainer",
        "AccessLoggingEnabled": false,
        "Name": "ExampleContainer"
    }
]
}

```

For more information, see [Viewing a List of Containers](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [ListContainers](#) in *AWS CLI Command Reference*.

list-items

The following code example shows how to use `list-items`.

AWS CLI

Example 1: To view a list of objects and folders in a specific container

The following `list-items` example displays items (objects and folders) stored in the specified container.

```

aws mediastore-data list-items \
  --endpoint https://aaabbbcccddee.data.mediastore.us-west-2.amazonaws.com

```

Output:

```

{
  "Items": [
    {

```



```

        "ContentType": "image/jpeg",
        "LastModified": 1563571859.379,
        "Name": "filename.jpg",
        "Type": "OBJECT",
        "ETag":
"543ab21abcd1a234ab123456a1a2b12345ab12abc12a1234abc1a2bc12345a12",
        "ContentLength": 3784
    },
    {
        "Type": "FOLDER",
        "Name": "ExampleLiveDemo"
    }
]
}

```

Example 2: To view a list of objects and folders in a specific folder

The following `list-items` example displays items (objects and folders) stored in a specific folder.

```

aws mediastore-data list-items \
  --endpoint https://aaabbbccdddee.data.mediastore.us-west-2.amazonaws.com

```

Output:

```

{
  "Items": [
    {
      "ContentType": "image/jpeg",
      "LastModified": 1563571859.379,
      "Name": "filename.jpg",
      "Type": "OBJECT",
      "ETag":
"543ab21abcd1a234ab123456a1a2b12345ab12abc12a1234abc1a2bc12345a12",
      "ContentLength": 3784
    },
    {
      "Type": "FOLDER",
      "Name": "ExampleLiveDemo"
    }
  ]
}

```

For more information, see [Viewing a List of Objects](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [ListItems](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags for a container

The following `list-tags-for-resource` example displays the tag keys and values assigned to the specified container.

```
aws mediastore list-tags-for-resource \  
  --resource arn:aws:mediastore:us-west-2:1213456789012:container/ExampleContainer
```

Output:

```
{  
  "Tags": [  
    {  
      "Value": "Test",  
      "Key": "Environment"  
    },  
    {  
      "Value": "West",  
      "Key": "Region"  
    }  
  ]  
}
```

For more information, see [ListTagsForResource](#) in the *AWS Elemental MediaStore API Reference*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

put-container-policy

The following code example shows how to use `put-container-policy`.

AWS CLI

To edit a container policy

The following `put-container-policy` example assigns a different policy to the specified container. In this example, the updated policy is defined in a file named `LiveEventsContainerPolicy.json`.

```
aws mediastore put-container-policy \  
  --container-name LiveEvents \  
  --policy file://LiveEventsContainerPolicy.json
```

This command produces no output.

For more information, see [Editing a Container Policy](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [PutContainerPolicy](#) in *AWS CLI Command Reference*.

put-cors-policy

The following code example shows how to use `put-cors-policy`.

AWS CLI

Example 1: To add a CORS policy

The following `put-cors-policy` example adds a cross-origin resource sharing (CORS) policy to the specified container. The contents of the CORS policy are in the file named `corsPolicy.json`.

```
aws mediastore put-cors-policy \  
  --container-name ExampleContainer \  
  --cors-policy file://corsPolicy.json
```

This command produces no output.

For more information, see [Adding a CORS Policy to a Container](#) in the *AWS Elemental MediaStore User Guide*.

Example 2: To edit a CORS policy

The following `put-cors-policy` example updates the cross-origin resource sharing (CORS) policy that is assigned to the specified container. The contents of the updated CORS policy are in the file named `corsPolicy2.json`.

For more information, see [Editing a CORS Policy](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [PutCorsPolicy](#) in *AWS CLI Command Reference*.

put-lifecycle-policy

The following code example shows how to use `put-lifecycle-policy`.

AWS CLI

To create an object lifecycle policy

The following `put-lifecycle-policy` example attaches an object lifecycle policy to the specified container. This enables you to specify how long the service should store objects in your container. MediaStore deletes objects in the container once they reach their expiration date, as indicated in the policy, which is in the file named `LiveEventsLifecyclePolicy.json`.

```
aws mediastore put-lifecycle-policy \  
  --container-name ExampleContainer \  
  --lifecycle-policy file://ExampleLifecyclePolicy.json
```

This command produces no output.

For more information, see [Adding an Object Lifecycle Policy to a Container](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [PutLifecyclePolicy](#) in *AWS CLI Command Reference*.

put-object

The following code example shows how to use `put-object`.

AWS CLI

To upload an object

The following `put-object` example uploads an object to the specified container. You can specify a folder path where the object will be saved within the container. If the folder already exists, AWS Elemental MediaStore stores the object in the folder. If the folder doesn't exist, the service creates it, and then stores the object in the folder.

```
aws mediastore-data put-object \  
  --endpoint https://aaabbbccdddee.data.mediastore.us-west-2.amazonaws.com \  
  --body README.md \  
  --path /folder_name/README.md \  
  --cache-control "max-age=6, public" \  
  --content-type binary/octet-stream
```

Output:

```
{  
  "ContentSHA256":  
    "74b5fdb517f423ed750ef214c44adfe2be36e37d861eafe9c842cbe1bf387a9d",  
  "StorageClass": "TEMPORAL",  
  "ETag": "af3e4731af032167a106015d1f2fe934e68b32ed1aa297a9e325f5c64979277b"  
}
```

For more information, see [Uploading an Object](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [PutObject](#) in *AWS CLI Command Reference*.

start-access-logging

The following code example shows how to use `start-access-logging`.

AWS CLI

To enable access logging on a container

The following `start-access-logging` example enable access logging on the specified container.

```
aws mediastore start-access-logging \  
  --container-name LiveEvents
```

This command produces no output.

For more information, see [Enabling Access Logging for a Container](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [StartAccessLogging](#) in *AWS CLI Command Reference*.

stop-access-logging

The following code example shows how to use stop-access-logging.

AWS CLI

To disable access logging on a container

The following stop-access-logging example disables access logging on the specified container.

```
aws mediastore stop-access-logging \  
  --container-name LiveEvents
```

This command produces no output.

For more information, see [Disabling Access Logging for a Container](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [StopAccessLogging](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To add tags to a container

The following tag-resource example adds tag keys and values to the specified container.

```
aws mediastore tag-resource \  
  --resource arn:aws:mediastore:us-west-2:123456789012:container/ExampleContainer \  
  --tags '[{"Key": "Region", "Value": "West"}, {"Key": "Environment", "Value": "Test"}]'
```

This command produces no output.

For more information, see [TagResource](#) in the *AWS Elemental MediaStore API Reference*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from a container

The following `untag-resource` example removes the specified tag key and its associated value from a container.

```
aws mediastore untag-resource \  
  --resource arn:aws:mediastore:us-west-2:123456789012:container/ExampleContainer \  
  --tag-keys Region
```

This command produces no output.

For more information, see [UntagResource](#) in the *AWS Elemental MediaStore API Reference*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

Amazon EMR examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon EMR.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-instance-fleet

The following code example shows how to use `add-instance-fleet`.

AWS CLI

To add a task instance fleet to a cluster

This example adds a new task instance fleet to the cluster specified.

Command:

```
aws emr add-instance-fleet --cluster-id 'j-12ABCDEFGHI34JK' --instance-fleet
InstanceFleetType=TASK,TargetSpotCapacity=1,LaunchSpecifications={SpotSpecification='{Timeo
```

Output:

```
{
  "ClusterId": "j-12ABCDEFGHI34JK",
  "InstanceFleetId": "if-23ABCDEFGHI45JJ"
}
```

- For API details, see [AddInstanceFleet](#) in *AWS CLI Command Reference*.

add-steps

The following code example shows how to use `add-steps`.

AWS CLI

1. To add Custom JAR steps to a cluster

Command:

```
aws emr add-steps --cluster-id j-XXXXXXXX --steps
Type=CUSTOM_JAR,Name=CustomJAR,ActionOnFailure=CONTINUE,Jar=s3://mybucket/
mytest.jar,Args=arg1,arg2,arg3
```



```
Type=CUSTOM_JAR,Name=CustomJAR,ActionOnFailure=CONTINUE,Jar=s3://mybucket/mytest.jar,MainClass=mymainclass,Args=arg1,arg2,arg3
```

Required parameters:

```
Jar
```

Optional parameters:

```
Type, Name, ActionOnFailure, Args
```

Output:

```
{
  "StepIds":[
    "s-XXXXXXXX",
    "s-YYYYYYYY"
  ]
}
```

2. To add Streaming steps to a cluster

Command:

```
aws emr add-steps --cluster-id j-XXXXXXXX --steps Type=STREAMING,Name='Streaming Program',ActionOnFailure=CONTINUE,Args=[-files,s3://elasticmapreduce/samples/wordcount/wordSplitter.py,-mapper,wordSplitter.py,-reducer,aggregate,-input,s3://elasticmapreduce/samples/wordcount/input,-output,s3://mybucket/wordcount/output]
```

Required parameters:

```
Type, Args
```

Optional parameters:

```
Name, ActionOnFailure
```

JSON equivalent (contents of step.json):

```
[
```

```
{
  "Name": "JSON Streaming Step",
  "Args": ["-files", "s3://elasticmapreduce/samples/wordcount/wordSplitter.py", "-mapper", "wordSplitter.py", "-reducer", "aggregate", "-input", "s3://elasticmapreduce/samples/wordcount/input", "-output", "s3://mybucket/wordcount/output"],
  "ActionOnFailure": "CONTINUE",
  "Type": "STREAMING"
}
```

NOTE: JSON arguments must include options and values as their own items in the list.

Command (using step.json):

```
aws emr add-steps --cluster-id j-XXXXXXXX --steps file://./step.json
```

Output:

```
{
  "StepIds": [
    "s-XXXXXXXX",
    "s-YYYYYYYY"
  ]
}
```

3. To add a Streaming step with multiple files to a cluster (JSON only)

JSON (multiplefiles.json):

```
[
  {
    "Name": "JSON Streaming Step",
    "Type": "STREAMING",
    "ActionOnFailure": "CONTINUE",
    "Args": [
      "-files",
      "s3://mybucket/mapper.py,s3://mybucket/reducer.py",
      "-mapper",
      "mapper.py",
      "-reducer",
      "reducer.py",
      "-input",
      "s3://mybucket/input",
    ]
  }
]
```

```
        "-output",
        "s3://mybucket/output"]
    }
]
```

Command:

```
aws emr add-steps --cluster-id j-XXXXXXXX --steps file://./multiplefiles.json
```

Required parameters:

Type, Args

Optional parameters:

Name, ActionOnFailure

Output:

```
{
  "StepIds": [
    "s-XXXXXXXX",
  ]
}
```

4. To add Hive steps to a cluster**Command:**

```
aws emr add-steps --cluster-id j-XXXXXXXX --steps Type=HIVE,Name='Hive
program',ActionOnFailure=CONTINUE,Args=[-f,s3://mybucket/myhivescript.q,-
d,INPUT=s3://mybucket/myhiveinput,-d,OUTPUT=s3://mybucket/myhiveoutput,arg1,arg2]
Type=HIVE,Name='Hive steps',ActionOnFailure=TERMINATE_CLUSTER,Args=[-
f,s3://elasticmapreduce/samples/hive-ads/libs/model-build.q,-d,INPUT=s3://
elasticmapreduce/samples/hive-ads/tables,-d,OUTPUT=s3://mybucket/hive-ads/
output/2014-04-18/11-07-32,-d,LIBS=s3://elasticmapreduce/samples/hive-ads/libs]
```

Required parameters:

Type, Args

Optional parameters:

Name, ActionOnFailure

Output:

```
{
  "StepIds": [
    "s-XXXXXXXX",
    "s-YYYYYYYY"
  ]
}
```

5. To add Pig steps to a cluster**Command:**

```
aws emr add-steps --cluster-id j-XXXXXXXX --steps Type=PIG,Name='Pig
program',ActionOnFailure=CONTINUE,Args=[-f,s3://mybucket/mypigscript.pig,-
p,INPUT=s3://mybucket/mypiginput,-p,OUTPUT=s3://mybucket/mypigoutput,arg1,arg2]
Type=PIG,Name='Pig program',Args=[-f,s3://elasticmapreduce/samples/pig-apache/do-
reports2.pig,-p,INPUT=s3://elasticmapreduce/samples/pig-apache/input,-p,OUTPUT=s3://
mybucket/pig-apache/output,arg1,arg2]
```

Required parameters:

Type, Args

Optional parameters:

Name, ActionOnFailure

Output:

```
{
  "StepIds": [
    "s-XXXXXXXX",
    "s-YYYYYYYY"
  ]
}
```

6. To add Impala steps to a cluster

Command:

```
aws emr add-steps --cluster-id j-XXXXXXXX --steps Type=IMPALA,Name='Impala
program',ActionOnFailure=CONTINUE,Args=--impala-script,s3://myimpala/input,--
console-output-path,s3://myimpala/output
```

Required parameters:

Type, Args

Optional parameters:

Name, ActionOnFailure

Output:

```
{
  "StepIds":[
    "s-XXXXXXXX",
    "s-YYYYYYYY"
  ]
}
```

- For API details, see [AddSteps](#) in *AWS CLI Command Reference*.

add-tags

The following code example shows how to use add-tags.

AWS CLI

1. To add tags to a cluster

Command:

```
aws emr add-tags --resource-id j-xxxxxxx --tags name="John Doe" age=29 sex=male
address="123 East NW Seattle"
```

Output:

None

2. To list tags of a cluster

--Command:

```
aws emr describe-cluster --cluster-id j-XXXXXXYY --query Cluster.Tags
```

Output:

```
[
  {
    "Value": "male",
    "Key": "sex"
  },
  {
    "Value": "123 East NW Seattle",
    "Key": "address"
  },
  {
    "Value": "John Doe",
    "Key": "name"
  },
  {
    "Value": "29",
    "Key": "age"
  }
]
```

- For API details, see [AddTags](#) in *AWS CLI Command Reference*.

create-cluster-examples

The following code example shows how to use `create-cluster-examples`.

AWS CLI

Most of the following examples assume that you specified your Amazon EMR service role and Amazon EC2 instance profile. If you have not done this, you must specify each required IAM role or use the `--use-default-roles` parameter when creating your cluster. For more

information about specifying IAM roles, see [Configure IAM Roles for Amazon EMR Permissions to AWS Services](#) in the *Amazon EMR Management Guide*.

Example 1: To create a cluster

The following `create-cluster` example creates a simple EMR cluster.

```
aws emr create-cluster \  
  --release-label emr-5.14.0 \  
  --instance-type m4.large \  
  --instance-count 2
```

This command produces no output.

Example 2: To create an Amazon EMR cluster with default `ServiceRole` and `InstanceProfile` roles

The following `create-cluster` example creates an Amazon EMR cluster that uses the `--instance-groups` configuration.

```
aws emr create-cluster \  
  --release-label emr-5.14.0 \  
  --service-role EMR_DefaultRole \  
  --ec2-attributes InstanceProfile=EMR_EC2_DefaultRole \  
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large  
  InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large
```

Example 3: To create an Amazon EMR cluster that uses an instance fleet

The following `create-cluster` example creates an Amazon EMR cluster that uses the `--instance-fleets` configuration, specifying two instance types for each fleet and two EC2 Subnets.

```
aws emr create-cluster \  
  --release-label emr-5.14.0 \  
  --service-role EMR_DefaultRole \  
  --ec2-attributes InstanceProfile=EMR_EC2_DefaultRole,SubnetIds=['subnet-  
ab12345c','subnet-de67890f'] \  
  --instance-fleets  
  InstanceFleetType=MASTER,TargetOnDemandCapacity=1,InstanceTypeConfigs=['{InstanceType=m4.la  
  InstanceFleetType=CORE,TargetSpotCapacity=11,InstanceTypeConfigs=['{InstanceType=m4.large,B
```

Example 4: To create a cluster with default roles

The following `create-cluster` example uses the `--use-default-roles` parameter to specify the default service role and instance profile.

```
aws emr create-cluster \  
  --release-label emr-5.9.0 \  
  --use-default-roles \  
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large \  
  InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large \  
  --auto-terminate
```

Example 5: To create a cluster and specify the applications to install

The following `create-cluster` example uses the `--applications` parameter to specify the applications that Amazon EMR installs. This example installs Hadoop, Hive and Pig.

```
aws emr create-cluster \  
  --applications Name=Hadoop Name=Hive Name=Pig \  
  --release-label emr-5.9.0 \  
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large \  
  InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large \  
  --auto-terminate
```

Example 6: To create a cluster that includes Spark

The following example installs Spark.

```
aws emr create-cluster \  
  --release-label emr-5.9.0 \  
  --applications Name=Spark \  
  --ec2-attributes KeyName=myKey \  
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large \  
  InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large \  
  --auto-terminate
```

Example 7: To specify a custom AMI to use for cluster instances

The following `create-cluster` example creates a cluster instance based on the Amazon Linux AMI with ID `ami-a518e6df`.

```
aws emr create-cluster \  
  --release-label emr-5.9.0 \  
  --instance-profile emr-5.9.0-ec2-instance-profile \  
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large \  
  InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large \  
  --auto-terminate
```



```
--name "Cluster with My Custom AMI" \  
--custom-ami-id ami-a518e6df \  
--ebs-root-volume-size 20 \  
--release-label emr-5.9.0 \  
--use-default-roles \  
--instance-count 2 \  
--instance-type m4.large
```

Example 8: To customize application configurations

The following examples use the `--configurations` parameter to specify a JSON configuration file that contains application customizations for Hadoop. For more information, see [Configuring Applications](#) in the *Amazon EMR Release Guide*.

Contents of `configurations.json`:

```
[  
  {  
    "Classification": "mapred-site",  
    "Properties": {  
      "mapred.tasktracker.map.tasks.maximum": 2  
    }  
  },  
  {  
    "Classification": "hadoop-env",  
    "Properties": {},  
    "Configurations": [  
      {  
        "Classification": "export",  
        "Properties": {  
          "HADOOP_DATANODE_HEAPSIZE": 2048,  
          "HADOOP_NAMENODE_OPTS": "-XX:GCTimeRatio=19"  
        }  
      }  
    ]  
  }  
]
```

The following example references `configurations.json` as a local file.

```
aws emr create-cluster \  
--configurations file://configurations.json \  
--release-label emr-5.9.0 \  

```

```
--instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large
InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large \
--auto-terminate
```

The following example references `configurations.json` as a file in Amazon S3.

```
aws emr create-cluster \
--configurations https://s3.amazonaws.com/myBucket/configurations.json \
--release-label emr-5.9.0 \
--instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large
InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large \
--auto-terminate
```

Example 9: To create a cluster with master, core, and task instance groups

The following `create-cluster` example uses `--instance-groups` to specify the type and number of EC2 instances to use for master, core, and task instance groups.

```
aws emr create-cluster \
--release-label emr-5.9.0 \
--instance-groups
Name=Master,InstanceGroupType=MASTER,InstanceType=m4.large,InstanceCount=1
Name=Core,InstanceGroupType=CORE,InstanceType=m4.large,InstanceCount=2
Name=Task,InstanceGroupType=TASK,InstanceType=m4.large,InstanceCount=2
```

Example 10: To specify that a cluster should terminate after completing all steps

The following `create-cluster` example uses `--auto-terminate` to specify that the cluster should shut down automatically after completing all steps.

```
aws emr create-cluster \
--release-label emr-5.9.0 \
--instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large
InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large \
--auto-terminate
```

Example 11: To specify cluster configuration details such as the Amazon EC2 key pair, network configuration, and security groups

The following `create-cluster` example creates a cluster with the Amazon EC2 key pair named `myKey` and a customized instance profile named `myProfile`. Key pairs are used to

authorize SSH connections to cluster nodes, most often the master node. For more information, see [Use an Amazon EC2 Key Pair for SSH Credentials](#) in the *Amazon EMR Management Guide*.

```
aws emr create-cluster \  
  --ec2-attributes KeyName=myKey,InstanceProfile=myProfile \  
  --release-label emr-5.9.0 \  
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large  
  InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large \  
  --auto-terminate
```

The following example creates a cluster in an Amazon VPC subnet.

```
aws emr create-cluster \  
  --ec2-attributes SubnetId=subnet-xxxxx \  
  --release-label emr-5.9.0 \  
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large  
  InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large \  
  --auto-terminate
```

The following example creates a cluster in the us-east-1b availability zone.

```
aws emr create-cluster \  
  --ec2-attributes AvailabilityZone=us-east-1b \  
  --release-label emr-5.9.0 \  
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large  
  InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large
```

The following example creates a cluster and specifies only the Amazon EMR-managed security groups.

```
aws emr create-cluster \  
  --release-label emr-5.9.0 \  
  --service-role myServiceRole \  
  --ec2-attributes InstanceProfile=myRole,EmrManagedMasterSecurityGroup=sg-  
  master1,EmrManagedSlaveSecurityGroup=sg-slave1 \  
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large  
  InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large
```

The following example creates a cluster and specifies only additional Amazon EC2 security groups.

```
aws emr create-cluster \
  --release-label emr-5.9.0 \
  --service-role myServiceRole \
  --ec2-attributes InstanceProfile=myRole,AdditionalMasterSecurityGroups=[sg-
addMaster1,sg-addMaster2,sg-addMaster3,sg-
addMaster4],AdditionalSlaveSecurityGroups=[sg-addSlave1,sg-addSlave2,sg-
addSlave3,sg-addSlave4] \
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large
InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large
```

The following example creates a cluster and specifies the EMR-Managed security groups, as well as additional security groups.

```
aws emr create-cluster \
  --release-label emr-5.9.0 \
  --service-role myServiceRole \
  --ec2-attributes InstanceProfile=myRole,EmrManagedMasterSecurityGroup=sg-
master1,EmrManagedSlaveSecurityGroup=sg-slave1,AdditionalMasterSecurityGroups=[sg-
addMaster1,sg-addMaster2,sg-addMaster3,sg-
addMaster4],AdditionalSlaveSecurityGroups=[sg-addSlave1,sg-addSlave2,sg-
addSlave3,sg-addSlave4] \
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large
InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large
```

The following example creates a cluster in a VPC private subnet and use a specific Amazon EC2 security group to enable Amazon EMR service access, which is required for clusters in private subnets.

```
aws emr create-cluster \
  --release-label emr-5.9.0 \
  --service-role myServiceRole \
  --ec2-attributes InstanceProfile=myRole,ServiceAccessSecurityGroup=sg-service-
access,EmrManagedMasterSecurityGroup=sg-master,EmrManagedSlaveSecurityGroup=sg-slave
\
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large
InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large
```

The following example specifies security group configuration parameters using a JSON file named `ec2_attributes.json` that is stored locally. NOTE: JSON arguments must include options and values as their own items in the list.

```
aws emr create-cluster \
  --release-label emr-5.9.0 \
  --service-role myServiceRole \
  --ec2-attributes file://ec2_attributes.json \
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large
  InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large
```

Contents of `ec2_attributes.json`:

```
[
  {
    "SubnetId": "subnet-xxxxxx",
    "KeyName": "myKey",
    "InstanceProfile": "myRole",
    "EmrManagedMasterSecurityGroup": "sg-master1",
    "EmrManagedSlaveSecurityGroup": "sg-slave1",
    "ServiceAccessSecurityGroup": "sg-service-access",
    "AdditionalMasterSecurityGroups": ["sg-addMaster1", "sg-addMaster2", "sg-
addMaster3", "sg-addMaster4"],
    "AdditionalSlaveSecurityGroups": ["sg-addSlave1", "sg-addSlave2", "sg-
addSlave3", "sg-addSlave4"]
  }
]
```

Example 12: To enable debugging and specify a log URI

The following `create-cluster` example uses the `--enable-debugging` parameter, which allows you to view log files more easily using the debugging tool in the Amazon EMR console. The `--log-uri` parameter is required with `--enable-debugging`.

```
aws emr create-cluster \
  --enable-debugging \
  --log-uri s3://myBucket/myLog \
  --release-label emr-5.9.0 \
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large
  InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large \
  --auto-terminate
```

Example 13: To add tags when creating a cluster

Tags are key-value pairs that help you identify and manage clusters. The following `create-cluster` example uses the `--tags` parameter to create three tags for a cluster, one with the

key name `name` and the value `Shirley Rodriguez`, a second with the key name `age` and the value `29`, and a third tag with the key name `department` and the value `Analytics`.

```
aws emr create-cluster \  
  --tags name="Shirley Rodriguez" age=29 department="Analytics" \  
  --release-label emr-5.32.0 \  
  --instance-type m5.xlarge \  
  --instance-count 3 \  
  --use-default-roles
```

The following example lists the tags applied to a cluster.

```
aws emr describe-cluster \  
  --cluster-id j-XXXXXXXXYY \  
  --query Cluster.Tags
```

Example 14: To use a security configuration that enables encryption and other security features

The following `create-cluster` example uses the `--security-configuration` parameter to specify a security configuration for an EMR cluster. You can use security configurations with Amazon EMR version 4.8.0 or later.

```
aws emr create-cluster \  
  --instance-type m4.large \  
  --release-label emr-5.9.0 \  
  --security-configuration mySecurityConfiguration
```

Example 15: To create a cluster with additional EBS storage volumes configured for the instance groups

When specifying additional EBS volumes, the following arguments are required: `VolumeType`, `SizeInGB` if `EbsBlockDeviceConfigs` is specified.

The following `create-cluster` example creates a cluster with multiple EBS volumes attached to EC2 instances in the core instance group.

```
aws emr create-cluster \  
  --release-label emr-5.9.0 \  
  --use-default-roles \  
  --instance-groups
```

```

--instance-groups
InstanceGroupType=MASTER,InstanceCount=1,InstanceType=d2.xlarge
'InstanceGroupType=CORE,InstanceCount=2,InstanceType=d2.xlarge,EbsConfiguration={EbsOptimiz
{VolumeSpecification={VolumeType=io1,SizeInGB=100,Iops=100},VolumesPerInstance=4}}]'
\
--auto-terminate

```

The following example creates a cluster with multiple EBS volumes attached to EC2 instances in the master instance group.

```

aws emr create-cluster \
--release-label emr-5.9.0 \
--use-default-roles \
--instance-groups 'InstanceGroupType=MASTER, InstanceCount=1,
InstanceType=d2.xlarge, EbsConfiguration={EbsOptimized=true,
EbsBlockDeviceConfigs=[{VolumeSpecification={VolumeType=io1, SizeInGB=100,
Iops=100}}],
{VolumeSpecification={VolumeType=standard,SizeInGB=50},VolumesPerInstance=3}}]'
InstanceGroupType=CORE,InstanceCount=2,InstanceType=d2.xlarge \
--auto-terminate

```

Example 16: To create a cluster with an automatic scaling policy

You can attach automatic scaling policies to core and task instance groups using Amazon EMR version 4.0 and later. The automatic scaling policy dynamically adds and removes EC2 instances in response to an Amazon CloudWatch metric. For more information, see [Using Automatic Scaling in Amazon EMR](https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-automatic-scaling.html) in the *Amazon EMR Management Guide*.

When attaching an automatic scaling policy, you must also specify the default role for automatic scaling using `--auto-scaling-role EMR_AutoScaling_DefaultRole`.

The following `create-cluster` example specifies the automatic scaling policy for the CORE instance group using the `AutoScalingPolicy` argument with an embedded JSON structure, which specifies the scaling policy configuration. Instance groups with an embedded JSON structure must have the entire collection of arguments enclosed in single quotes. Using single quotes is optional for instance groups without an embedded JSON structure.

```

aws emr create-cluster
--release-label emr-5.9.0 \
--use-default-roles --auto-scaling-role EMR_AutoScaling_DefaultRole \

```

```
--instance-groups
InstanceGroupType=MASTER,InstanceType=d2.xlarge,InstanceCount=1
'InstanceGroupType=CORE,InstanceType=d2.xlarge,InstanceCount=2,AutoScalingPolicy={Constraints
```

The following example uses a JSON file, `instancegroupconfig.json`, to specify the configuration of all instance groups in a cluster. The JSON file specifies the automatic scaling policy configuration for the core instance group.

```
aws emr create-cluster \
  --release-label emr-5.9.0 \
  --service-role EMR_DefaultRole \
  --ec2-attributes InstanceProfile=EMR_EC2_DefaultRole \
  --instance-groups file://myfolder/instancegroupconfig.json \
  --auto-scaling-role EMR_AutoScaling_DefaultRole
```

Contents of `instancegroupconfig.json`:

```
[
  {
    "InstanceCount": 1,
    "Name": "MyMasterIG",
    "InstanceGroupType": "MASTER",
    "InstanceType": "m4.large"
  },
  {
    "InstanceCount": 2,
    "Name": "MyCoreIG",
    "InstanceGroupType": "CORE",
    "InstanceType": "m4.large",
    "AutoScalingPolicy": {
      "Constraints": {
        "MinCapacity": 2,
        "MaxCapacity": 10
      },
      "Rules": [
        {
          "Name": "Default-scale-out",
          "Description": "Replicates the default scale-out rule in the
console for YARN memory.",
          "Action": {
            "SimpleScalingPolicyConfiguration": {
              "AdjustmentType": "CHANGE_IN_CAPACITY",
```



```

        "ScalingAdjustment": 1,
        "CoolDown": 300
    }
},
"Trigger": {
    "CloudWatchAlarmDefinition": {
        "ComparisonOperator": "LESS_THAN",
        "EvaluationPeriods": 1,
        "MetricName": "YARNMemoryAvailablePercentage",
        "Namespace": "AWS/ElasticMapReduce",
        "Period": 300,
        "Threshold": 15,
        "Statistic": "AVERAGE",
        "Unit": "PERCENT",
        "Dimensions": [
            {
                "Key": "JobFlowId",
                "Value": "${emr.clusterId}"
            }
        ]
    }
}
}
]
}
]

```

Example 17: Add custom JAR steps when creating a cluster

The following `create-cluster` example adds steps by specifying a JAR file stored in Amazon S3. Steps submit work to a cluster. The main function defined in the JAR file executes after EC2 instances are provisioned, any bootstrap actions have executed, and applications are installed. The steps are specified using `Type=CUSTOM_JAR`.

Custom JAR steps require the `Jar=` parameter, which specifies the path and file name of the JAR. Optional parameters are `Type`, `Name`, `ActionOnFailure`, `Args`, and `MainClass`. If main class is not specified, the JAR file should specify `Main-Class` in its manifest file.

```

aws emr create-cluster \
    --steps Type=CUSTOM_JAR,Name=CustomJAR,ActionOnFailure=CONTINUE,Jar=s3://
myBucket/mytest.jar,Args=arg1,arg2,arg3

```

```
Type=CUSTOM_JAR,Name=CustomJAR,ActionOnFailure=CONTINUE,Jar=s3://myBucket/
mytest.jar,MainClass=mymainclass,Args=arg1,arg2,arg3 \
  --release-label emr-5.3.1 \
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large
InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large \
  --auto-terminate
```

Example 18: To add streaming steps when creating a cluster

The following `create-cluster` examples add a streaming step to a cluster that terminates after all steps run. Streaming steps require parameters `Type` and `Args`. Streaming steps optional parameters are `Name` and `ActionOnFailure`.

The following example specifies the step inline.

```
aws emr create-cluster \
  --steps Type=STREAMING,Name='Streaming Program',ActionOnFailure=CONTINUE,Args=[-
files,s3://elasticmapreduce/samples/wordcount/wordSplitter.py,-
mapper,wordSplitter.py,-reducer,aggregate,-input,s3://elasticmapreduce/samples/
wordcount/input,-output,s3://mybucket/wordcount/output] \
  --release-label emr-5.3.1 \
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large
InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large \
  --auto-terminate
```

The following example uses a locally stored JSON configuration file named `multiplefiles.json`. The JSON configuration specifies multiple files. To specify multiple files within a step, you must use a JSON configuration file to specify the step. JSON arguments must include options and values as their own items in the list.

```
aws emr create-cluster \
  --steps file://./multiplefiles.json \
  --release-label emr-5.9.0 \
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large
InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large \
  --auto-terminate
```

Contents of `multiplefiles.json`:

```
[
  {
```

```

    "Name": "JSON Streaming Step",
    "Args": [
        "-files",
        "s3://elasticmapreduce/samples/wordcount/wordSplitter.py",
        "-mapper",
        "wordSplitter.py",
        "-reducer",
        "aggregate",
        "-input",
        "s3://elasticmapreduce/samples/wordcount/input",
        "-output",
        "s3://mybucket/wordcount/output"
    ],
    "ActionOnFailure": "CONTINUE",
    "Type": "STREAMING"
}
]

```

Example 19: To add Hive steps when creating a cluster

The following example add Hive steps when creating a cluster. Hive steps require parameters Type and Args. Hive steps optional parameters are Name and ActionOnFailure.

```

aws emr create-cluster \
  --steps Type=HIVE,Name='Hive
  program',ActionOnFailure=CONTINUE,ActionOnFailure=TERMINATE_CLUSTER,Args=[-
  f,s3://elasticmapreduce/samples/hive-ads/libs/model-build.q,-d,INPUT=s3://
  elasticmapreduce/samples/hive-ads/tables,-d,OUTPUT=s3://mybucket/hive-ads/
  output/2014-04-18/11-07-32,-d,LIBS=s3://elasticmapreduce/samples/hive-ads/libs] \
  --applications Name=Hive \
  --release-label emr-5.3.1 \
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large
  InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large

```

Example 20: To add Pig steps when creating a cluster

The following example adds Pig steps when creating a cluster. Pig steps required parameters are Type and Args. Pig steps optional parameters are Name and ActionOnFailure.

```

aws emr create-cluster \
  --steps Type=PIG,Name='Pig program',ActionOnFailure=CONTINUE,Args=[-f,s3://
  elasticmapreduce/samples/pig-apache/do-reports2.pig,-p,INPUT=s3://elasticmapreduce/
  samples/pig-apache/input,-p,OUTPUT=s3://mybucket/pig-apache/output] \

```

```
--applications Name=Pig \  
--release-label emr-5.3.1 \  
--instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large  
InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large
```

Example 21: To add bootstrap actions

The following `create-cluster` example runs two bootstrap actions defined as scripts that are stored in Amazon S3.

```
aws emr create-cluster \  
  --bootstrap-actions Path=s3://mybucket/  
myscript1,Name=BootstrapAction1,Args=[arg1,arg2] Path=s3://mybucket/  
myscript2,Name=BootstrapAction2,Args=[arg1,arg2] \  
  --release-label emr-5.3.1 \  
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large  
InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large \  
  --auto-terminate
```

Example 22: To enable EMRFS consistent view and customize the `RetryCount` and `RetryPeriod` settings

The following `create-cluster` example specifies the retry count and retry period for EMRFS consistent view. The `Consistent=true` argument is required.

```
aws emr create-cluster \  
  --instance-type m4.large \  
  --release-label emr-5.9.0 \  
  --emrfs Consistent=true,RetryCount=6,RetryPeriod=30
```

The following example specifies the same EMRFS configuration as the previous example, using a locally stored JSON configuration file named `emrfsconfig.json`.

```
aws emr create-cluster \  
  --instance-type m4.large \  
  --release-label emr-5.9.0 \  
  --emrfs file://emrfsconfig.json
```

Contents of `emrfsconfig.json`:

```
{
```

```
"Consistent": true,  
"RetryCount": 6,  
"RetryPeriod": 30  
}
```

Example 23: To create a cluster with Kerberos configured

The following `create-cluster` examples create a cluster using a security configuration with Kerberos enabled, and establishes Kerberos parameters for the cluster using `--kerberos-attributes`.

The following command specifies Kerberos attributes for the cluster inline.

```
aws emr create-cluster \  
  --instance-type m3.xlarge \  
  --release-label emr-5.10.0 \  
  --service-role EMR_DefaultRole \  
  --ec2-attributes InstanceProfile=EMR_EC2_DefaultRole \  
  --security-configuration mySecurityConfiguration \  
  --kerberos-attributes  
  Realm=EC2.INTERNAL,KdcAdminPassword=123,CrossRealmTrustPrincipalPassword=123
```

The following command specifies the same attributes, but references a locally stored JSON file named `kerberos_attributes.json`. In this example, the file is saved in the same directory where you run the command. You can also reference a configuration file saved in Amazon S3.

```
aws emr create-cluster \  
  --instance-type m3.xlarge \  
  --release-label emr-5.10.0 \  
  --service-role EMR_DefaultRole \  
  --ec2-attributes InstanceProfile=EMR_EC2_DefaultRole \  
  --security-configuration mySecurityConfiguration \  
  --kerberos-attributes file://kerberos_attributes.json
```

Contents of `kerberos_attributes.json`:

```
{  
  "Realm": "EC2.INTERNAL",  
  "KdcAdminPassword": "123",  
  "CrossRealmTrustPrincipalPassword": "123",  
}
```

The following `create-cluster` example creates an Amazon EMR cluster that uses the `--instance-groups` configuration and has a managed scaling policy.

```
aws emr create-cluster \
  --release-label emr-5.30.0 \
  --service-role EMR_DefaultRole \
  --ec2-attributes InstanceProfile=EMR_EC2_DefaultRole \
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large
InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large
  --managed-scaling-policy
ComputeLimits='{MinimumCapacityUnits=2,MaximumCapacityUnits=4,UnitType=Instances}'
```

The following `create-cluster` example creates an Amazon EMR cluster that uses the `--log-encryption-kms-key-id` to define KMS key ID utilized for Log encryption.

```
aws emr create-cluster \
  --release-label emr-5.30.0 \
  --log-uri s3://myBucket/myLog \
  --log-encryption-kms-key-id arn:aws:kms:us-east-1:110302272565:key/
dd559181-283e-45d7-99d1-66da348c4d33 \
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large
InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large
```

The following `create-cluster` example creates an Amazon EMR cluster that uses the `--placement-group-configs` configuration to place master nodes in a high-availability (HA) cluster within an EC2 placement group using SPREAD placement strategy.

```
aws emr create-cluster \
  --release-label emr-5.30.0 \
  --service-role EMR_DefaultRole \
  --ec2-attributes InstanceProfile=EMR_EC2_DefaultRole \
  --instance-groups
InstanceGroupType=MASTER,InstanceCount=3,InstanceType=m4.largeInstanceGroupType=CORE,Instan
\
  --placement-group-configs InstanceRole=MASTER
```

The following `create-cluster` example creates an Amazon EMR cluster that uses the `--auto-termination-policy` configuration to place an automatic idle termination threshold for the cluster.

```
aws emr create-cluster \
```

```
--release-label emr-5.34.0 \  
--service-role EMR_DefaultRole \  
--ec2-attributes InstanceProfile=EMR_EC2_DefaultRole \  
--instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large  
InstanceGroupType=CORE,InstanceCount=1,InstanceType=m4.large \  
--auto-termination-policy IdleTimeout=100
```

The following `create-cluster` example creates an Amazon EMR cluster that uses the "`--os-release-label`" to define an Amazon Linux release for cluster launch

```
aws emr create-cluster \  
  --release-label emr-6.6.0 \  
  --os-release-label 2.0.20220406.1 \  
  --service-role EMR_DefaultRole \  
  --ec2-attributes InstanceProfile=EMR_EC2_DefaultRole \  
  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large  
  InstanceGroupType=CORE,InstanceCount=1,InstanceType=m4.large
```

Example 24: To specify an EBS root volume attributes: size, iops and throughput for cluster instances created with EMR releases 6.15.0 and later

The following `create-cluster` example creates an Amazon EMR cluster that uses root volume attributes to configure root volumes specifications for the EC2 instances.

```
aws emr create-cluster \  
  --name "Cluster with My Custom AMI" \  
  --custom-ami-id ami-a518e6df \  
  --ebs-root-volume-size 20 \  
  --ebs-root-volume-iops 3000 \  
  --ebs-root-volume-throughput 125 \  
  --release-label emr-6.15.0 \  
  --use-default-roles \  
  --instance-count 2 \  
  --instance-type m4.large
```

- For API details, see [CreateClusterExamples](#) in *AWS CLI Command Reference*.

create-default-roles

The following code example shows how to use `create-default-roles`.

AWS CLI

1. To create the default IAM role for EC2

Command:

```
aws emr create-default-roles
```

Output:

If the role already exists then the command returns nothing.

If the role does not exist then the output will be:

```
[
  {
    "RolePolicy": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Action": [
            "cloudwatch:*",
            "dynamodb:*",
            "ec2:Describe*",
            "elasticmapreduce:Describe*",
            "elasticmapreduce:ListBootstrapActions",
            "elasticmapreduce:ListClusters",
            "elasticmapreduce:ListInstanceGroups",
            "elasticmapreduce:ListInstances",
            "elasticmapreduce:ListSteps",
            "kinesis:CreateStream",
            "kinesis>DeleteStream",
            "kinesis:DescribeStream",
            "kinesis:GetRecords",
            "kinesis:GetShardIterator",
            "kinesis:MergeShards",
            "kinesis:PutRecord",
            "kinesis:SplitShard",
            "rds:Describe*",
            "s3:*",
            "sdb:*",
            "sns:*",
            "sqs:*"
          ]
        }
      ]
    }
  }
]
```



```

        ],
        "Resource": "*",
        "Effect": "Allow"
    }
]
},
"Role": {
    "AssumeRolePolicyDocument": {
        "Version": "2008-10-17",
        "Statement": [
            {
                "Action": "sts:AssumeRole",
                "Sid": "",
                "Effect": "Allow",
                "Principal": {
                    "Service": "ec2.amazonaws.com"
                }
            }
        ]
    },
    "RoleId": "AR0AIQ5SIUGL5KMYBJX6",
    "CreateDate": "2015-06-09T17:09:04.602Z",
    "RoleName": "EMR_EC2_DefaultRole",
    "Path": "/",
    "Arn": "arn:aws:iam::176430881729:role/EMR_EC2_DefaultRole"
}
},
{
    "RolePolicy": {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Action": [
                    "ec2:AuthorizeSecurityGroupIngress",
                    "ec2:CancelSpotInstanceRequests",
                    "ec2:CreateSecurityGroup",
                    "ec2:CreateTags",
                    "ec2>DeleteTags",
                    "ec2:DescribeAvailabilityZones",
                    "ec2:DescribeAccountAttributes",
                    "ec2:DescribeInstances",
                    "ec2:DescribeInstanceStatus",
                    "ec2:DescribeKeyPairs",
                    "ec2:DescribePrefixLists",

```

```
        "ec2:DescribeRouteTables",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSpotInstanceRequests",
        "ec2:DescribeSpotPriceHistory",
        "ec2:DescribeSubnets",
        "ec2:DescribeVpcAttribute",
        "ec2:DescribeVpcEndpoints",
        "ec2:DescribeVpcEndpointServices",
        "ec2:DescribeVpcs",
        "ec2:ModifyImageAttribute",
        "ec2:ModifyInstanceAttribute",
        "ec2:RequestSpotInstances",
        "ec2:RunInstances",
        "ec2:TerminateInstances",
        "iam:GetRole",
        "iam:GetRolePolicy",
        "iam>ListInstanceProfiles",
        "iam>ListRolePolicies",
        "iam:PassRole",
        "s3>CreateBucket",
        "s3:Get*",
        "s3>List*",
        "sdb:BatchPutAttributes",
        "sdb>Select",
        "sqs>CreateQueue",
        "sqs>Delete*",
        "sqs:GetQueue*",
        "sqs:ReceiveMessage"
    ],
    "Resource": "*",
    "Effect": "Allow"
  }
]
},
"Role": {
  "AssumeRolePolicyDocument": {
    "Version": "2008-10-17",
    "Statement": [
      {
        "Action": "sts:AssumeRole",
        "Sid": "",
        "Effect": "Allow",
        "Principal": {
          "Service": "elasticmapreduce.amazonaws.com"
```

```

    }
  }
]
},
"RoleId": "AROAI3SRVPPVSRDLARBPY",
"CreateDate": "2015-06-09T17:09:10.401Z",
"RoleName": "EMR_DefaultRole",
"Path": "/",
"Arn": "arn:aws:iam::176430881729:role/EMR_DefaultRole"
}
}
]

```

- For API details, see [CreateDefaultRoles](#) in *AWS CLI Command Reference*.

create-security-configuration

The following code example shows how to use create-security-configuration.

AWS CLI

1. To create a security configuration with in-transit encryption enabled with PEM for certificate provider, and at-rest encryption enabled with SSE-S3 for S3 encryption and AWS-KMS for local disk key provider

Command:

```

aws emr create-security-configuration --name MySecurityConfig --security-
configuration '{
  "EncryptionConfiguration": {
    "EnableInTransitEncryption" : true,
    "EnableAtRestEncryption" : true,
    "InTransitEncryptionConfiguration" : {
      "TLSCertificateConfiguration" : {
        "CertificateProviderType" : "PEM",
        "S3Object" : "s3://mycertstore/artifacts/
MyCerts.zip"
      }
    },
    "AtRestEncryptionConfiguration" : {
      "S3EncryptionConfiguration" : {
        "EncryptionMode" : "SSE-S3"
      }
    }
  },
}

```

```

        "LocalDiskEncryptionConfiguration" : {
            "EncryptionKeyProviderType" : "AwsKms",
            "AwsKmsKey" : "arn:aws:kms:us-
east-1:123456789012:key/12345678-1234-1234-1234-123456789012"
        }
    }
}'

```

Output:

```

{
  "CreationDateTime": 1474070889.129,
  "Name": "MySecurityConfig"
}

```

JSON equivalent (contents of security_configuration.json):

```

{
  "EncryptionConfiguration": {
    "EnableInTransitEncryption": true,
    "EnableAtRestEncryption": true,
    "InTransitEncryptionConfiguration": {
      "TLSCertificateConfiguration": {
        "CertificateProviderType": "PEM",
        "S3Object": "s3://mycertstore/artifacts/MyCerts.zip"
      }
    },
    "AtRestEncryptionConfiguration": {
      "S3EncryptionConfiguration": {
        "EncryptionMode": "SSE-S3"
      },
      "LocalDiskEncryptionConfiguration": {
        "EncryptionKeyProviderType": "AwsKms",
        "AwsKmsKey": "arn:aws:kms:us-
east-1:123456789012:key/12345678-1234-1234-1234-123456789012"
      }
    }
  }
}

```

Command (using security_configuration.json):

```
aws emr create-security-configuration --name "MySecurityConfig" --security-configuration file:///./security_configuration.json
```

Output:

```
{
  "CreationDateTime": 1474070889.129,
  "Name": "MySecurityConfig"
}
```

2. To create a security configuration with Kerberos enabled using cluster-dedicated KDC and cross-realm trust

Command:

```
aws emr create-security-configuration --name MySecurityConfig --security-configuration '{
  "AuthenticationConfiguration": {
    "KerberosConfiguration": {
      "Provider": "ClusterDedicatedKdc",
      "ClusterDedicatedKdcConfiguration": {
        "TicketLifetimeInHours": 24,
        "CrossRealmTrustConfiguration": {
          "Realm": "AD.DOMAIN.COM",
          "Domain": "ad.domain.com",
          "AdminServer": "ad.domain.com",
          "KdcServer": "ad.domain.com"
        }
      }
    }
  }
}'
```

Output:

```
{
  "CreationDateTime": 1490225558.982,
  "Name": "MySecurityConfig"
}
```

JSON equivalent (contents of security_configuration.json):

```
{
  "AuthenticationConfiguration": {
    "KerberosConfiguration": {
      "Provider": "ClusterDedicatedKdc",
      "ClusterDedicatedKdcConfiguration": {
        "TicketLifetimeInHours": 24,
        "CrossRealmTrustConfiguration": {
          "Realm": "AD.DOMAIN.COM",
          "Domain": "ad.domain.com",
          "AdminServer": "ad.domain.com",
          "KdcServer": "ad.domain.com"
        }
      }
    }
  }
}
```

Command (using security_configuration.json):

```
aws emr create-security-configuration --name "MySecurityConfig" --security-configuration file:///./security_configuration.json
```

Output:

```
{
  "CreationDateTime": 1490225558.982,
  "Name": "MySecurityConfig"
}
```

- For API details, see [CreateSecurityConfiguration](#) in *AWS CLI Command Reference*.

delete-security-configuration

The following code example shows how to use delete-security-configuration.

AWS CLI

To delete a security configuration in the current region

Command:

```
aws emr delete-security-configuration --name MySecurityConfig
```

Output:

```
None
```

- For API details, see [DeleteSecurityConfiguration](#) in *AWS CLI Command Reference*.

describe-cluster

The following code example shows how to use `describe-cluster`.

AWS CLI**Command:**

```
aws emr describe-cluster --cluster-id j-XXXXXXXX
```

Output:

For release-label based uniform instance groups cluster:

```
{
  "Cluster": {
    "Status": {
      "Timeline": {
        "ReadyDateTime": 1436475075.199,
        "CreationDateTime": 1436474656.563,
      },
      "State": "WAITING",
      "StateChangeReason": {
        "Message": "Waiting for steps to run"
      }
    },
    "Ec2InstanceAttributes": {
      "ServiceAccessSecurityGroup": "sg-xxxxxxx",
      "EmrManagedMasterSecurityGroup": "sg-xxxxxxx",
      "IamInstanceProfile": "EMR_EC2_DefaultRole",
      "Ec2KeyName": "myKey",
      "Ec2AvailabilityZone": "us-east-1c",
      "EmrManagedSlaveSecurityGroup": "sg-yyyyyyyyy"
    }
  }
}
```

```
    },
    "Name": "My Cluster",
    "ServiceRole": "EMR_DefaultRole",
    "Tags": [],
    "TerminationProtected": true,
    "UnhealthyNodeReplacement": true,
    "ReleaseLabel": "emr-4.0.0",
    "NormalizedInstanceHours": 96,
    "InstanceGroups": [
      {
        "RequestedInstanceCount": 2,
        "Status": {
          "Timeline": {
            "ReadyDateTime": 1436475074.245,
            "CreationDateTime": 1436474656.564,
            "EndDateTime": 1436638158.387
          },
          "State": "RUNNING",
          "StateChangeReason": {
            "Message": ""
          }
        },
        "Name": "CORE",
        "InstanceGroupType": "CORE",
        "Id": "ig-YYYYYYYY",
        "Configurations": [],
        "InstanceType": "m3.large",
        "Market": "ON_DEMAND",
        "RunningInstanceCount": 2
      },
      {
        "RequestedInstanceCount": 1,
        "Status": {
          "Timeline": {
            "ReadyDateTime": 1436475074.245,
            "CreationDateTime": 1436474656.564,
            "EndDateTime": 1436638158.387
          },
          "State": "RUNNING",
          "StateChangeReason": {
            "Message": ""
          }
        },
        "Name": "MASTER",
```



```

        "InstanceGroupType": "MASTER",
        "Id": "ig-XXXXXXXXXX",
        "Configurations": [],
        "InstanceType": "m3.large",
        "Market": "ON_DEMAND",
        "RunningInstanceCount": 1
    }
],
"Applications": [
    {
        "Name": "Hadoop"
    }
],
"VisibleToAllUsers": true,
"BootstrapActions": [],
"MasterPublicDnsName": "ec2-54-147-144-78.compute-1.amazonaws.com",
"AutoTerminate": false,
"Id": "j-XXXXXXXXXX",
"Configurations": [
    {
        "Properties": {
            "fs.s3.consistent.retryPeriodSeconds": "20",
            "fs.s3.enableServerSideEncryption": "true",
            "fs.s3.consistent": "false",
            "fs.s3.consistent.retryCount": "2"
        },
        "Classification": "emrfs-site"
    }
]
}
}
}

```

For release-label based instance fleet cluster:

```

{
  "Cluster": {
    "Status": {
      "Timeline": {
        "ReadyDateTime": 1487897289.705,
        "CreationDateTime": 1487896933.942
      },
      "State": "WAITING",
      "StateChangeReason": {
        "Message": "Waiting for steps to run"
      }
    }
  }
}

```

```

    }
  },
  "Ec2InstanceAttributes": {
    "EmrManagedMasterSecurityGroup": "sg-xxxxx",
    "RequestedEc2AvailabilityZones": [],
    "RequestedEc2SubnetIds": [],
    "IamInstanceProfile": "EMR_EC2_DefaultRole",
    "Ec2AvailabilityZone": "us-east-1a",
    "EmrManagedSlaveSecurityGroup": "sg-xxxxx"
  },
  "Name": "My Cluster",
  "ServiceRole": "EMR_DefaultRole",
  "Tags": [],
  "TerminationProtected": false,
  "UnhealthyNodeReplacement": false,
  "ReleaseLabel": "emr-5.2.0",
  "NormalizedInstanceHours": 472,
  "InstanceCollectionType": "INSTANCE_FLEET",
  "InstanceFleets": [
    {
      "Status": {
        "Timeline": {
          "ReadyDateTime": 1487897212.74,
          "CreationDateTime": 1487896933.948
        },
        "State": "RUNNING",
        "StateChangeReason": {
          "Message": ""
        }
      },
      "ProvisionedSpotCapacity": 1,
      "Name": "MASTER",
      "InstanceFleetType": "MASTER",
      "LaunchSpecifications": {
        "SpotSpecification": {
          "TimeoutDurationMinutes": 60,
          "TimeoutAction": "TERMINATE_CLUSTER"
        }
      },
      "TargetSpotCapacity": 1,
      "ProvisionedOnDemandCapacity": 0,
      "InstanceTypeSpecifications": [
        {
          "BidPrice": "0.5",

```

```

        "InstanceType": "m3.xlarge",
        "WeightedCapacity": 1
    }
],
    "Id": "if-xxxxxxx",
    "TargetOnDemandCapacity": 0
}
],
"Applications": [
    {
        "Version": "2.7.3",
        "Name": "Hadoop"
    }
],
"ScaleDownBehavior": "TERMINATE_AT_INSTANCE_HOUR",
"VisibleToAllUsers": true,
"BootstrapActions": [],
"MasterPublicDnsName": "ec2-xxx-xx-xxx-xx.compute-1.amazonaws.com",
"AutoTerminate": false,
"Id": "j-xxxxx",
"Configurations": []
}
}

```

For ami based uniform instance group cluster:

```

{
    "Cluster": {
        "Status": {
            "Timeline": {
                "ReadyDateTime": 1399400564.432,
                "CreationDateTime": 1399400268.62
            },
            "State": "WAITING",
            "StateChangeReason": {
                "Message": "Waiting for steps to run"
            }
        },
        "Ec2InstanceAttributes": {
            "IamInstanceProfile": "EMR_EC2_DefaultRole",
            "Ec2AvailabilityZone": "us-east-1c"
        },
        "Name": "My Cluster",
        "Tags": [],
    }
}

```

```
"TerminationProtected": true,
"UnhealthyNodeReplacement": true,
"RunningAmiVersion": "2.5.4",
"InstanceGroups": [
  {
    "RequestedInstanceCount": 1,
    "Status": {
      "Timeline": {
        "ReadyDateTime": 1399400558.848,
        "CreationDateTime": 1399400268.621
      },
      "State": "RUNNING",
      "StateChangeReason": {
        "Message": ""
      }
    },
    "Name": "Master instance group",
    "InstanceGroupType": "MASTER",
    "InstanceType": "m1.small",
    "Id": "ig-ABCD",
    "Market": "ON_DEMAND",
    "RunningInstanceCount": 1
  },
  {
    "RequestedInstanceCount": 2,
    "Status": {
      "Timeline": {
        "ReadyDateTime": 1399400564.439,
        "CreationDateTime": 1399400268.621
      },
      "State": "RUNNING",
      "StateChangeReason": {
        "Message": ""
      }
    },
    "Name": "Core instance group",
    "InstanceGroupType": "CORE",
    "InstanceType": "m1.small",
    "Id": "ig-DEF",
    "Market": "ON_DEMAND",
    "RunningInstanceCount": 2
  }
],
"Applications": [
```

```

        {
            "Version": "1.0.3",
            "Name": "hadoop"
        }
    ],
    "BootstrapActions": [],
    "VisibleToAllUsers": false,
    "RequestedAmiVersion": "2.4.2",
    "LogUri": "s3://myLogUri/",
    "AutoTerminate": false,
    "Id": "j-XXXXXXXX"
}
}

```

- For API details, see [DescribeCluster](#) in *AWS CLI Command Reference*.

describe-step

The following code example shows how to use describe-step.

AWS CLI

The following command describes a step with the step ID s-3LZC0QUT43AM in a cluster with the cluster ID j-3SD91U2E1L2QX:

```
aws emr describe-step --cluster-id j-3SD91U2E1L2QX --step-id s-3LZC0QUT43AM
```

Output:

```

{
  "Step": {
    "Status": {
      "Timeline": {
        "EndTime": 1433200470.481,
        "CreationDateTime": 1433199926.597,
        "StartTime": 1433200404.959
      },
      "State": "COMPLETED",
      "StateChangeReason": {}
    },
    "Config": {
      "Args": [

```

```

        "s3://us-west-2.elasticmapreduce/libs/hive/hive-script",
        "--base-path",
        "s3://us-west-2.elasticmapreduce/libs/hive/",
        "--install-hive",
        "--hive-versions",
        "0.13.1"
    ],
    "Jar": "s3://us-west-2.elasticmapreduce/libs/script-runner/script-
runner.jar",
    "Properties": {}
},
    "Id": "s-3LZC0QUT43AM",
    "ActionOnFailure": "TERMINATE_CLUSTER",
    "Name": "Setup hive"
}
}

```

- For API details, see [DescribeStep](#) in *AWS CLI Command Reference*.

get

The following code example shows how to use `get`.

AWS CLI

The following downloads the `hadoop-examples.jar` archive from the master instance in a cluster with the cluster ID `j-3SD91U2E1L2QX`:

```
aws emr get --cluster-id j-3SD91U2E1L2QX --key-pair-file ~/.ssh/mykey.pem --src /
home/hadoop-examples.jar --dest ~
```

- For API details, see [Get](#) in *AWS CLI Command Reference*.

list-clusters

The following code example shows how to use `list-clusters`.

AWS CLI

The following command lists all active EMR clusters in the current region:

```
aws emr list-clusters --active
```

Output:

```
{
  "Clusters": [
    {
      "Status": {
        "Timeline": {
          "ReadyDateTime": 1433200405.353,
          "CreationDateTime": 1433199926.596
        },
        "State": "WAITING",
        "StateChangeReason": {
          "Message": "Waiting after step completed"
        }
      },
      "NormalizedInstanceHours": 6,
      "Id": "j-3SD91U2E1L2QX",
      "Name": "my-cluster"
    }
  ]
}
```

- For API details, see [ListClusters](#) in *AWS CLI Command Reference*.

list-instance-fleets

The following code example shows how to use `list-instance-fleets`.

AWS CLI**To get configuration details of instance fleets in a cluster**

This example lists the details of instance fleets in the cluster specified.

Command:

```
list-instance-fleets --cluster-id 'j-12ABCDEFGH134JK'
```

Output:

```
{
  "InstanceFleets": [
    {
      "Status": {
        "Timeline": {
          "ReadyDateTime": 1488759094.637,
          "CreationDateTime": 1488758719.817
        },
        "State": "RUNNING",
        "StateChangeReason": {
          "Message": ""
        }
      },
      "ProvisionedSpotCapacity": 6,
      "Name": "CORE",
      "InstanceFleetType": "CORE",
      "LaunchSpecifications": {
        "SpotSpecification": {
          "TimeoutDurationMinutes": 60,
          "TimeoutAction": "TERMINATE_CLUSTER"
        }
      },
      "ProvisionedOnDemandCapacity": 2,
      "InstanceTypeSpecifications": [
        {
          "BidPrice": "0.5",
          "InstanceType": "m3.xlarge",
          "WeightedCapacity": 2
        }
      ],
      "Id": "if-1ABC2DEFGHIJ3"
    },
    {
      "Status": {
        "Timeline": {
          "ReadyDateTime": 1488759058.598,
          "CreationDateTime": 1488758719.811
        },
        "State": "RUNNING",
        "StateChangeReason": {
          "Message": ""
        }
      }
    }
  ],
}
```



```

    "ProvisionedSpotCapacity": 0,
    "Name": "MASTER",
    "InstanceFleetType": "MASTER",
    "ProvisionedOnDemandCapacity": 1,
    "InstanceTypeSpecifications": [
      {
        "BidPriceAsPercentageOfOnDemandPrice": 100.0,
        "InstanceType": "m3.xlarge",
        "WeightedCapacity": 1
      }
    ],
    "Id": "if-2ABC4DEFGHIJ4"
  }
]
}

```

- For API details, see [ListInstanceFleets](#) in *AWS CLI Command Reference*.

list-instances

The following code example shows how to use `list-instances`.

AWS CLI

The following command lists all of the instances in a cluster with the cluster ID `j-3C6XNQ39VR9WL`:

```
aws emr list-instances --cluster-id j-3C6XNQ39VR9WL
```

Output:

```

For a uniform instance group based cluster
{
  "Instances": [
    {
      "Status": {
        "Timeline": {
          "ReadyDateTime": 1433200400.03,
          "CreationDateTime": 1433199960.152
        },
        "State": "RUNNING",
        "StateChangeReason": {}
      }
    }
  ]
}

```

```
    },
    "Ec2InstanceId": "i-f19ecfee",
    "PublicDnsName": "ec2-52-52-41-150.us-west-2.compute.amazonaws.com",
    "PrivateDnsName": "ip-172-21-11-216.us-west-2.compute.internal",
    "PublicIpAddress": "52.52.41.150",
    "Id": "ci-3NNHQ2TWB6Y",
    "PrivateIpAddress": "172.21.11.216"
  },
  {
    "Status": {
      "Timeline": {
        "ReadyDateTime": 1433200400.031,
        "CreationDateTime": 1433199949.102
      },
      "State": "RUNNING",
      "StateChangeReason": {}
    },
    "Ec2InstanceId": "i-1f1ee4c2",
    "PublicDnsName": "ec2-52-63-246-32.us-west-2.compute.amazonaws.com",
    "PrivateDnsName": "ip-172-31-24-130.us-west-2.compute.internal",
    "PublicIpAddress": "52.63.246.32",
    "Id": "ci-GAOCMKNKDCV7",
    "PrivateIpAddress": "172.21.11.215"
  },
  {
    "Status": {
      "Timeline": {
        "ReadyDateTime": 1433200400.031,
        "CreationDateTime": 1433199949.102
      },
      "State": "RUNNING",
      "StateChangeReason": {}
    },
    "Ec2InstanceId": "i-15cfee3",
    "PublicDnsName": "ec2-52-25-246-63.us-west-2.compute.amazonaws.com",
    "PrivateDnsName": "ip-172-31-24-129.us-west-2.compute.internal",
    "PublicIpAddress": "52.25.246.63",
    "Id": "ci-2W3TDFFB47UAD",
    "PrivateIpAddress": "172.21.11.214"
  }
]
}
```

For a fleet based cluster:

```
{
  "Instances": [
    {
      "Status": {
        "Timeline": {
          "ReadyDateTime": 1487810810.878,
          "CreationDateTime": 1487810588.367,
          "EndDateTime": 1488022990.924
        },
        "State": "TERMINATED",
        "StateChangeReason": {
          "Message": "Instance was terminated."
        }
      },
      "Ec2InstanceId": "i-xxxxx",
      "InstanceFleetId": "if-xxxxx",
      "EbsVolumes": [],
      "PublicDnsName": "ec2-xx-xxx-xxx-xxx.compute-1.amazonaws.com",
      "InstanceType": "m3.xlarge",
      "PrivateDnsName": "ip-xx-xx-xxx-xx.ec2.internal",
      "Market": "SPOT",
      "PublicIpAddress": "xx.xx.xxx.xxx",
      "Id": "ci-xxxxx",
      "PrivateIpAddress": "10.47.191.80"
    }
  ]
}
```

- For API details, see [ListInstances](#) in *AWS CLI Command Reference*.

list-security-configurations

The following code example shows how to use `list-security-configurations`.

AWS CLI

To list security configurations in the current region

Command:

```
aws emr list-security-configurations
```

Output:

```
{
  "SecurityConfigurations": [
    {
      "CreationDateTime": 1473889697.417,
      "Name": "MySecurityConfig-1"
    },
    {
      "CreationDateTime": 1473889697.417,
      "Name": "MySecurityConfig-2"
    }
  ]
}
```

- For API details, see [ListSecurityConfigurations](#) in *AWS CLI Command Reference*.

list-steps

The following code example shows how to use `list-steps`.

AWS CLI

The following command lists all of the steps in a cluster with the cluster ID `j-3SD91U2E1L2QX`:

```
aws emr list-steps --cluster-id j-3SD91U2E1L2QX
```

- For API details, see [ListSteps](#) in *AWS CLI Command Reference*.

modify-cluster-attributes

The following code example shows how to use `modify-cluster-attributes`.

AWS CLI

The following command sets the visibility of an EMR cluster with the ID `j-301CDNY0J5XM4` to all users:

```
aws emr modify-cluster-attributes --cluster-id j-301CDNY0J5XM4 --visible-to-all-users
```

- For API details, see [ModifyClusterAttributes](#) in *AWS CLI Command Reference*.

modify-instance-fleet

The following code example shows how to use `modify-instance-fleet`.

AWS CLI

To change the target capacities of an instance fleet

This example changes the On-Demand and Spot target capacities to 1 for the instance fleet specified.

Command:

```
aws emr modify-instance-fleet --cluster-id 'j-12ABCDEFGH134JK' --instance-fleet InstanceFleetId='if-2ABC4DEFGHIJ4',TargetOnDemandCapacity=1,TargetSpotCapacity=1
```

- For API details, see [ModifyInstanceFleet](#) in *AWS CLI Command Reference*.

put

The following code example shows how to use `put`.

AWS CLI

The following command uploads a file named `healthcheck.sh` to the master instance in a cluster with the cluster ID `j-3SD91U2E1L2QX`:

```
aws emr put --cluster-id j-3SD91U2E1L2QX --key-pair-file ~/.ssh/mykey.pem --src ~/scripts/healthcheck.sh --dest /home/hadoop/bin/healthcheck.sh
```

- For API details, see [Put](#) in *AWS CLI Command Reference*.

remove-tags

The following code example shows how to use `remove-tags`.

AWS CLI

The following command removes a tag with the key `prod` from a cluster with the cluster ID `j-3SD91U2E1L2QX`:

```
aws emr remove-tags --resource-id j-3SD91U2E1L2QX --tag-keys prod
```

- For API details, see [RemoveTags](#) in *AWS CLI Command Reference*.

schedule-hbase-backup

The following code example shows how to use `schedule-hbase-backup`.

AWS CLI

Note: This command can only be used with HBase on AMI version 2.x and 3.x

1. To schedule a full HBase backup >>>>>> 06ab6d6e13564b5733d75abaf3b599f93cf39a23

Command:

```
aws emr schedule-hbase-backup --cluster-id j-XXXXXXYY --type full --dir  
s3://myBucket/backup --interval 10 --unit hours --start-time  
2014-04-21T05:26:10Z --consistent
```

Output:

```
None
```

2. To schedule an incremental HBase backup

Command:

```
aws emr schedule-hbase-backup --cluster-id j-XXXXXXYY --type incremental  
--dir s3://myBucket/backup --interval 30 --unit minutes --start-time  
2014-04-21T05:26:10Z --consistent
```

Output:

```
None
```

- For API details, see [ScheduleHbaseBackup](#) in *AWS CLI Command Reference*.

socks

The following code example shows how to use socks.

AWS CLI

The following command opens a socks connection with the master instance in a cluster with the cluster ID j-3SD91U2E1L2QX:

```
aws emr socks --cluster-id j-3SD91U2E1L2QX --key-pair-file ~/.ssh/mykey.pem
```

The key pair file option takes a local path to a private key file.

- For API details, see [Socks](#) in *AWS CLI Command Reference*.

ssh

The following code example shows how to use ssh.

AWS CLI

The following command opens an ssh connection with the master instance in a cluster with the cluster ID j-3SD91U2E1L2QX:

```
aws emr ssh --cluster-id j-3SD91U2E1L2QX --key-pair-file ~/.ssh/mykey.pem
```

The key pair file option takes a local path to a private key file.

Output:

```
ssh -o StrictHostKeyChecking=no -o ServerAliveInterval=10 -i /home/local/user/.ssh/mykey.pem hadoop@ec2-52-52-41-150.us-west-2.compute.amazonaws.com
Warning: Permanently added 'ec2-52-52-41-150.us-west-2.compute.amazonaws.com,52.52.41.150' (ECDSA) to the list of known hosts.
Last login: Mon Jun  1 23:15:38 2015

  _|  _|_ )
  _| (    /  Amazon Linux AMI
```

```
—|\—|—|

https://aws.amazon.com/amazon-linux-ami/2015.03-release-notes/
26 package(s) needed for security, out of 39 available
Run "sudo yum update" to apply all updates.

-----

Welcome to Amazon Elastic MapReduce running Hadoop and Amazon Linux.

Hadoop is installed in /home/hadoop. Log files are in /mnt/var/log/hadoop. Check
/mnt/var/log/hadoop/steps for diagnosing step failures.

The Hadoop UI can be accessed via the following commands:

ResourceManager    lynx http://ip-172-21-11-216:9026/
NameNode           lynx http://ip-172-21-11-216:9101/

-----

[hadoop@ip-172-31-16-216 ~]$
```

- For API details, see [Ssh](#) in *AWS CLI Command Reference*.

Amazon EMR on EKS examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon EMR on EKS.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

update-role-trust-policy

The following code example shows how to use `update-role-trust-policy`.

AWS CLI

To update the trust policy of an IAM Role to be used with Amazon EMR on EKS

This example command updates the trust policy of a role named `example_iam_role` such that it can be used with Amazon EMR on EKS with `example_namespace` namespace from an EKS cluster named `example_cluster`.

Command:

```
aws emr-containers update-role-trust-policy \  
  --cluster example_cluster \  
  --namespace example_namespace \  
  --role-name example_iam_role
```

Output:

```
If the trust policy has already been updated, then the output will be:  
Trust policy statement already exists for role example_iam_role. No  
changes were made!
```

```
If the trust policy has not been updated yet, then the output will be:  
Successfully updated trust policy of role example_iam_role.
```

- For API details, see [UpdateRoleTrustPolicy](#) in *AWS CLI Command Reference*.

EventBridge examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with EventBridge.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

delete-rule

The following code example shows how to use `delete-rule`.

AWS CLI

To delete a CloudWatch Events rule

This example deletes the rule named `EC2InstanceStateChanges`:

```
aws events delete-rule --name "EC2InstanceStateChanges"
```

- For API details, see [DeleteRule](#) in *AWS CLI Command Reference*.

describe-rule

The following code example shows how to use `describe-rule`.

AWS CLI

To display information about a CloudWatch Events rule

This example displays information about the rule named `DailyLambdaFunction`:

```
aws events describe-rule --name "DailyLambdaFunction"
```

- For API details, see [DescribeRule](#) in *AWS CLI Command Reference*.

disable-rule

The following code example shows how to use `disable-rule`.

AWS CLI

To disable a CloudWatch Events rule

This example disables the rule named `DailyLambdaFunction`. The rule is not deleted:

```
aws events disable-rule --name "DailyLambdaFunction"
```

- For API details, see [DisableRule](#) in *AWS CLI Command Reference*.

enable-rule

The following code example shows how to use `enable-rule`.

AWS CLI

To enable a CloudWatch Events rule

This example enables the rule named `DailyLambdaFunction`, which had been previously disabled:

```
aws events enable-rule --name "DailyLambdaFunction"
```

- For API details, see [EnableRule](#) in *AWS CLI Command Reference*.

list-rule-names-by-target

The following code example shows how to use `list-rule-names-by-target`.

AWS CLI

To display all the rules that have a specified target

This example displays all rules that have the Lambda function named `"MyFunctionName"` as the target:

```
aws events list-rule-names-by-target --target-arn "arn:aws:lambda:us-east-1:123456789012:function:MyFunctionName"
```

- For API details, see [ListRuleNamesByTarget](#) in *AWS CLI Command Reference*.

list-rules

The following code example shows how to use `list-rules`.

AWS CLI

To display a list of all CloudWatch Events rules

This example displays all CloudWatch Events rules in the region:

```
aws events list-rules
```

To display a list of CloudWatch Events rules beginning with a certain string.

This example displays all CloudWatch Events rules in the region that have a name starting with "Daily":

```
aws events list-rules --name-prefix "Daily"
```

- For API details, see [ListRules](#) in *AWS CLI Command Reference*.

list-targets-by-rule

The following code example shows how to use `list-targets-by-rule`.

AWS CLI

To display all the targets for a CloudWatch Events rule

This example displays all the targets of the rule named `DailyLambdaFunction`:

```
aws events list-targets-by-rule --rule "DailyLambdaFunction"
```

- For API details, see [ListTargetsByRule](#) in *AWS CLI Command Reference*.

put-events

The following code example shows how to use `put-events`.

AWS CLI

To send a custom event to CloudWatch Events

This example sends a custom event to CloudWatch Events. The event is contained within the `putevents.json` file:

```
aws events put-events --entries file://putevents.json
```

Here are the contents of the `putevents.json` file:

```
[
  {
    "Source": "com.mycompany.myapp",
    "Detail": "{ \"key1\": \"value1\", \"key2\": \"value2\" }",
    "Resources": [
      "resource1",
      "resource2"
    ],
    "DetailType": "myDetailType"
  },
  {
    "Source": "com.mycompany.myapp",
    "Detail": "{ \"key1\": \"value3\", \"key2\": \"value4\" }",
    "Resources": [
      "resource1",
      "resource2"
    ],
    "DetailType": "myDetailType"
  }
]
```

- For API details, see [PutEvents](#) in *AWS CLI Command Reference*.

put-rule

The following code example shows how to use `put-rule`.

AWS CLI

To create CloudWatch Events rules

This example creates a rule that triggers every day at 9:00am (UTC). If you use `put-targets` to add a Lambda function as a target of this rule, you could run the Lambda function every day at the specified time:

```
aws events put-rule --name "DailyLambdaFunction" --schedule-expression "cron(0 9 * * ? *)"
```

This example creates a rule that triggers when any EC2 instance in the region changes state:

```
aws events put-rule --name "EC2InstanceStateChanges" --event-pattern "{\"source\": [\"aws.ec2\"], \"detail-type\": [\"EC2 Instance State-change Notification\"]}" --role-arn "arn:aws:iam::123456789012:role/MyRoleForThisRule"
```

This example creates a rule that triggers when any EC2 instance in the region is stopped or terminated:

```
aws events put-rule --name "EC2InstanceStateChangeStopOrTerminate" --event-pattern "{\"source\": [\"aws.ec2\"], \"detail-type\": [\"EC2 Instance State-change Notification\"], \"detail\": {\"state\": [\"stopped\", \"terminated\"]}}" --role-arn "arn:aws:iam::123456789012:role/MyRoleForThisRule"
```

- For API details, see [PutRule](#) in *AWS CLI Command Reference*.

put-targets

The following code example shows how to use `put-targets`.

AWS CLI

To add targets for CloudWatch Events rules

This example adds a Lambda function as the target of a rule:

```
aws events put-targets --rule DailyLambdaFunction --targets "Id"="1", "Arn"="arn:aws:lambda:us-east-1:123456789012:function:MyFunctionName"
```

This example sets an Amazon Kinesis stream as the target, so that events caught by this rule are relayed to the stream:

```
aws events put-targets --rule EC2InstanceStateChanges --targets
  "Id"="1", "Arn"="arn:aws:kinesis:us-east-1:123456789012:stream/
MyStream", "RoleArn"="arn:aws:iam::123456789012:role/MyRoleForThisRule"
```

This example sets two Amazon Kinesis streams as targets for one rule:

```
aws events put-targets --rule DailyLambdaFunction --targets
  "Id"="Target1", "Arn"="arn:aws:kinesis:us-east-1:379642911888:stream/
MyStream1", "RoleArn"="arn:aws:iam::379642911888:role/ MyRoleToAccessLambda"
  "Id"="Target2", " Arn"="arn:aws:kinesis:us-east-1:379642911888:stream/
MyStream2", "RoleArn"="arn:aws:iam::379642911888:role/MyRoleToAccessLambda"
```

- For API details, see [PutTargets](#) in *AWS CLI Command Reference*.

remove-targets

The following code example shows how to use `remove-targets`.

AWS CLI

To remove a target for an event

This example removes the Amazon Kinesis stream named `MyStream1` from being a target of the rule `DailyLambdaFunction`. When `DailyLambdaFunction` was created, this stream was set as a target with an ID of `Target1`:

```
aws events remove-targets --rule "DailyLambdaFunction" --ids "Target1"
```

- For API details, see [RemoveTargets](#) in *AWS CLI Command Reference*.

test-event-pattern

The following code example shows how to use `test-event-pattern`.

AWS CLI

To check whether an event pattern matches a specified event

This example tests whether the pattern `"source:com.mycompany.myapp"` matches the specified event. In this example, the output would be `"true"`:

```
aws events test-event-pattern --event-pattern "{\"source\":[\"com.mycompany.myapp\n\"]}" --event "{\"id\":"1","\source\":"com.mycompany.myapp","\detail-type\":"myDetailType","\account\":"123456789012","\region\":"us-east-1","\time\":"2017-04-11T20:11:04Z\"}"
```

- For API details, see [TestEventPattern](#) in *AWS CLI Command Reference*.

Firewall Manager examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Firewall Manager.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-admin-account

The following code example shows how to use `associate-admin-account`.

AWS CLI

To set the Firewall Manager administrator account

The following `associate-admin-account` example sets the administrator account for Firewall Manager.

```
aws fms associate-admin-account \  
  --admin-account 123456789012
```


This command produces no output.

For more information, see [Set the AWS Firewall Manager Administrator Account](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [AssociateAdminAccount](#) in *AWS CLI Command Reference*.

delete-notification-channel

The following code example shows how to use `delete-notification-channel`.

AWS CLI

To remove the SNS topic information for Firewall Manager logs

The following `delete-notification-channel` example removes the SNS topic information.

```
aws fms delete-notification-channel
```

This command produces no output.

For more information, see [Configure Amazon SNS Notifications and Amazon CloudWatch Alarms](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [DeleteNotificationChannel](#) in *AWS CLI Command Reference*.

delete-policy

The following code example shows how to use `delete-policy`.

AWS CLI

To delete a Firewall Manager policy

The following `delete-policy` example removes the policy with the specified ID, along with all of its resources.

```
aws fms delete-policy \  
  --policy-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \  
  --delete-all-policy-resources
```

This command produces no output.

For more information, see [Working with AWS Firewall Manager Policies](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [DeletePolicy](#) in *AWS CLI Command Reference*.

disassociate-admin-account

The following code example shows how to use `disassociate-admin-account`.

AWS CLI

To remove the Firewall Manager administrator account

The following `disassociate-admin-account` example removes the current administrator account association from Firewall Manager.

```
aws fms disassociate-admin-account
```

This command produces no output.

For more information, see [Set the AWS Firewall Manager Administrator Account](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [DisassociateAdminAccount](#) in *AWS CLI Command Reference*.

get-admin-account

The following code example shows how to use `get-admin-account`.

AWS CLI

To retrieve the Firewall Manager administrator account

The following `get-admin-account` example retrieves the administrator account.

```
aws fms get-admin-account
```

Output:

```
{
  "AdminAccount": "123456789012",
  "RoleStatus": "READY"
```

```
}
```

For more information, see [AWS Firewall Manager Prerequisites](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [GetAdminAccount](#) in *AWS CLI Command Reference*.

get-compliance-detail

The following code example shows how to use `get-compliance-detail`.

AWS CLI

To retrieve the compliance information for an account

The following `get-compliance-detail` example retrieves compliance information for the specified policy and member account.

```
aws fms get-compliance-detail \  
  --policy-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \  
  --member-account 123456789012
```

Output:

```
{  
  "PolicyComplianceDetail": {  
    "EvaluationLimitExceeded": false,  
    "IssueInfoMap": {},  
    "MemberAccount": "123456789012",  
    "PolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
    "PolicyOwner": "123456789012",  
    "Violators": []  
  }  
}
```

For more information, see [Viewing Resource Compliance with a Policy](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [GetComplianceDetail](#) in *AWS CLI Command Reference*.

get-notification-channel

The following code example shows how to use `get-notification-channel`.

AWS CLI

To retrieve the SNS topic information for Firewall Manager logs

The following `get-notification-channel` example retrieves the SNS topic information.

```
aws fms get-notification-channel
```

Output:

```
{
  "SnsTopicArn": "arn:aws:sns:us-west-2:123456789012:us-west-2-fms",
  "SnsRoleName": "arn:aws:iam::123456789012:role/aws-service-role/
fms.amazonaws.com/AWSServiceRoleForFMS"
}
```

For more information, see [Configure Amazon SNS Notifications and Amazon CloudWatch Alarms](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [GetNotificationChannel](#) in *AWS CLI Command Reference*.

get-policy

The following code example shows how to use `get-policy`.

AWS CLI

To retrieve a Firewall Manager policy

The following `get-policy` example retrieves the policy with the specified ID.

```
aws fms get-policy \
  --policy-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{
  "Policy": {
    "PolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "PolicyName": "test",
    "PolicyUpdateToken": "1:p+2RpKR4wPFx7mcrL1U0QQ==",
    "SecurityServicePolicyData": {
```

```

        "Type": "SECURITY_GROUPS_COMMON",
        "ManagedServiceData": "{\"type\":\"SECURITY_GROUPS_COMMON\",
\\revertManualSecurityGroupChanges\":true,\\exclusiveResourceSecurityGroupManagement
\\:false,\\securityGroups\":[\\id\\:\\sg-045c43ccc9724e63e\\]}\"
    },
    "ResourceType": "AWS::EC2::Instance",
    "ResourceTags": [],
    "ExcludeResourceTags": false,
    "RemediationEnabled": false
},
"PolicyArn": "arn:aws:fms:us-west-2:123456789012:policy/d1ac59b8-938e-42b3-
b2e0-7c620422ddc2"
}

```

For more information, see [Working with AWS Firewall Manager Policies](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [GetPolicy](#) in *AWS CLI Command Reference*.

list-compliance-status

The following code example shows how to use `list-compliance-status`.

AWS CLI

To retrieve the policy compliance information for member accounts

The following `list-compliance-status` example retrieves member account compliance information for the specified policy.

```

aws fms list-compliance-status \
  --policy-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111

```

Output:

```

{
  "PolicyComplianceStatusList": [
    {
      "PolicyOwner": "123456789012",
      "PolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "PolicyName": "test",
      "MemberAccount": "123456789012",
    }
  ]
}

```

```
    "EvaluationResults": [
      {
        "ComplianceStatus": "COMPLIANT",
        "ViolatorCount": 0,
        "EvaluationLimitExceeded": false
      },
      {
        "ComplianceStatus": "NON_COMPLIANT",
        "ViolatorCount": 2,
        "EvaluationLimitExceeded": false
      }
    ],
    "LastUpdated": 1576283774.0,
    "IssueInfoMap": {}
  }
]
```

For more information, see [Viewing Resource Compliance with a Policy](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [ListComplianceStatus](#) in *AWS CLI Command Reference*.

list-member-accounts

The following code example shows how to use `list-member-accounts`.

AWS CLI

To retrieve the member accounts in the organization

The following `list-member-accounts` example lists all of the member accounts that are in the Firewall Manager administrator's organization.

```
aws fms list-member-accounts
```

Output:

```
{
  "MemberAccounts": [
    "222222222222",
    "333333333333",
```

```

    "4444444444444444"
  ]
}

```

For more information, see [AWS Firewall Manager](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [ListMemberAccounts](#) in *AWS CLI Command Reference*.

list-policies

The following code example shows how to use `list-policies`.

AWS CLI

To retrieve all Firewall Manager policies

The following `list-policies` example retrieves the list of policies for the account. In this example, the output is limited to two results per request. Each call returns a `NextToken` that can be used as the value for the `--starting-token` parameter in the next `list-policies` call to get the next set of results for the list.

```

aws fms list-policies \
  --max-items 2

```

Output:

```

{
  "PolicyList": [
    {
      "PolicyArn": "arn:aws:fms:us-west-2:123456789012:policy/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "PolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "PolicyName": "test",
      "ResourceType": "AWS::EC2::Instance",
      "SecurityServiceType": "SECURITY_GROUPS_COMMON",
      "RemediationEnabled": false
    },
    {
      "PolicyArn": "arn:aws:fms:us-west-2:123456789012:policy/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
      "PolicyId": "457c9b21-fc94-406c-ae63-21217395ba72",

```

```
        "PolicyName": "test",
        "ResourceType": "AWS::EC2::Instance",
        "SecurityServiceType": "SECURITY_GROUPS_COMMON",
        "RemediationEnabled": false
    }
],
"NextToken": "eyJ0ZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAyfQ=="
}
```

For more information, see [Working with AWS Firewall Manager Policies](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [ListPolicies](#) in *AWS CLI Command Reference*.

put-notification-channel

The following code example shows how to use `put-notification-channel`.

AWS CLI

To set the SNS topic information for Firewall Manager logs

The following `put-notification-channel` example sets the SNS topic information.

```
aws fms put-notification-channel \
  --sns-topic-arn arn:aws:sns:us-west-2:123456789012:us-west-2-fms \
  --sns-role-name arn:aws:iam::123456789012:role/aws-service-role/
fms.amazonaws.com/AWSServiceRoleForFMS
```

This command produces no output.

For more information, see [Configure Amazon SNS Notifications and Amazon CloudWatch Alarms](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [PutNotificationChannel](#) in *AWS CLI Command Reference*.

put-policy

The following code example shows how to use `put-policy`.

AWS CLI

To create a Firewall Manager policy

The following `put-policy` example creates a Firewall Manager security group policy.

```
aws fms put-policy \
  --cli-input-json file://policy.json
```

Contents of `policy.json`:

```
{
  "Policy": {
    "PolicyName": "test",
    "SecurityServicePolicyData": {
      "Type": "SECURITY_GROUPS_USAGE_AUDIT",
      "ManagedServiceData": "{\"type\":\"SECURITY_GROUPS_USAGE_AUDIT\",
\\\"deleteUnusedSecurityGroups\\\":false,\\\"coalesceRedundantSecurityGroups\\\":true}"
    },
    "ResourceType": "AWS::EC2::SecurityGroup",
    "ResourceTags": [],
    "ExcludeResourceTags": false,
    "RemediationEnabled": false
  },
  "TagList": [
    {
      "Key": "foo",
      "Value": "foo"
    }
  ]
}
```

Output:

```
{
  "Policy": {
    "PolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "PolicyName": "test",
    "PolicyUpdateToken": "1:X9QGexP7HASDlsFp+G31Iw==",
    "SecurityServicePolicyData": {
      "Type": "SECURITY_GROUPS_USAGE_AUDIT",
      "ManagedServiceData": "{\"type\":\"SECURITY_GROUPS_USAGE_AUDIT\",
\\\"deleteUnusedSecurityGroups\\\":false,\\\"coalesceRedundantSecurityGroups\\\":true,
\\\"optionalDelayForUnusedInMinutes\\\":null}"
    },
    "ResourceType": "AWS::EC2::SecurityGroup",
```

```
    "ResourceTags": [],
    "ExcludeResourceTags": false,
    "RemediationEnabled": false
  },
  "PolicyArn": "arn:aws:fms:us-west-2:123456789012:policy/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
}
```

For more information, see [Working with AWS Firewall Manager Policies](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [PutPolicy](#) in *AWS CLI Command Reference*.

AWS FIS examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS FIS.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-experiment-template

The following code example shows how to use `create-experiment-template`.

AWS CLI

To create an experiment template

The following `create-experiment-template` example creates an experiment template in your AWS FIS account.

```
aws fis create-experiment-template \  
  --cli-input-json file://myfile.json
```

Contents of `myfile.json`:

```
{  
  "description": "experimentTemplate",  
  "stopConditions": [  
    {  
      "source": "aws:cloudwatch:alarm",  
      "value": "arn:aws:cloudwatch:us-west-2:123456789012:alarm:alarmName"  
    }  
  ],  
  "targets": {  
    "Instances-Target-1": {  
      "resourceType": "aws:ec2:instance",  
      "resourceArns": [  
        "arn:aws:ec2:us-west-2:123456789012:instance/i-12a3b4c56d78e9012"  
      ],  
      "selectionMode": "ALL"  
    }  
  },  
  "actions": {  
    "reboot": {  
      "actionId": "aws:ec2:reboot-instances",  
      "description": "reboot",  
      "parameters": {},  
      "targets": {  
        "Instances": "Instances-Target-1"  
      }  
    }  
  },  
  "roleArn": "arn:aws:iam::123456789012:role/myRole"  
}
```

Output:

```
{  
  "experimentTemplate": {
```

```

    "id": "ABCDE1fgHIJkLmNop",
    "description": "experimentTemplate",
    "targets": {
      "Instances-Target-1": {
        "resourceType": "aws:ec2:instance",
        "resourceArns": [
          "arn:aws:ec2:us-west-2:123456789012:instance/
i-12a3b4c56d78e9012"
        ],
        "selectionMode": "ALL"
      }
    },
    "actions": {
      "reboot": {
        "actionId": "aws:ec2:reboot-instances",
        "description": "reboot",
        "parameters": {},
        "targets": {
          "Instances": "Instances-Target-1"
        }
      }
    },
    "stopConditions": [
      {
        "source": "aws:cloudwatch:alarm",
        "value": "arn:aws:cloudwatch:us-west-2:123456789012:alarm:alarmName"
      }
    ],
    "creationTime": 1616434850.659,
    "lastUpdateTime": 1616434850.659,
    "roleArn": "arn:aws:iam::123456789012:role/myRole",
    "tags": {}
  }
}

```

For more information, see [Create an experiment template](#) in the *AWS Fault Injection Simulator User Guide*.

- For API details, see [CreateExperimentTemplate](#) in *AWS CLI Command Reference*.

delete-experiment-template

The following code example shows how to use delete-experiment-template.

AWS CLI

To delete an experiment template

The following `delete-experiment-template` example deletes the specified experiment template.

```
aws fis delete-experiment-template \  
  --id ABCDE1fgHIJkLmNop
```

Output:

```
{  
  "experimentTemplate": {  
    "id": "ABCDE1fgHIJkLmNop",  
    "description": "myExperimentTemplate",  
    "targets": {  
      "Instances-Target-1": {  
        "resourceType": "aws:ec2:instance",  
        "resourceArns": [  
          "arn:aws:ec2:us-west-2:123456789012:instance/  
i-12a3b4c56d78e9012"  
        ],  
        "selectionMode": "ALL"  
      }  
    },  
    "actions": {  
      "testaction": {  
        "actionId": "aws:ec2:stop-instances",  
        "parameters": {},  
        "targets": {  
          "Instances": "Instances-Target-1"  
        }  
      }  
    },  
    "stopConditions": [  
      {  
        "source": "none"  
      }  
    ],  
    "creationTime": 1616017191.124,  
    "lastUpdateTime": 1616017859.607,  
    "roleArn": "arn:aws:iam::123456789012:role/FISRole"
```

```
}  
}
```

For more information, see [Delete an experiment template](#) in the *AWS Fault Injection Simulator User Guide*.

- For API details, see [DeleteExperimentTemplate](#) in *AWS CLI Command Reference*.

get-action

The following code example shows how to use `get-action`.

AWS CLI

To get action details

The following `get-action` example gets the details of the specified action.

```
aws fis get-action \  
  --id aws:ec2:stop-instances
```

Output:

```
{  
  "action": {  
    "id": "aws:ec2:stop-instances",  
    "description": "Stop the specified EC2 instances.",  
    "parameters": {  
      "startInstancesAfterDuration": {  
        "description": "The time to wait before restarting the instances  
(ISO 8601 duration).",  
        "required": false  
      }  
    },  
    "targets": {  
      "Instances": {  
        "resourceType": "aws:ec2:instance"  
      }  
    },  
    "tags": {}  
  }  
}
```

For more information, see [Actions](#) in the *AWS Fault Injection Simulator User Guide*.

- For API details, see [GetAction](#) in *AWS CLI Command Reference*.

get-experiment-template

The following code example shows how to use `get-experiment-template`.

AWS CLI

To get experiment template details

The following `get-experiment-template` example gets the details of the specified experiment template.

```
aws fis get-experiment-template \  
  --id ABCDE1fgHIJkLmNop
```

Output:

```
{  
  "experimentTemplate": {  
    "id": "ABCDE1fgHIJkLmNop",  
    "description": "myExperimentTemplate",  
    "targets": {  
      "Instances-Target-1": {  
        "resourceType": "aws:ec2:instance",  
        "resourceArns": [  
          "arn:aws:ec2:us-west-2:123456789012:instance/  
i-12a3b4c56d78e9012"  
        ],  
        "selectionMode": "ALL"  
      }  
    },  
    "actions": {  
      "testaction": {  
        "actionId": "aws:ec2:stop-instances",  
        "parameters": {},  
        "targets": {  
          "Instances": "Instances-Target-1"  
        }  
      }  
    }  
  },  
}
```

```
    "stopConditions": [
      {
        "source": "none"
      }
    ],
    "creationTime": 1616017191.124,
    "lastUpdateTime": 1616017331.51,
    "roleArn": "arn:aws:iam::123456789012:role/FISRole",
    "tags": {
      "key": "value"
    }
  }
}
```

For more information, see [Experiment templates](#) in the *AWS Fault Injection Simulator User Guide*.

- For API details, see [GetExperimentTemplate](#) in *AWS CLI Command Reference*.

get-experiment

The following code example shows how to use `get-experiment`.

AWS CLI

To get experiment details

The following `get-experiment` example gets the details of the specified experiment.

```
aws fis get-experiment \
  --id ABC12DeFGhI3jKLMNOP
```

Output:

```
{
  "experiment": {
    "id": "ABC12DeFGhI3jKLMNOP",
    "experimentTemplateId": "ABCDE1fgHIJkLmNop",
    "roleArn": "arn:aws:iam::123456789012:role/myRole",
    "state": {
      "status": "completed",
      "reason": "Experiment completed."
    }
  },
}
```



```

    "targets": {
      "Instances-Target-1": {
        "resourceType": "aws:ec2:instance",
        "resourceArns": [
          "arn:aws:ec2:us-west-2:123456789012:instance/
i-12a3b4c56d78e9012"
        ],
        "selectionMode": "ALL"
      }
    },
    "actions": {
      "reboot": {
        "actionId": "aws:ec2:reboot-instances",
        "parameters": {},
        "targets": {
          "Instances": "Instances-Target-1"
        },
        "state": {
          "status": "completed",
          "reason": "Action was completed."
        }
      }
    },
    "stopConditions": [
      {
        "source": "none"
      }
    ],
    "creationTime": 1616432509.662,
    "startTime": 1616432509.962,
    "endTime": 1616432522.307,
    "tags": {}
  }
}

```

For more information, see [Experiments for AWS FIS](#) in the *AWS Fault Injection Simulator User Guide*.

- For API details, see [GetExperiment](#) in *AWS CLI Command Reference*.

list-actions

The following code example shows how to use `list-actions`.

AWS CLI

To list actions

The following `list-actions` example lists the available actions.

```
aws fis list-actions
```

Output:

```
{
  "actions": [
    {
      "id": "aws:ec2:reboot-instances",
      "description": "Reboot the specified EC2 instances.",
      "targets": {
        "Instances": {
          "resourceType": "aws:ec2:instance"
        }
      },
      "tags": {}
    },
    {
      "id": "aws:ec2:stop-instances",
      "description": "Stop the specified EC2 instances.",
      "targets": {
        "Instances": {
          "resourceType": "aws:ec2:instance"
        }
      },
      "tags": {}
    },
    {
      "id": "aws:ec2:terminate-instances",
      "description": "Terminate the specified EC2 instances.",
      "targets": {
        "Instances": {
          "resourceType": "aws:ec2:instance"
        }
      },
      "tags": {}
    },
    {
```

```
    "id": "aws:ecs:drain-container-instances",
    "description": "Drain percentage of underlying EC2 instances on an ECS
cluster.",
    "targets": {
      "Clusters": {
        "resourceType": "aws:ecs:cluster"
      }
    },
    "tags": {}
  },
  {
    "id": "aws:eks:terminate-nodegroup-instances",
    "description": "Terminates a percentage of the underlying EC2 instances
in an EKS cluster.",
    "targets": {
      "Nodegroups": {
        "resourceType": "aws:eks:nodegroup"
      }
    },
    "tags": {}
  },
  {
    "id": "aws:fis:inject-api-internal-error",
    "description": "Cause an AWS service to return internal error responses
for specific callers and operations.",
    "targets": {
      "Roles": {
        "resourceType": "aws:iam:role"
      }
    },
    "tags": {}
  },
  {
    "id": "aws:fis:inject-api-throttle-error",
    "description": "Cause an AWS service to return throttled responses for
specific callers and operations.",
    "targets": {
      "Roles": {
        "resourceType": "aws:iam:role"
      }
    },
    "tags": {}
  },
  {
```

```

    "id": "aws:fis:inject-api-unavailable-error",
    "description": "Cause an AWS service to return unavailable error
responses for specific callers and operations.",
    "targets": {
      "Roles": {
        "resourceType": "aws:iam:role"
      }
    },
    "tags": {}
  },
  {
    "id": "aws:fis:wait",
    "description": "Wait for the specified duration. Stop condition
monitoring will continue during this time.",
    "tags": {}
  },
  {
    "id": "aws:rds:failover-db-cluster",
    "description": "Failover a DB Cluster to one of the replicas.",
    "targets": {
      "Clusters": {
        "resourceType": "aws:rds:cluster"
      }
    },
    "tags": {}
  },
  {
    "id": "aws:rds:reboot-db-instances",
    "description": "Reboot the specified DB instances.",
    "targets": {
      "DBInstances": {
        "resourceType": "aws:rds:db"
      }
    },
    "tags": {}
  },
  {
    "id": "aws:ssm:send-command",
    "description": "Run the specified SSM document.",
    "targets": {
      "Instances": {
        "resourceType": "aws:ec2:instance"
      }
    },
  },

```

```
        "tags": {}
      }
    ]
  }
```

For more information, see [Actions](#) in the *AWS Fault Injection Simulator User Guide*.

- For API details, see [ListActions](#) in *AWS CLI Command Reference*.

list-experiment-templates

The following code example shows how to use `list-experiment-templates`.

AWS CLI

To list experiment templates

The following `list-experiment-templates` example lists the experiment templates in your AWS account.

```
aws fis list-experiment-templates
```

Output:

```
{
  "experimentTemplates": [
    {
      "id": "ABCDE1fgHIJkLmNop",
      "description": "myExperimentTemplate",
      "creationTime": 1616017191.124,
      "lastUpdateTime": 1616017191.124,
      "tags": {
        "key": "value"
      }
    }
  ]
}
```

For more information, see [Experiment templates](#) in the *AWS Fault Injection Simulator User Guide*.

- For API details, see [ListExperimentTemplates](#) in *AWS CLI Command Reference*.

list-experiments

The following code example shows how to use `list-experiments`.

AWS CLI

To list experiments

The following `list-experiments` example lists the experiments in your AWS account.

```
aws fis list-experiments
```

Output:

```
{
  "experiments": [
    {
      "id": "ABCdeF1GHiJkLM23NO",
      "experimentTemplateId": "ABCDE1fgHIJkLmNop",
      "state": {
        "status": "running",
        "reason": "Experiment is running."
      },
      "creationTime": 1616017341.197,
      "tags": {
        "key": "value"
      }
    }
  ]
}
```

For more information, see [Experiments](#) in the *AWS Fault Injection Simulator User Guide*.

- For API details, see [ListExperiments](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags for a resource

The following `list-tags-for-resource` example lists the tags for the specified resource.

```
aws fis list-tags-for-resource \  
  --resource-arn arn:aws:fis:us-west-2:123456789012:experiment/ABC12DeFGhI3jKLMNOP
```

Output:

```
{  
  "tags": {  
    "key1": "value1",  
    "key2": "value2"  
  }  
}
```

For more information, see [Tag your AWS FIS resources](#) in the *AWS Fault Injection Simulator User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

start-experiment

The following code example shows how to use `start-experiment`.

AWS CLI

To start an experiment

The following `start-experiment` example starts the specified experiment.

```
aws fis start-experiment \  
  --experiment-template-id ABCDE1fgHIJkLmNop
```

Output:

```
{  
  "experiment": {  
    "id": "ABC12DeFGhI3jKLMNOP",  
    "experimentTemplateId": "ABCDE1fgHIJkLmNop",  
    "roleArn": "arn:aws:iam::123456789012:role/myRole",  
    "state": {  
      "status": "initiating",  
      "reason": "Experiment is initiating."  
    }  
  }  
}
```

```

    },
    "targets": {
      "Instances-Target-1": {
        "resourceType": "aws:ec2:instance",
        "resourceArns": [
          "arn:aws:ec2:us-west-2:123456789012:instance/
i-12a3b4c56d78e9012"
        ],
        "selectionMode": "ALL"
      }
    },
    "actions": {
      "reboot": {
        "actionId": "aws:ec2:reboot-instances",
        "parameters": {},
        "targets": {
          "Instances": "Instances-Target-1"
        },
        "state": {
          "status": "pending",
          "reason": "Initial state"
        }
      }
    },
    "stopConditions": [
      {
        "source": "none"
      }
    ],
    "creationTime": 1616432464.025,
    "startTime": 1616432464.374,
    "tags": {}
  }
}

```

For more information, see [Experiments for AWS FIS](#) in the *AWS Fault Injection Simulator User Guide*.

- For API details, see [StartExperiment](#) in *AWS CLI Command Reference*.

stop-experiment

The following code example shows how to use stop-experiment.

AWS CLI

To stop an experiment

The following stop-experiment example stops the specified experiment from running.

```
aws fis stop-experiment \  
  --id ABC12DeFGhI3jKLMNOP
```

Output:

```
{  
  "experiment": {  
    "id": "ABC12DeFGhI3jKLMNOP",  
    "experimentTemplateId": "ABCDE1fgHIJkLmNop",  
    "roleArn": "arn:aws:iam::123456789012:role/myRole",  
    "state": {  
      "status": "stopping",  
      "reason": "Stopping Experiment."  
    },  
    "targets": {  
      "Instances-Target-1": {  
        "resourceType": "aws:ec2:instance",  
        "resourceArns": [  
          "arn:aws:ec2:us-west-2:123456789012:instance/  
i-12a3b4c56d78e9012"  
        ],  
        "selectionMode": "ALL"  
      }  
    },  
    "actions": {  
      "reboot": {  
        "actionId": "aws:ec2:reboot-instances",  
        "parameters": {},  
        "targets": {  
          "Instances": "Instances-Target-1"  
        },  
        "startAfter": [  
          "wait"  
        ],  
        "state": {  
          "status": "pending",  
          "reason": "Initial state."  
        }  
      }  
    }  
  }  
}
```

```

    }
  },
  "wait": {
    "actionId": "aws:fis:wait",
    "parameters": {
      "duration": "PT5M"
    },
    "state": {
      "status": "running",
      "reason": ""
    }
  }
},
"stopConditions": [
  {
    "source": "none"
  }
],
"creationTime": 1616432680.927,
"startTime": 1616432681.177,
"tags": {}
}
}

```

For more information, see [Experiments for AWS FIS](#) in the *AWS Fault Injection Simulator User Guide*.

- For API details, see [StopExperiment](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To tag a resource

The following tag-resource example tags the specified resource.

```

aws fis tag-resource \
  --resource-arn arn:aws:fis:us-west-2:123456789012:experiment/ABC12DeFGhI3jKLMNOP
\
  --tags key1=value1,key2=value2

```

This command produces no output.

For more information, see [Tag your AWS FIS resources](#) in the *AWS Fault Injection Simulator User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To untag a resource

The following `untag-resource` example removes the tags from the specified resource.

```
aws fis untag-resource \  
  --resource-arn arn:aws:fis:us-west-2:123456789012:experiment/ABC12DeFGhI3jKLMNOP
```

This command produces no output.

For more information, see [Tag your AWS FIS resources](#) in the *AWS Fault Injection Simulator User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-experiment-template

The following code example shows how to use `update-experiment-template`.

AWS CLI

To update an experiment template

The following `update-experiment-template` example updates the description of the specified experiment template.

```
aws fis update-experiment-template \  
  --id ABCDE1fgHIJkLmNop \  
  --description "Updated description"
```

```
---description myExperimentTemplate
```

Output:

```
{
  "experimentTemplate": {
    "id": "ABCDE1fgHIJKLmNop",
    "description": "myExperimentTemplate",
    "targets": {
      "Instances-Target-1": {
        "resourceType": "aws:ec2:instance",
        "resourceArns": [
          "arn:aws:ec2:us-west-2:123456789012:instance/
i-12a3b4c56d78e9012"
        ],
        "selectionMode": "ALL"
      }
    },
    "actions": {
      "testaction": {
        "actionId": "aws:ec2:stop-instances",
        "parameters": {},
        "targets": {
          "Instances": "Instances-Target-1"
        }
      }
    },
    "stopConditions": [
      {
        "source": "none"
      }
    ],
    "creationTime": 1616017191.124,
    "lastUpdateTime": 1616017859.607,
    "roleArn": "arn:aws:iam::123456789012:role/FISRole",
    "tags": {
      "key": "value"
    }
  }
}
```

For more information, see [Update an experiment template](#) in the *AWS Fault Injection Simulator User Guide*.

- For API details, see [UpdateExperimentTemplate](#) in *AWS CLI Command Reference*.

Amazon GameLift examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon GameLift.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-build

The following code example shows how to use create-build.

AWS CLI

Example1: To create a game build from files in an S3 bucket

The following create-build example creates a custom game build resource. It uses zipped files that are stored in an S3 location in an AWS account that you control. This example assumes that you've already created an IAM role that gives Amazon GameLift permission to access the S3 location. Since the request does not specify an operating system, the new build resource defaults to WINDOWS_2012.

```
aws gamelift create-build \  
  --storage-location file://storage-loc.json \  
  --name MegaFrogRaceServer.NA \  
  --build-version 12345.678
```

Contents of storage-loc.json:

```
{
  "Bucket": "MegaFrogRaceServer_NA_build_files"
  "Key": "MegaFrogRaceServer_build_123.zip"
  "RoleArn": "arn:aws:iam::123456789012:role/gamelift"
}
```

Output:

```
{
  "Build": {
    "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "CreationTime": 1496708916.18,
    "Name": "MegaFrogRaceServer.NA",
    "OperatingSystem": "WINDOWS_2012",
    "SizeOnDisk": 479303,
    "Status": "INITIALIZED",
    "Version": "12345.678"
  },
  "StorageLocation": {
    "Bucket": "MegaFrogRaceServer_NA_build_files",
    "Key": "MegaFrogRaceServer_build_123.zip"
  }
}
```

Example2: To create a game build resource for manually uploading files to GameLift

The following `create-build` example creates a new build resource. It also gets a storage location and temporary credentials that allow you to manually upload your game build to the GameLift location in Amazon S3. Once you've successfully uploaded your build, the GameLift service validates the build and updates the new build's status.

```
aws gamelift create-build \
  --name MegaFrogRaceServer.NA \
  --build-version 12345.678 \
  --operating-system AMAZON_LINUX
```

Output:

```
{
  "Build": {
    "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "CreationTime": 1496708916.18,
    "Name": "MegaFrogRaceServer.NA",
    "OperatingSystem": "AMAZON_LINUX",
    "SizeOnDisk": 0,
    "Status": "INITIALIZED",
    "Version": "12345.678"
  },
  "StorageLocation": {
    "Bucket": "gamelift-builds-us-west-2",
    "Key": "123456789012/build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
  },
  "UploadCredentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY",
    "SessionToken": "AgoGb3JpZ2luENZ...EXAMPLETOKEN=="
  }
}
```

For more information, see [Upload a Custom Server Build to GameLift](#) in the *Amazon GameLift Developer Guide*.

- For API details, see [CreateBuild](#) in *AWS CLI Command Reference*.

create-fleet

The following code example shows how to use `create-fleet`.

AWS CLI

Example 1: To create a basic Linux fleet

The following `create-fleet` example creates a minimally configured fleet of on-demand Linux instances to host a custom server build. You can complete the configuration by using `update-fleet`.

```
aws gamelift create-fleet \
  --name MegaFrogRaceServer.NA.v2 \
```

```

--description 'Hosts for v2 North America' \
--build-id build-1111aaaa-22bb-33cc-44dd-5555eeee66ff \
--certificate-configuration 'CertificateType=GENERATED' \
--ec2-instance-type c4.large \
--fleet-type ON_DEMAND \
--runtime-configuration 'ServerProcesses=[{LaunchPath=/local/game/release-na/
MegaFrogRace_Server.exe,ConcurrentExecutions=1}]'

```

Output:

```

{
  "FleetAttributes": {
    "BuildId": "build-1111aaaa-22bb-33cc-44dd-5555eeee66ff",
    "CertificateConfiguration": {
      "CertificateType": "GENERATED"
    },
    "CreationTime": 1496365885.44,
    "Description": "Hosts for v2 North America",
    "FleetArn": "arn:aws:gamelift:us-west-2:444455556666:fleet/
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
    "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
    "FleetType": "ON_DEMAND",
    "InstanceType": "c4.large",
    "MetricGroups": ["default"],
    "Name": "MegaFrogRace.NA.v2",
    "NewGameSessionProtectionPolicy": "NoProtection",
    "OperatingSystem": "AMAZON_LINUX",
    "ServerLaunchPath": "/local/game/release-na/MegaFrogRace_Server.exe",
    "Status": "NEW"
  }
}

```

Example 2: To create a basic Windows fleet

The following `create-fleet` example creates a minimally configured fleet of spot Windows instances to host a custom server build. You can complete the configuration by using `update-fleet`.

```

aws gamelift create-fleet \
  --name MegaFrogRace.NA.v2 \
  --description 'Hosts for v2 North America' \
  --build-id build-2222aaaa-33bb-44cc-55dd-6666eeee77ff \
  --certificate-configuration 'CertificateType=GENERATED' \

```



```
--ec2-instance-type c4.large \
--fleet-type SPOT \
--runtime-configuration 'ServerProcesses=[{LaunchPath=C:\game
\Bin64.Release.Dedicated\MegaFrogRace_Server.exe,ConcurrentExecutions=1}]'
```

Output:

```
{
  "FleetAttributes": {
    "BuildId": "build-2222aaaa-33bb-44cc-55dd-6666eeee77ff",
    "CertificateConfiguration": {
      "CertificateType": "GENERATED"
    },
    "CreationTime": 1496365885.44,
    "Description": "Hosts for v2 North America",
    "FleetArn": "arn:aws:gamelift:us-west-2:444455556666:fleet/
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
    "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
    "FleetType": "SPOT",
    "InstanceType": "c4.large",
    "MetricGroups": ["default"],
    "Name": "MegaFrogRace.NA.v2",
    "NewGameSessionProtectionPolicy": "NoProtection",
    "OperatingSystem": "WINDOWS_2012",
    "ServerLaunchPath": "C:\game\Bin64.Release.Dedicated
\MegaFrogRace_Server.exe",
    "Status": "NEW"
  }
}
```

Example 3: To create a fully configured fleet

The following `create-fleet` example creates a fleet of Spot Windows instances for a custom server build, with most commonly used configuration settings provided.

```
aws gamelift create-fleet \
--name MegaFrogRace.NA.v2 \
--description 'Hosts for v2 North America' \
--build-id build-2222aaaa-33bb-44cc-55dd-6666eeee77ff \
--certificate-configuration 'CertificateType=GENERATED' \
--ec2-instance-type c4.large \
--ec2-inbound-permissions
'FromPort=33435,ToPort=33435,IpRange=10.24.34.0/23,Protocol=UDP' \
```

```
--fleet-type SPOT \  
--new-game-session-protection-policy FullProtection \  
--runtime-configuration file://runtime-config.json \  
--metric-groups default \  
--instance-role-arn 'arn:aws:iam::444455556666:role/GameLiftS3Access'
```

Contents of runtime-config.json:

```
GameSessionActivationTimeoutSeconds=300,  
MaxConcurrentGameSessionActivations=2,  
ServerProcesses=[  
  {LaunchPath=C:\game\Bin64.Release.Dedicated\MegaFrogRace_Server.exe,Parameters=-  
debug,ConcurrentExecutions=1},  
  {LaunchPath=C:\game\Bin64.Release.Dedicated  
\MegaFrogRace_Server.exe,ConcurrentExecutions=1}]
```

Output:

```
{  
  "FleetAttributes": {  
    "InstanceRoleArn": "arn:aws:iam::444455556666:role/GameLiftS3Access",  
    "Status": "NEW",  
    "InstanceType": "c4.large",  
    "FleetArn": "arn:aws:gamelift:us-west-2:444455556666:fleet/  
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",  
    "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",  
    "Description": "Hosts for v2 North America",  
    "FleetType": "SPOT",  
    "OperatingSystem": "WINDOWS_2012",  
    "Name": "MegaFrogRace.NA.v2",  
    "CreationTime": 1569309011.11,  
    "MetricGroups": [  
      "default"  
    ],  
    "BuildId": "build-2222aaaa-33bb-44cc-55dd-6666eeee77ff",  
    "ServerLaunchParameters": "abc",  
    "ServerLaunchPath": "C:\\game\\Bin64.Release.Dedicated\  
\MegaFrogRace_Server.exe",  
    "NewGameSessionProtectionPolicy": "FullProtection",  
    "CertificateConfiguration": {  
      "CertificateType": "GENERATED"  
    }  
  }  
}
```

```
}

```

Example 4: To create a Realtime Servers fleet

The following `create-fleet` example creates a fleet of Spot instances with a Realtime configuration script that has been uploaded to Amazon GameLift. All Realtime servers are deployed onto Linux machines. For the purposes of this example, assume that the uploaded Realtime script includes multiple script files, with the `Init()` function located in the script file called `MainScript.js`. As shown, this file is identified as the launch script in the runtime configuration.

```
aws gamelift create-fleet \
  --name MegaFrogRace.NA.realtime \
  --description 'Mega Frog Race Realtime fleet' \
  --script-id script-1111aaaa-22bb-33cc-44dd-5555eeee66ff \
  --ec2-instance-type c4.large \
  --fleet-type SPOT \
  --certificate-configuration 'CertificateType=GENERATED' --runtime-configuration
'ServerProcesses=[{LaunchPath=/local/game/MainScript.js,Parameters=+map
Winter444,ConcurrentExecutions=5}]'
```

Output:

```
{
  "FleetAttributes": {
    "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
    "Status": "NEW",
    "CreationTime": 1569310745.212,
    "InstanceType": "c4.large",
    "NewGameSessionProtectionPolicy": "NoProtection",
    "CertificateConfiguration": {
      "CertificateType": "GENERATED"
    },
    "Name": "MegaFrogRace.NA.realtime",
    "ScriptId": "script-1111aaaa-22bb-33cc-44dd-5555eeee66ff",
    "FleetArn": "arn:aws:gamelift:us-west-2:444455556666:fleet/
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
    "FleetType": "SPOT",
    "MetricGroups": [
      "default"
    ],
    "Description": "Mega Frog Race Realtime fleet",
```

```

    "OperatingSystem": "AMAZON_LINUX"
  }
}

```

- For API details, see [CreateFleet](#) in *AWS CLI Command Reference*.

create-game-session-queue

The following code example shows how to use `create-game-session-queue`.

AWS CLI

Example1: To set up an ordered game session queue

The following `create-game-session-queue` example creates a new game session queue with destinations in two regions. It also configures the queue so that game session requests time out after waiting 10 minutes for placement. Since no latency policies are defined, GameLift attempts to place all game sessions with the first destination listed.

```

aws gamelift create-game-session-queue \
  --name MegaFrogRaceServer-NA \
  --destinations file://destinations.json \
  --timeout-in-seconds 600

```

Contents of `destinations.json`:

```

{
  "Destinations": [
    {
      "DestinationArn": "arn:aws:gamelift:us-west-2::fleet/fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    },
    {
      "DestinationArn": "arn:aws:gamelift:us-west-1::fleet/fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
    }
  ]
}

```

Output:

```

{
  "GameSessionQueues": [
    {
      "Name": "MegaFrogRaceServer-NA",

```

```

    "GameSessionQueueArn": "arn:aws:gamelift:us-
west-2:123456789012:gamesessionqueue/MegaFrogRaceServer-NA",
    "TimeoutInSeconds": 600,
    "Destinations": [
      {"DestinationArn": "arn:aws:gamelift:us-west-2::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"},
      {"DestinationArn": "arn:aws:gamelift:us-west-1::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"}
    ]
  }
]
}

```

Example2: To set up a game session queue with player latency policies

The following `create-game-session-queue` example creates a new game session queue with two player latency policies. The first policy sets a 100ms latency cap that is enforced during the first minute of a game session placement attempt. The second policy raises the latency cap to 200ms until the placement request times out at 3 minutes.

```

aws gamelift create-game-session-queue \
  --name MegaFrogRaceServer-NA \
  --destinations file://destinations.json \
  --player-latency-policies file://latency-policies.json \
  --timeout-in-seconds 180

```

Contents of `destinations.json`:

```

{
  "Destinations": [
    { "DestinationArn": "arn:aws:gamelift:us-west-2::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111" },
    { "DestinationArn": "arn:aws:gamelift:us-east-1::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222" }
  ]
}

```

Contents of `latency-policies.json`:

```

{
  "PlayerLatencyPolicies": [

```

```

        {"MaximumIndividualPlayerLatencyMilliseconds": 200},
        {"MaximumIndividualPlayerLatencyMilliseconds": 100, "PolicyDurationSeconds":
60}
    ]
}

```

Output:

```

{
  "GameSessionQueue": {
    "Name": "MegaFrogRaceServer-NA",
    "GameSessionQueueArn": "arn:aws:gamelift:us-
west-2:111122223333:gamesessionqueue/MegaFrogRaceServer-NA",
    "TimeoutInSeconds": 600,
    "PlayerLatencyPolicies": [
      {
        "MaximumIndividualPlayerLatencyMilliseconds": 100,
        "PolicyDurationSeconds": 60
      },
      {
        "MaximumIndividualPlayerLatencyMilliseconds": 200
      }
    ]
    "Destinations": [
      {"DestinationArn": "arn:aws:gamelift:us-west-2::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"},
      {"DestinationArn": "arn:aws:gamelift:us-east-1::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"}
    ],
  }
}

```

For more information, see [Create a Queue](#) in the *Amazon GameLift Developer Guide*.

- For API details, see [CreateGameSessionQueue](#) in *AWS CLI Command Reference*.

delete-build

The following code example shows how to use delete-build.

AWS CLI

To delete a custom game build

The following `delete-build` example removes a build from your Amazon GameLift account. After the build is deleted, you cannot use it to create new fleets. This operation cannot be undone.

```
aws gamelift delete-build \  
  --build-id build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

This command produces no output.

- For API details, see [DeleteBuild](#) in *AWS CLI Command Reference*.

delete-fleet

The following code example shows how to use `delete-fleet`.

AWS CLI

To delete a fleet that is no longer in use

The following `delete-fleet` example removes a fleet that has been scaled down to zero instances. If the fleet capacity is greater than zero, the request fails with an HTTP 400 error.

```
aws gamelift delete-fleet \  
  --fleet-id fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

This command produces no output.

For more information, see [Manage GameLift Fleets](#) in the *Amazon GameLift Developer Guide*.

- For API details, see [DeleteFleet](#) in *AWS CLI Command Reference*.

delete-game-session-queue

The following code example shows how to use `delete-game-session-queue`.

AWS CLI

To delete a game session queue

The following `delete-game-session-queue` example deletes a specified game session queue.

```
aws gamelift delete-game-session-queue \  
  --name MegaFrogRace-NA
```

This command produces no output.

- For API details, see [DeleteGameSessionQueue](#) in *AWS CLI Command Reference*.

describe-build

The following code example shows how to use describe-build.

AWS CLI

To get information on a custom game build

The following describe-build example retrieves properties for a game server build resource.

```
aws gamelift describe-build \  
  --build-id build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{  
  "Build": {  
    "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-  
cdef-EXAMPLE11111",  
    "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
    "CreationTime": 1496708916.18,  
    "Name": "My_Game_Server_Build_One",  
    "OperatingSystem": "AMAZON_LINUX",  
    "SizeOnDisk": 1304924,  
    "Status": "READY",  
    "Version": "12345.678"  
  }  
}
```

For more information, see [Upload a Custom Server Build to GameLift](#) in the *Amazon GameLift Developer Guide*.

- For API details, see [DescribeBuild](#) in *AWS CLI Command Reference*.

describe-ec2-instance-limits

The following code example shows how to use `describe-ec2-instance-limits`.

AWS CLI

To retrieve service limits for an EC2 instance type

The following `describe-ec2-instance-limits` example displays the maximum allowed instances and current instances in use for the specified EC2 instance type in the current Region. The result indicates that only five of the allowed twenty instances are being used.

```
aws gamelift describe-ec2-instance-limits \  
  --ec2-instance-type m5.large
```

Output:

```
{  
  "EC2InstanceLimits": [  
    {  
      "EC2InstanceType": "m5.large",  
      "CurrentInstances": 5,  
      "InstanceLimit": 20  
    }  
  ]  
}
```

For more information, see [Choose Computing Resources](#) in the *Amazon GameLift Developer Guide*.

- For API details, see [DescribeEc2InstanceLimits](#) in *AWS CLI Command Reference*.

describe-fleet-attributes

The following code example shows how to use `describe-fleet-attributes`.

AWS CLI

Example1: To view attributes for a list of fleets

The following `describe-fleet-attributes` example retrieves fleet attributes for two specified fleets. As shown, the requested fleets are deployed with the same build, one for On-Demand instances and one for Spot instances, with some minor configuration differences.

```
aws gamelift describe-fleet-attributes \  
  --fleet-ids arn:aws:gamelift:us-west-2::fleet/fleet-a1b2c3d4-5678-90ab-cdef-  
EXAMPLE11111 fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222
```

Output:

```
{  
  "FleetAttributes": [  
    {  
      "FleetId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
      "FleetArn": "arn:aws:gamelift:us-west-2::fleet/fleet-a1b2c3d4-5678-90ab-  
cdef-EXAMPLE11111",  
      "FleetType": "ON_DEMAND",  
      "InstanceType": "c4.large",  
      "Description": "On-demand hosts for v2 North America",  
      "Name": "MegaFrogRaceServer.NA.v2-od",  
      "CreationTime": 1568836191.995,  
      "Status": "ACTIVE",  
      "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",  
      "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-  
cdef-EXAMPLE33333",  
      "ServerLaunchPath": "C:\\\\game\\\\MegaFrogRace_Server.exe",  
      "ServerLaunchParameters": "+gamelift_start_server",  
      "NewGameSessionProtectionPolicy": "NoProtection",  
      "OperatingSystem": "WINDOWS_2012",  
      "MetricGroups": [  
        "default"  
      ],  
      "CertificateConfiguration": {  
        "CertificateType": "DISABLED"  
      }  
    },  
    {  
      "FleetId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",  
      "FleetArn": "arn:aws:gamelift:us-west-2::fleet/fleet-a1b2c3d4-5678-90ab-  
cdef-EXAMPLE22222",  
      "FleetType": "SPOT",  
      "InstanceType": "c4.large",  
      "Description": "On-demand hosts for v2 North America",
```

```

    "Name": "MegaFrogRaceServer.NA.v2-spot",
    "CreationTime": 1568838275.379,
    "Status": "ACTIVATING",
    "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-
cdef-EXAMPLE33333",
    "ServerLaunchPath": "C:\\\\game\\\\MegaFrogRace_Server.exe",
    "NewGameSessionProtectionPolicy": "NoProtection",
    "OperatingSystem": "WINDOWS_2012",
    "MetricGroups": [
      "default"
    ],
    "CertificateConfiguration": {
      "CertificateType": "GENERATED"
    }
  }
]
}

```

Example2: To request attributes for all fleets

The following `describe-fleet-attributes` returns fleet attributes for all fleets with any status. This example illustrates the use of pagination parameters to return one fleet at a time.

```

aws gamelift describe-fleet-attributes \
  --limit 1

```

Output:

```

{
  "FleetAttributes": [
    {
      "FleetId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
      "FleetArn": "arn:aws:gamelift:us-west-2::fleet/fleet-a1b2c3d4-5678-90ab-
cdef-EXAMPLE22222",
      "FleetType": "SPOT",
      "InstanceType": "c4.large",
      "Description": "On-demand hosts for v2 North America",
      "Name": "MegaFrogRaceServer.NA.v2-spot",
      "CreationTime": 1568838275.379,
      "Status": "ACTIVATING",
      "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",

```

```

    "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-
cdef-EXAMPLE33333",
    "ServerLaunchPath": "C:\\\\game\\\\MegaFrogRace_Server.exe",
    "NewGameSessionProtectionPolicy": "NoProtection",
    "OperatingSystem": "WINDOWS_2012",
    "MetricGroups": [
        "default"
    ],
    "CertificateConfiguration": {
        "CertificateType": "GENERATED"
    }
}
],
"NextToken":
"eyJhd3NBdW50SWQiOnsicyI6IjMwMjc3NjAxNjM5OCJ9LCJidWlsZE1kIj7InMi0iJidWlsZC01NWYxZTZmMS1"
}

```

The output includes a `NextToken` value that you can use when you call the command a second time. Pass the value to the `--next-token` parameter to specify where to pick up the output. The following command returns the second result in the output.

```

aws gamelift describe-fleet-attributes \
  --limit 1 \
  --next-token
eyJhd3NBdW50SWQiOnsicyI6IjMwMjc3NjAxNjM5OCJ9LCJidWlsZE1kIj7InMi0iJidWlsZC01NWYxZTZmMS1

```

Repeat until the response doesn't include a `NextToken` value.

For more information, see [Setting Up GameLift Fleets](#) in the *Amazon GameLift Developer Guide*.

- For API details, see [DescribeFleetAttributes](#) in *AWS CLI Command Reference*.

describe-fleet-capacity

The following code example shows how to use `describe-fleet-capacity`.

AWS CLI

To view capacity status for a list of fleets

The following `describe-fleet-capacity` example retrieves current capacity for two specified fleets.

```
aws gamelift describe-fleet-capacity \
  --fleet-ids arn:aws:gamelift:us-west-2::fleet/fleet-a1b2c3d4-5678-90ab-cdef-
EXAMPLE111111 fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE222222
```

Output:

```
{
  "FleetCapacity": [
    {
      "FleetId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE111111",
      "InstanceType": "c5.large",
      "InstanceCounts": {
        "DESIRED": 10,
        "MINIMUM": 1,
        "MAXIMUM": 20,
        "PENDING": 0,
        "ACTIVE": 10,
        "IDLE": 3,
        "TERMINATING": 0
      }
    },
    {
      "FleetId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE222222",
      "InstanceType": "c5.large",
      "InstanceCounts": {
        "DESIRED": 13,
        "MINIMUM": 1,
        "MAXIMUM": 20,
        "PENDING": 0,
        "ACTIVE": 15,
        "IDLE": 2,
        "TERMINATING": 2
      }
    }
  ]
}
```

For more information, see [GameLift Metrics for Fleets](#) in the *Amazon GameLift Developer Guide*.

- For API details, see [DescribeFleetCapacity](#) in *AWS CLI Command Reference*.

describe-fleet-events

The following code example shows how to use describe-fleet-events.

AWS CLI

To request events for a specified time span

The following describe-fleet-events example displays details of all fleet-related events that occurred during the specified time span.

```
aws gamelift describe-fleet-events \  
  --fleet-id arn:aws:gamelift:us-west-2::fleet/fleet-a1b2c3d4-5678-90ab-cdef-  
EXAMPLE11111 \  
  --start-time 1579647600 \  
  --end-time 1579649400 \  
  --limit 5
```

Output:

```
{  
  "Events": [  
    {  
      "EventId": "a37b6892-5d07-4d3b-8b47-80244ecf66b9",  
      "ResourceId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
      "EventCode": "FLEET_STATE_ACTIVE",  
      "Message": "Fleet fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 changed  
state to ACTIVE",  
      "EventTime": 1579649342.191  
    },  
    {  
      "EventId": "67da4ec9-92a3-4d95-886a-5d6772c24063",  
      "ResourceId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
      "EventCode": "FLEET_STATE_ACTIVATING",  
      "Message": "Fleet fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 changed  
state to ACTIVATING",  
      "EventTime": 1579649321.427  
    },  
    {  
      "EventId": "23813a46-a9e6-4a53-8847-f12e6a8381ac",  
      "ResourceId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
      "EventCode": "FLEET_STATE_BUILDING",
```

```

    "Message": "Fleet fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 changed
state to BUILDING",
    "EventTime": 1579649321.243
  },
  {
    "EventId": "3bf217d0-1d44-42f9-9202-433ed475d2e8",
    "ResourceId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "EventCode": "FLEET_STATE_VALIDATING",
    "Message": "Fleet fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 changed
state to VALIDATING",
    "EventTime": 1579649197.449
  },
  {
    "EventId": "2ecd0130-5986-44eb-99a7-62df27741084",
    "ResourceId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "EventCode": "FLEET_VALIDATION_LAUNCH_PATH_NOT_FOUND",
    "Message": "Failed to find a valid path",
    "EventTime": 1569319075.839,
    "PreSignedLogUrl": "https://gamelift-event-logs-prod-
us-west-2.s3.us-west-2.amazonaws.com/logs/fleet-83422059-8329-42a2-
a4d6-c4444386a6f8/events/2ecd0130-5986-44eb-99a7-62df27741084/
FLEET_VALIDATION_LAUNCH_PATH_NOT_FOUND.txt?X-Amz-Security-
Token=IQoJb3JpZ2luX2VjEB8aCXVzLXdlc3QtMiJHMEUCIHV5K%2FLPx8h310D
%2FAvx0%2FZxsDy5XA3cJ0wPdu3T0eBa%2FAiEA1yovokcZYy%2FV4CWW6126aFyiSHO
%2Bxz%2FBMAhEHYHMqNcQkQMImp%2F%2F%2F%2F%2F%2F%2F%2F%2F%2F
%2FARAAGgw3NDEwNjE0TixNzEiDI8rsZtzLzlwEDQhXSrlAtl5Ae
%2Fgo6FCIzqxPbXfB0nSvFYqeDlriZarEpKqKrUt8mXQv9iqHResqCph9AKo491wgSYTT2QoSxnrD7%2FUgv
%2BZm2pVuczvuKtUA0fCx6s0GxpjIAzdIE%2F5P%2F7B9M%2BVZ
%2F9KF82hbJi0HTE6Y7BjKsEgFCvk4UXILhfjtan9iQ18%2F21ZTurAcJbm7Y5tuLF9SWSK3%2BEa7VX0cCK4D401sMj
%2FIaXoHkNvg0RVTa0hIqdvpDQlsSBNdqTXbjHTu6fETE9Y9Ky%2BiJK5KiUG
%2F59GjCpDcvS1FqKeLUEmKT7wysGmvjMc2n%2Fr
%2F9VxQfte7w9srXwLLAQuwhiXAAyI5ICMz5JvzjzQwTqD4CHTVKUUDwL
%2BRZzbuuqkJ0bZml02CkRGp%2B74RTAzLbWptVqZTIIfzctiCTmWxb
%2FmKyELRYsVLrwNJ%2B7G7%2BCrN0RC%2FjlgfLYIZyeAqjPgAu5HjgX
%2BM7jCo9M7wBTInAXK0FQuf9dvA84SuwX0JFp17LYGjrHMKv0qC3GfbTMrZ6kzeNV9awKCpXB2Gnx9z2KvI1JdqirWV
%2F9C6%2B4jIZPME3jXmZcEHqqw5uvAVF7aeIavtUZU8pXPDIWT0YE4p3Kriy2AA7ziCRKtVfjV839InyLk8LUjsioWk
%2BYUq8%2FDTL1Lxqj1S%2Fi04TI0Wo7ilAo%2FKKWWF4guuNDexj8E00ynSp1yImB
%2BZf2Fua3044W4eEXAMPLE33333&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Date=20170621T231808Z&X-Amz-SignedHeaders=host&X-Amz-Expires=900&X-Amz-
Credential=AKIAIOSFODNN7EXAMPLE%2F20170621%2Fus-west-2%2Fs3%2Faws4_request&X-Amz-
Signature=wJalrXUtnFEMI/K7MDENG/bPxrFfiCYEXAMPLEKEY"
  }
],

```

```
"NextToken":  
"eyJhd3NBWY2NvdW50SWQiOnsicyI6IjMwMjc3NjAxNjM5OCJ9LCJidWlsZEIkJp7InMiOiJidWlsZC01NWYxZTZmMS"  
}
```

For more information, see [Debug GameLift Fleet Issues](#) in the *Amazon GameLift Developer Guide*.

- For API details, see [DescribeFleetEvents](#) in *AWS CLI Command Reference*.

describe-fleet-port-settings

The following code example shows how to use `describe-fleet-port-settings`.

AWS CLI

To view inbound connection permissions for a fleet

The following `describe-fleet-port-settings` example retrieves connection settings for a specified fleet.

```
aws gamelift describe-fleet-port-settings \  
  --fleet-id arn:aws:gamelift:us-west-2::fleet/fleet-a1b2c3d4-5678-90ab-cdef-  
  EXAMPLE11111
```

Output:

```
{  
  "InboundPermissions": [  
    {  
      "FromPort": 33400,  
      "ToPort": 33500,  
      "IpRange": "0.0.0.0/0",  
      "Protocol": "UDP"  
    },  
    {  
      "FromPort": 1900,  
      "ToPort": 2000,  
      "IpRange": "0.0.0.0/0",  
      "Protocol": "TCP"  
    }  
  ]  
}
```


For more information, see [Setting Up GameLift Fleets](#) in the *Amazon GameLift Developer Guide*.

- For API details, see [DescribeFleetPortSettings](#) in *AWS CLI Command Reference*.

describe-fleet-utilization

The following code example shows how to use describe-fleet-utilization.

AWS CLI

Example1: To view usage data for a list of fleets

The following describe-fleet-utilization example retrieves current usage information for one specified fleet.

```
aws gamelift describe-fleet-utilization \  
  --fleet-ids arn:aws:gamelift:us-west-2::fleet/fleet-a1b2c3d4-5678-90ab-cdef-  
  EXAMPLE11111
```

Output:

```
{  
  "FleetUtilization": [  
    {  
      "FleetId": "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
      "ActiveServerProcessCount": 100,  
      "ActiveGameSessionCount": 62,  
      "CurrentPlayerSessionCount": 329,  
      "MaximumPlayerSessionCount": 1000  
    }  
  ]  
}
```

Example2: To request usage data for all fleets

The following describe-fleet-utilization returns fleet usage data for all fleets with any status. This example uses pagination parameters to return data for two fleets at a time.

```
aws gamelift describe-fleet-utilization \  
  --limit 2
```

Output:

```
{
  "FleetUtilization": [
    {
      "FleetId": "fleet-1111aaaa-22bb-33cc-44dd-5555eeee66ff",
      "ActiveServerProcessCount": 100,
      "ActiveGameSessionCount": 13,
      "CurrentPlayerSessionCount": 98,
      "MaximumPlayerSessionCount": 1000
    },
    {
      "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
      "ActiveServerProcessCount": 100,
      "ActiveGameSessionCount": 62,
      "CurrentPlayerSessionCount": 329,
      "MaximumPlayerSessionCount": 1000
    }
  ],
  "NextToken":
  "eyJhd3NBWY2NvdW50SWQiOnsicyI6IjMwMjc3NjAxNjM5OCJ9LCJidWlsZEIkJp7InMi0iJidWlsZC01NWYxZTZmMS1"
}
```

Call the command a second time, passing the `NextToken` value as the argument to the `--next-token` parameter to see the next two results.

```
aws gamelift describe-fleet-utilization \
  --limit 2 \
  --next-token
eyJhd3NBWY2NvdW50SWQiOnsicyI6IjMwMjc3NjAxNjM5OCJ9LCJidWlsZEIkJp7InMi0iJidWlsZC01NWYxZTZmMS1
```

Repeat until the response no longer includes a `NextToken` value in the output.

For more information, see [GameLift Metrics for Fleets](#) in the *Amazon GameLift Developer Guide*.

- For API details, see [DescribeFleetUtilization](#) in *AWS CLI Command Reference*.

describe-game-session-queues

The following code example shows how to use `describe-game-session-queues`.

AWS CLI

To view game session queues

The following `describe-game-session-queues` example retrieves properties for two specified queues.

```
aws gamelift describe-game-session-queues \
  --names MegaFrogRace-NA MegaFrogRace-EU
```

Output:

```
{
  "GameSessionQueues": [{
    "Destinations": [{
      "DestinationArn": "arn:aws:gamelift:us-west-2::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    },
    {
      "DestinationArn": "arn:aws:gamelift:us-west-2::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
    }
  ],
  "Name": "MegaFrogRace-NA",
  "TimeoutInSeconds": 600,
  "GameSessionQueueArn": "arn:aws:gamelift:us-west-2::gamesessionqueue/
MegaFrogRace-NA",
  "PlayerLatencyPolicies": [{
    "MaximumIndividualPlayerLatencyMilliseconds": 200
  },
  {
    "MaximumIndividualPlayerLatencyMilliseconds": 100,
    "PolicyDurationSeconds": 60
  }
  ],
  "FilterConfiguration": {
    "AllowedLocations": ["us-west-2", "ap-south-1", "us-east-1"]
  },
  "PriorityConfiguration": {
    "PriorityOrder": ["LOCATION", "FLEET_TYPE", "DESTINATION"],
    "LocationOrder": ["us-west-2", "ap-south-1", "us-east-1"]
  }
  },
  {
    "Destinations": [{
      "DestinationArn": "arn:aws:gamelift:eu-west-3::fleet/fleet-
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
```

```

    ]],
    "Name": "MegaFrogRace-EU",
    "TimeoutInSeconds": 600,
    "GameSessionQueueArn": "arn:aws:gamelift:us-west-2::gamesessionqueue/
MegaFrogRace-EU"
  }
]
}

```

For more information, see [Using Multi-Region Queues](#) in the *Amazon GameLift Developer Guide*.

- For API details, see [DescribeGameSessionQueues](#) in *AWS CLI Command Reference*.

describe-runtime-configuration

The following code example shows how to use `describe-runtime-configuration`.

AWS CLI

To request the runtime configuration for a fleet

The following `describe-runtime-configuration` example retrieves details about the current runtime configuration for a specified fleet.

```

aws gamelift describe-runtime-configuration \
  --fleet-id fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111

```

Output:

```

{
  "RuntimeConfiguration": {
    "ServerProcesses": [
      {
        "LaunchPath": "C:\game\Bin64.Release.Dedicated
\MegaFrogRace_Server.exe",
        "Parameters": "+gamelift_start_server",
        "ConcurrentExecutions": 3
      },
      {
        "LaunchPath": "C:\game\Bin64.Release.Dedicated
\MegaFrogRace_Server.exe",
        "Parameters": "+gamelift_start_server +debug",

```

```
        "ConcurrentExecutions": 1
      }
    ],
    "MaxConcurrentGameSessionActivations": 2147483647,
    "GameSessionActivationTimeoutSeconds": 300
  }
}
```

For more information, see [Run Multiple Processes on a Fleet](#) in the *Amazon GameLift Developer Guide*.

- For API details, see [DescribeRuntimeConfiguration](#) in *AWS CLI Command Reference*.

list-builds

The following code example shows how to use `list-builds`.

AWS CLI

Example1: To get a list of custom game builds

The following `list-builds` example retrieves properties for all game server builds in the current Region. The sample request illustrates how to use the pagination parameters, `Limit` and `NextToken`, to retrieve the results in sequential sets. The first command retrieves the first two builds. Because there are more than two available, the response includes a `NextToken` to indicate that more results are available.

```
aws gamelift list-builds \
  --limit 2
```

Output:

```
{
  "Builds": [
    {
      "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "CreationTime": 1495664528.723,
      "Name": "My_Game_Server_Build_One",
      "OperatingSystem": "WINDOWS_2012",
```

```

        "SizeOnDisk": 8567781,
        "Status": "READY",
        "Version": "12345.678"
    },
    {
        "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-
cdef-EXAMPLE22222",
        "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "CreationTime": 1495528748.555,
        "Name": "My_Game_Server_Build_Two",
        "OperatingSystem": "AMAZON_LINUX_2",
        "SizeOnDisk": 8567781,
        "Status": "FAILED",
        "Version": "23456.789"
    }
],
"NextToken":
"eyJhd3NBdW50SWQiOnsicyI6IjMwMjc3NjAxNjM5OCJ9LCJidWlsZElkIjpw7InMiOiJidWlsZC01NWYxZTZmMS1
}

```

You can then call the command again with the `--next-token` parameter as follows to see the next two builds.

```

aws gamelift list-builds \
  --limit 2
  --next-token
eyJhd3NBdW50SWQiOnsicyI6IjMwMjc3NjAxNjM5OCJ9LCJidWlsZElkIjpw7InMiOiJidWlsZC01NWYxZTZmMS1

```

Repeat until the response doesn't include a `NextToken` value.

Example2: To get a list of custom game builds in failure status

The following `list-builds` example retrieves properties for all game server builds in the current region that currently have status `FAILED`.

```

aws gamelift list-builds \
  --status FAILED

```

Output:

```
{
```

```
"Builds": [
  {
    "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "CreationTime": 1495528748.555,
    "Name": "My_Game_Server_Build_Two",
    "OperatingSystem": "AMAZON_LINUX_2",
    "SizeOnDisk": 8567781,
    "Status": "FAILED",
    "Version": "23456.789"
  }
]
```

- For API details, see [ListBuilds](#) in *AWS CLI Command Reference*.

list-fleets

The following code example shows how to use `list-fleets`.

AWS CLI

Example1: To get a list of all fleets in a Region

The following `list-fleets` example displays the fleet IDs of all fleets in the current Region. This example uses pagination parameters to retrieve two fleet IDs at a time. The response includes a `next-token` attribute, which indicates that there are more results to retrieve.

```
aws gamelift list-fleets \
  --limit 2
```

Output:

```
{
  "FleetIds": [
    "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
  ],
  "NextToken":
    "eyJhd3NBZ2NvdW50SWQiOmsicyI6IjMwMjc3NjAxNjM5OCJ9LCJidWlsZElkIjp7InMiOiJidWlsZC01NWYxZTZmMS"
}
```

```
}
```

You can pass the `NextToken` value from the previous response in the next command, as shown here to get the next two results.

```
aws gamelift list-fleets \  
  --limit 2 \  
  --next-token  
eyJhd3NBW50SWQ0OncicyI6IjMwMjc3NjAxNjM50Cj9LCjIdW1sZE1kIj7InMi0iJidW1sZC00NDRlZjQxZS1
```

Example2: To get a list of all fleets in a Region with a specific build or script

The following `list-builds` example retrieves the IDs of fleets that are deployed with the specified game build. If you're working with Realtime Servers, you can provide a script ID in place of a build ID. Because this example does not specify the `limit` parameter, the results can include up to 16 fleet IDs.

```
aws gamelift list-fleets \  
  --build-id build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{  
  "FleetIds": [  
    "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",  
    "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",  
    "fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE44444"  
  ]  
}
```

- For API details, see [ListFleets](#) in *AWS CLI Command Reference*.

request-upload-credentials

The following code example shows how to use `request-upload-credentials`.

AWS CLI

To refresh access credentials for uploading a build

The following `create-build` example obtains new, valid access credentials for uploading a GameLift build file to an Amazon S3 location. Credentials have a limited life span. You get the build ID from the response to the original `CreateBuild` request.

```
aws gamelift request-upload-credentials \  
  --build-id build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{  
  "StorageLocation": {  
    "Bucket": "gamelift-builds-us-west-2",  
    "Key": "123456789012/build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"  
  },  
  "UploadCredentials": {  
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",  
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY",  
    "SessionToken": "AgoGb3JpZ21uENZ...EXAMPLETOKEN=="  
  }  
}
```

For more information, see [Upload a Custom Server Build to GameLift](#) in the *Amazon GameLift Developer Guide*.

- For API details, see [RequestUploadCredentials](#) in *AWS CLI Command Reference*.

start-fleet-actions

The following code example shows how to use `start-fleet-actions`.

AWS CLI

To restart fleet automatic scaling activity

The following `start-fleet-actions` example resumes the use of all scaling policies that are defined for the specified fleet but were stopped by calling `stop-fleet-actions`. After starting, the scaling policies immediately begin tracking their respective metrics.

```
aws gamelift start-fleet-actions \  
  --fleet-id fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \  
  --actions AUTO_SCALING
```

This command produces no output.

- For API details, see [StartFleetActions](#) in *AWS CLI Command Reference*.

stop-fleet-actions

The following code example shows how to use stop-fleet-actions.

AWS CLI

To stop a fleet's automatic scaling activity

The following stop-fleet-actions example stops the use of all scaling policies that are defined for the specified fleet. After the policies are suspended, fleet capacity remains at the same active instance count unless you adjust it manually.

```
aws gamelift start-fleet-actions \  
  --fleet-id fleet-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \  
  --actions AUTO_SCALING
```

This command produces no output.

- For API details, see [StopFleetActions](#) in *AWS CLI Command Reference*.

update-build

The following code example shows how to use update-build.

AWS CLI

To update a custom game build

The following update-build example changes the name and version information that is associated with a specified build resource. The returned build object verifies that the changes were made successfully.

```
aws gamelift update-build \  
  --build-id build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \  
  --name MegaFrogRaceServer.NA.east \  
  --build-version 12345.east
```

Output:

```
{
  "Build": {
    "BuildArn": "arn:aws:gamelift:us-west-2::build/build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "BuildId": "build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "CreationTime": 1496708916.18,
    "Name": "MegaFrogRaceServer.NA.east",
    "OperatingSystem": "AMAZON_LINUX_2",
    "SizeOnDisk": 1304924,
    "Status": "READY",
    "Version": "12345.east"
  }
}
```

For more information, see [Update Your Build Files](#) in the *Amazon GameLift Developer Guide*.

- For API details, see [UpdateBuild](#) in *AWS CLI Command Reference*.

update-game-session-queue

The following code example shows how to use `update-game-session-queue`.

AWS CLI

To update a game session queue configuration

The following `update-game-session-queue` example adds a new destination and updates the player latency policies for an existing game session queue.

```
aws gamelift update-game-session-queue \
  --name MegaFrogRace-NA \
  --destinations file://destinations.json \
  --player-latency-policies file://latency-policies.json
```

Contents of `destinations.json`:

```
{
  "Destinations": [
    {"DestinationArn": "arn:aws:gamelift:us-west-2::fleet/fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d"},
    {"DestinationArn": "arn:aws:gamelift:us-east-1::fleet/fleet-5c6d3c4d-5e6f-7a8b-9c0d-1e2f3a4b5a2b"},
  ]
}
```

```

    {"DestinationArn": "arn:aws:gamelift:us-east-1::alias/
alias-11aa22bb-3c4d-5e6f-000a-1111aaaa22bb"}
  ]
}

```

Contents of latency-policies.json:

```

{
  "PlayerLatencyPolicies": [
    {"MaximumIndividualPlayerLatencyMilliseconds": 200},
    {"MaximumIndividualPlayerLatencyMilliseconds": 150, "PolicyDurationSeconds":
120},
    {"MaximumIndividualPlayerLatencyMilliseconds": 100, "PolicyDurationSeconds":
120}
  ]
}

```

Output:

```

{
  "GameSessionQueue": {
    "Destinations": [
      {"DestinationArn": "arn:aws:gamelift:us-west-2::fleet/
fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d"},
      {"DestinationArn": "arn:aws:gamelift:us-east-1::fleet/
fleet-5c6d3c4d-5e6f-7a8b-9c0d-1e2f3a4b5a2b"},
      {"DestinationArn": "arn:aws:gamelift:us-east-1::alias/
alias-11aa22bb-3c4d-5e6f-000a-1111aaaa22bb"}
    ],
    "GameSessionQueueArn": "arn:aws:gamelift:us-
west-2:111122223333:gamesessionqueue/MegaFrogRace-NA",
    "Name": "MegaFrogRace-NA",
    "TimeoutInSeconds": 600,
    "PlayerLatencyPolicies": [
      {"MaximumIndividualPlayerLatencyMilliseconds": 200},
      {"MaximumIndividualPlayerLatencyMilliseconds": 150,
"PolicyDurationSeconds": 120},
      {"MaximumIndividualPlayerLatencyMilliseconds": 100,
"PolicyDurationSeconds": 120}
    ]
  }
}

```

For more information, see [Using Multi-Region Queues](#) in the *Amazon GameLift Developer Guide*.

- For API details, see [UpdateGameSessionQueue](#) in *AWS CLI Command Reference*.

upload-build

The following code example shows how to use upload-build.

AWS CLI

Example1: To upload a Linux game server build

The following upload-build example uploads Linux game server build files from a file directory to the GameLift service and creates a build resource.

```
aws gamelift upload-build \  
  --name MegaFrogRaceServer.NA \  
  --build-version 2.0.1 \  
  --build-root ~/MegaFrogRace_Server/release-na \  
  --operating-system AMAZON_LINUX_2 \  
  --server-sdk-version 4.0.2
```

Output:

```
Uploading ~/MegaFrogRace_Server/release-na: 16.0 KiB / 74.6 KiB (21.45%)  
Uploading ~/MegaFrogRace_Server/release-na: 32.0 KiB / 74.6 KiB (42.89%)  
Uploading ~/MegaFrogRace_Server/release-na: 48.0 KiB / 74.6 KiB (64.34%)  
Uploading ~/MegaFrogRace_Server/release-na: 64.0 KiB / 74.6 KiB (85.79%)  
Uploading ~/MegaFrogRace_Server/release-na: 74.6 KiB / 74.6 KiB (100.00%)  
Successfully uploaded ~/MegaFrogRace_Server/release-na to AWS GameLift  
Build ID: build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Example2: To upload a Windows game server build

The following upload-build example uploads Windows game server build files from a directory to the GameLift service and creates a build record.

```
aws gamelift upload-build \  
  --name MegaFrogRaceServer.NA \  
  --build-version 2.0.1 \  
  --build-root C:\MegaFrogRace_Server\release-na \  
  --operating-system WINDOWS_2012
```

```
--server-sdk-version 4.0.2
```

Output:

```
Uploading C:\MegaFrogRace_Server\release-na: 16.0 KiB / 74.6 KiB (21.45%)
Uploading C:\MegaFrogRace_Server\release-na: 32.0 KiB / 74.6 KiB (42.89%)
Uploading C:\MegaFrogRace_Server\release-na: 48.0 KiB / 74.6 KiB (64.34%)
Uploading C:\MegaFrogRace_Server\release-na: 64.0 KiB / 74.6 KiB (85.79%)
Uploading C:\MegaFrogRace_Server\release-na: 74.6 KiB / 74.6 KiB (100.00%)
Successfully uploaded C:\MegaFrogRace_Server\release-na to AWS GameLift
Build ID: build-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

For more information, see [Upload a Custom Server Build to GameLift](#) in the *Amazon GameLift Developer Guide*.

- For API details, see [UploadBuild](#) in *AWS CLI Command Reference*.

Global Accelerator examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Global Accelerator.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-custom-routing-endpoints

The following code example shows how to use `add-custom-routing-endpoints`.

AWS CLI

To add a VPC subnet endpoint to an endpoint group for a custom routing accelerator

The following `add-custom-routing-endpoints` example adds a VPC subnet endpoint to an endpoint group for a custom routing accelerator.

```
aws globalaccelerator add-custom-routing-endpoints \  
  --endpoint-group-arn  
  arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-  
abcd-1234abcdefgh/listener/0123vxyz/endpoint-group/4321abcd \  
  --endpoint-configurations "EndpointId=subnet-1234567890abcdef0"
```

Output:

```
{  
  "EndpointDescriptions": [  
    {  
      "EndpointId": "subnet-1234567890abcdef0"  
    }  
  ],  
  
  "EndpointGroupArn": "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-  
abcd-1234-abcd-1234abcdefgh/listener/0123vxyz/endpoint-group/4321abcd"  
}
```

For more information, see [VPC subnet endpoints for custom routing accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [AddCustomRoutingEndpoints](#) in *AWS CLI Command Reference*.

advertise-byoip-cidr

The following code example shows how to use `advertise-byoip-cidr`.

AWS CLI

To advertise an address range

The following `advertise-byoip-cidr` example requests AWS to advertise an address range that you've provisioned for use with your AWS resources.

```
aws globalaccelerator advertise-byoip-cidr \  
  --
```

```
--cidr 198.51.100.0/24
```

Output:

```
{
  "ByoipCidr": {
    "Cidr": "198.51.100.0/24",
    "State": "PENDING_ADVERTISING"
  }
}
```

For more information, see [Bring Your Own IP Address in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [AdvertiseByoipCidr](#) in *AWS CLI Command Reference*.

allow-custom-routing-traffic

The following code example shows how to use `allow-custom-routing-traffic`.

AWS CLI

To allow traffic to specific Amazon EC2 instance destinations in a VPC subnet for a custom routing accelerator

The following `allow-custom-routing-traffic` example specifies that traffic is allowed to certain Amazon EC2 instance (destination) IP addresses and ports for a VPC subnet endpoint in a custom routing accelerator can receive traffic.

```
aws globalaccelerator allow-custom-routing-traffic \
  --endpoint-group-arn
  arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
  abcd-1234abcdefgh/listener/0123vxyz/endpoint-group/ab8888example \
  --endpoint-id subnet-abcd123example \
  --destination-addresses "172.31.200.6" "172.31.200.7" \
  --destination-ports 80 81
```

This command produces no output.

For more information, see [VPC subnet endpoints for custom routing accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [AllowCustomRoutingTraffic](#) in *AWS CLI Command Reference*.

create-accelerator

The following code example shows how to use `create-accelerator`.

AWS CLI

To create an accelerator

The following `create-accelerator` example creates an accelerator with two tags with two BYOIP static IP addresses. You must specify the US-West-2 (Oregon) Region to create or update an accelerator.

```
aws globalaccelerator create-accelerator \  
  --name ExampleAccelerator \  
  --tags Key="Name",Value="Example Name" Key="Project",Value="Example Project" \  
  --ip-addresses 192.0.2.250 198.51.100.52
```

Output:

```
{  
  "Accelerator": {  
    "AcceleratorArn":  
      "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-  
abcd-1234abcdefgh",  
    "IpAddressType": "IPV4",  
    "Name": "ExampleAccelerator",  
    "Enabled": true,  
    "Status": "IN_PROGRESS",  
    "IpSets": [  
      {  
        "IpAddresses": [  
          "192.0.2.250",  
          "198.51.100.52"  
        ],  
        "IpFamily": "IPv4"  
      }  
    ],  
    "DnsName": "a1234567890abcdef.awsglobalaccelerator.com",  
    "CreatedTime": 1542394847.0,  
    "LastModifiedTime": 1542394847.0
```

```
}  
}
```

For more information, see [Accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [CreateAccelerator](#) in *AWS CLI Command Reference*.

create-custom-routing-accelerator

The following code example shows how to use `create-custom-routing-accelerator`.

AWS CLI

To create a custom routing accelerator

The following `create-custom-routing-accelerator` example creates a custom routing accelerator with the tags `Name` and `Project`.

```
aws globalaccelerator create-custom-routing-accelerator \  
  --name ExampleCustomRoutingAccelerator \  
  --tags Key="Name",Value="Example Name" Key="Project",Value="Example Project" \  
  --ip-addresses 192.0.2.250 198.51.100.52
```

Output:

```
{  
  "Accelerator": {  
    "AcceleratorArn":  
      "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-  
abcd-1234abcdefg",  
    "IpAddressType": "IPV4",  
    "Name": "ExampleCustomRoutingAccelerator",  
    "Enabled": true,  
    "Status": "IN_PROGRESS",  
    "IpSets": [  
      {  
        "IpAddresses": [  
          "192.0.2.250",  
          "198.51.100.52"  
        ],  
        "IpFamily": "IPv4"  
      }  
    ]  
  }  
}
```

```

    }
  ],
  "DnsName": "a1234567890abcdef.awsglobalaccelerator.com",
  "CreatedTime": 1542394847.0,
  "LastModifiedTime": 1542394847.0
}
}

```

For more information, see [Custom routing accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [CreateCustomRoutingAccelerator](#) in *AWS CLI Command Reference*.

create-custom-routing-endpoint-group

The following code example shows how to use `create-custom-routing-endpoint-group`.

AWS CLI

To create an endpoint group for a custom routing accelerator

The following `create-custom-routing-endpoint-group` example creates an endpoint group for a custom routing accelerator.

```

aws globalaccelerator create-custom-routing-endpoint-group \
  --listener-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh/listener/0123vxyz \
  --endpoint-group-region us-east-2 \
  --destination-configurations "FromPort=80,ToPort=81,Protocols=TCP,UDP"

```

Output:

```

{
  "EndpointGroup": {
    "EndpointGroupArn":
    "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/0123vxyz/endpoint-group/4321abcd",
    "EndpointGroupRegion": "us-east-2",
    "DestinationDescriptions": [
      {
        "FromPort": 80,
        "ToPort": 81,

```

```

        "Protocols": [
            "TCP",
            "UDP"
        ]
    },
    "EndpointDescriptions": []
}

```

For more information, see [Endpoint groups for custom routing accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [CreateCustomRoutingEndpointGroup](#) in *AWS CLI Command Reference*.

create-custom-routing-listener

The following code example shows how to use `create-custom-routing-listener`.

AWS CLI

To create a listener for a custom routing accelerator

The following `create-custom-routing-listener` example creates a listener with a port range from 5000 to 10000 for a custom routing accelerator.

```

aws globalaccelerator create-custom-routing-listener \
  --accelerator-arn arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh \
  --port-ranges FromPort=5000,ToPort=10000

```

Output:

```

{
  "Listener": {
    "PortRange": [
      "FromPort": 5000,
      "ToPort": 10000
    ],
    "ListenerArn":
    "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/0123vxyz"
  }
}

```

```
}
}
```

For more information, see [Listeners for custom routing accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [CreateCustomRoutingListener](#) in *AWS CLI Command Reference*.

create-endpoint-group

The following code example shows how to use `create-endpoint-group`.

AWS CLI

To create an endpoint group

The following `create-endpoint-group` example creates an endpoint group with one endpoint.

```
aws globalaccelerator create-endpoint-group \
  --listener-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh/listener/0123vxyz \
  --endpoint-group-region us-east-1 \
  --endpoint-configurations EndpointId=i-1234567890abcdef0,Weight=128
```

Output:

```
{
  "EndpointGroup": {
    "TrafficDialPercentage": 100.0,
    "EndpointDescriptions": [
      {
        "Weight": 128,
        "EndpointId": "i-1234567890abcdef0"
      }
    ],
    "EndpointGroupArn":
    "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/0123vxyz/endpoint-group/098765zyxwvu",
    "EndpointGroupRegion": "us-east-1"
  }
}
```

```
}
```

For more information, see [Endpoint groups in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [CreateEndpointGroup](#) in *AWS CLI Command Reference*.

create-listener

The following code example shows how to use `create-listener`.

AWS CLI

To create a listener

The following `create-listener` example creates a listener with two ports.

```
aws globalaccelerator create-listener \  
  --accelerator-arn arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-  
abcd-1234-abcd-1234abcdefgh \  
  --port-ranges FromPort=80,ToPort=80 FromPort=81,ToPort=81 \  
  --protocol TCP
```

Output:

```
{  
  "Listener": {  
    "PortRanges": [  
      {  
        "ToPort": 80,  
        "FromPort": 80  
      },  
      {  
        "ToPort": 81,  
        "FromPort": 81  
      }  
    ],  
    "ClientAffinity": "NONE",  
    "Protocol": "TCP",  
    "ListenerArn":  
    "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-  
abcd-1234abcdefgh/listener/0123vxyz"
```

```
}  
}
```

For more information, see [Listeners in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [CreateListener](#) in *AWS CLI Command Reference*.

deny-custom-routing-traffic

The following code example shows how to use deny-custom-routing-traffic.

AWS CLI

To specify a destination address that cannot receive traffic in a custom routing accelerator

The following deny-custom-routing-traffic example specifies destination address or addresses in a subnet endpoint that cannot receive traffic for a custom routing accelerator. To specify more than one destination address, separate the addresses with a space. There's no response for a successful deny-custom-routing-traffic call.

```
aws globalaccelerator deny-custom-routing-traffic \  
  --endpoint-group-arn  
  "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-  
abcd-1234abcdefgh/listener/0123vxyz/endpoint-group/ab8888example" \  
  --endpoint-id "subnet-abcd123example" \  
  --destination-addresses "198.51.100.52"
```

This command produces no output.

For more information, see [VPC subnet endpoints for custom routing accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [DenyCustomRoutingTraffic](#) in *AWS CLI Command Reference*.

deprovision-byoip-cidr

The following code example shows how to use deprovision-byoip-cidr.

AWS CLI

To deprovision an address range

The following `deprovision-byoip-cidr` example releases the specified address range that you provisioned to use with your AWS resources.

```
aws globalaccelerator deprovision-byoip-cidr \  
  --cidr "198.51.100.0/24"
```

Output:

```
{  
  "ByoipCidr": {  
    "Cidr": "198.51.100.0/24",  
    "State": "PENDING_DEPROVISIONING"  
  }  
}
```

For more information, see [Bring your own IP address in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [DeprovisionByoipCidr](#) in *AWS CLI Command Reference*.

describe-accelerator-attributes

The following code example shows how to use `describe-accelerator-attributes`.

AWS CLI

To describe an accelerator's attributes

The following `describe-accelerator-attributes` example retrieves the attribute details for an accelerator.

```
aws globalaccelerator describe-accelerator-attributes \  
  --accelerator-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-  
abcd-1234-abcd-1234abcdefgh
```

Output:

```
{  
  "AcceleratorAttributes": {  
    "FlowLogsEnabled": true
```



```
    "FlowLogsS3Bucket": flowlogs-abc
    "FlowLogsS3Prefix": bucketprefix-abc
  }
}
```

For more information, see [Accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [DescribeAcceleratorAttributes](#) in *AWS CLI Command Reference*.

describe-accelerator

The following code example shows how to use `describe-accelerator`.

AWS CLI

To describe an accelerator

The following `describe-accelerator` example retrieves the details about the specified accelerator.

```
aws globalaccelerator describe-accelerator \
  --accelerator-arn arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh
```

Output:

```
{
  "Accelerator": {
    "AcceleratorArn":
"arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh",
    "IpAddressType": "IPV4",
    "Name": "ExampleAccelerator",
    "Enabled": true,
    "Status": "IN_PROGRESS",
    "IpSets": [
      {
        "IpAddresses": [
          "192.0.2.250",
          "198.51.100.52"
        ]
      }
    ]
  }
}
```

```
        "IpFamily": "IPv4"
      }
    ],
    "DnsName": "a1234567890abcdef.awsglobalaccelerator.com",
    "CreatedTime": 1542394847,
    "LastModifiedTime": 1542395013
  }
}
```

For more information, see [Accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [DescribeAccelerator](#) in *AWS CLI Command Reference*.

describe-custom-routing-accelerator-attributes

The following code example shows how to use `describe-custom-routing-accelerator-attributes`.

AWS CLI

To describe a custom routing accelerator's attributes

The following `describe-custom-routing-accelerator-attributes` example describes the attributes for a custom routing accelerator.

```
aws globalaccelerator describe-custom-routing-accelerator-attributes \
  --accelerator-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh
```

Output:

```
{
  "AcceleratorAttributes": {
    "FlowLogsEnabled": false
  }
}
```

For more information, see [Custom routing accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [DescribeCustomRoutingAcceleratorAttributes](#) in *AWS CLI Command Reference*.

describe-custom-routing-accelerator

The following code example shows how to use `describe-custom-routing-accelerator`.

AWS CLI

To describe a custom routing accelerator

The following `describe-custom-routing-accelerator` example retrieves the details about the specified custom routing accelerator.

```
aws globalaccelerator describe-custom-routing-accelerator \
  --accelerator-arn arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh
```

Output:

```
{
  "Accelerator": {
    "AcceleratorArn":
    "arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh",
    "IpAddressType": "IPv4",
    "Name": "ExampleCustomRoutingAccelerator",
    "Enabled": true,
    "Status": "IN_PROGRESS",
    "IpSets": [
      {
        "IpAddresses": [
          "192.0.2.250",
          "198.51.100.52"
        ],
        "IpFamily": "IPv4"
      }
    ],
    "DnsName": "a1234567890abcdef.awsglobalaccelerator.com",
    "CreatedTime": 1542394847,
    "LastModifiedTime": 1542395013
  }
}
```

```
}
```

For more information, see [Custom routing accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [DescribeCustomRoutingAccelerator](#) in *AWS CLI Command Reference*.

describe-custom-routing-endpoint-group

The following code example shows how to use `describe-custom-routing-endpoint-group`.

AWS CLI

To describe an endpoint group for a custom routing accelerator

The following `describe-custom-routing-endpoint-group` example describes an endpoint group for a custom routing accelerator.

```
aws globalaccelerator describe-custom-routing-endpoint-group \  
  --endpoint-group-arn  
  arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-  
abcd-1234abcdefgh/listener/6789vxyz/endpoint-group/ab88888example
```

Output:

```
{  
  "EndpointGroup": {  
    "EndpointGroupArn":  
    "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-  
abcd-1234abcdefgh/listener/6789vxyz/endpoint-group/ab88888example",  
    "EndpointGroupRegion": "us-east-2",  
    "DestinationDescriptions": [  
      {  
        "FromPort": 5000,  
        "ToPort": 10000,  
        "Protocols": [  
          "UDP"  
        ]  
      }  
    ],  
    "EndpointDescriptions": [  
      {
```

```

        "EndpointId": "subnet-1234567890abcdef0"
      }
    ]
  }
}

```

For more information, see [Endpoint groups for custom routing accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [DescribeCustomRoutingEndpointGroup](#) in *AWS CLI Command Reference*.

describe-custom-routing-listener

The following code example shows how to use `describe-custom-routing-listener`.

AWS CLI

To describe a listener for a custom routing accelerator

The following `describe-custom-routing-listener` example describes a listener for a custom routing accelerator.

```

aws globalaccelerator describe-custom-routing-listener \
  --listener-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh/listener/abcdef1234

```

Output:

```

{
  "Listener": {
    "PortRanges": [
      "FromPort": 5000,
      "ToPort": 10000
    ],
    "ListenerArn":
    "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/abcdef1234"
  }
}

```

For more information, see [Listeners for custom routing accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [DescribeCustomRoutingListener](#) in *AWS CLI Command Reference*.

describe-endpoint-group

The following code example shows how to use `describe-endpoint-group`.

AWS CLI

To describe an endpoint group

The following `describe-endpoint-group` example retrieves details about an endpoint group with the following endpoints: an Amazon EC2 instance, an ALB, and an NLB.

```
aws globalaccelerator describe-endpoint-group \  
  --endpoint-group-arn \  
  arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-  
abcd-1234abcdefgh/listener/6789vxyz-vxyz-6789-vxyz-6789lmnopqrs/endpoint-group/  
ab88888example
```

Output:

```
{  
  "EndpointGroup": {  
    "TrafficDialPercentage": 100.0,  
    "EndpointDescriptions": [  
      {  
        "Weight": 128,  
        "EndpointId": "i-1234567890abcdef0"  
      },  
      {  
        "Weight": 128,  
        "EndpointId": "arn:aws:elasticloadbalancing:us-  
east-1:000123456789:loadbalancer/app/ALBTesting/alb01234567890xyz"  
      },  
      {  
        "Weight": 128,  
        "EndpointId": "arn:aws:elasticloadbalancing:us-  
east-1:000123456789:loadbalancer/net/NLBTesting/alb01234567890qrs"  
      }  
    ],  
    "EndpointGroupArn":  
    "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
```

```
abcd-1234abcdefgh/listener/6789vxyz-vxyz-6789-vxyz-6789lmnopqrs/endpoint-
group/4321abcd-abcd-4321-abcd-4321abcdefg",
    "EndpointGroupRegion": "us-east-1"
  }
}
```

For more information, see [Endpoint groups in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [DescribeEndpointGroup](#) in *AWS CLI Command Reference*.

describe-listener

The following code example shows how to use `describe-listener`.

AWS CLI

To describe a listener

The following `describe-listener` example describes a listener.

```
aws globalaccelerator describe-listener \
  --listener-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh/listener/abcdef1234
```

Output:

```
{
  "Listener": {
    "ListenerArn":
    "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/abcdef1234",
    "PortRanges": [
      {
        "FromPort": 80,
        "ToPort": 80
      }
    ],
    "Protocol": "TCP",
    "ClientAffinity": "NONE"
  }
}
```


Output:

```
{
  "ByoipCidrs": [
    {
      "Cidr": "198.51.100.0/24",
      "State": "READY"
    }
    {
      "Cidr": "203.0.113.25/24",
      "State": "READY"
    }
  ]
}
```

For more information, see [Bring your own IP address in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [ListByoipCidr](#) in *AWS CLI Command Reference*.

list-custom-routing-accelerators

The following code example shows how to use `list-custom-routing-accelerators`.

AWS CLI**To list your custom routing accelerators**

The following `list-custom-routing-accelerators` example lists the custom routing accelerators in an AWS account.

```
aws globalaccelerator list-custom-routing-accelerators
```

Output:

```
{
  "Accelerators": [
    {
      "AcceleratorArn":
        "arn:aws:globalaccelerator::012345678901:accelerator/5555abcd-abcd-5555-
        abcd-5555EXAMPLE1",
      "Name": "TestCustomRoutingAccelerator",
```


list-custom-routing-endpoint-groups

The following code example shows how to use `list-custom-routing-endpoint-groups`.

AWS CLI

To list endpoint groups for a listener in a custom routing accelerator

The following `list-custom-routing-endpoint-groups` example lists the endpoint groups for a listener in a custom routing accelerator.

```
aws globalaccelerator list-custom-routing-endpoint-groups \  
  --listener-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-  
abcd-1234-abcd-1234abcdefgh/listener/abcdef1234
```

Output:

```
{  
  "EndpointGroups": [  
    {  
      "EndpointGroupArn":  
"arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-  
abcd-1234abcdefgh/listener/abcdef1234/endpoint-group/ab8888example",  
      "EndpointGroupRegion": "eu-central-1",  
      "DestinationDescriptions": [  
        {  
          "FromPort": 80,  
          "ToPort": 80,  
          "Protocols": [  
            "TCP",  
            "UDP"  
          ]  
        }  
      ]  
      "EndpointDescriptions": [  
        {  
          "EndpointId": "subnet-abcd123example"  
        }  
      ]  
    }  
  ]  
}
```

For more information, see [Endpoint groups for custom routing accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [ListCustomRoutingEndpointGroups](#) in *AWS CLI Command Reference*.

list-custom-routing-listeners

The following code example shows how to use `list-custom-routing-listeners`.

AWS CLI

To list listeners for custom routing accelerators

The following `list-custom-routing-listeners` example lists the listeners for a custom routing accelerator.

```
aws globalaccelerator list-custom-routing-listeners \
  --accelerator-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh
```

Output:

```
{
  "Listeners": [
    {
      "ListenerArn":
      "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/abcdef1234",
      "PortRanges": [
        {
          "FromPort": 5000,
          "ToPort": 10000
        }
      ],
      "Protocol": "TCP"
    }
  ]
}
```

For more information, see [Listeners for custom routing accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [ListCustomRoutingListeners](#) in *AWS CLI Command Reference*.

list-custom-routing-port-mappings-by-destination

The following code example shows how to use `list-custom-routing-port-mappings-by-destination`.

AWS CLI

To list the port mappings for a specific custom routing accelerator destination

The following `list-custom-routing-port-mappings-by-destination` example provides the port mappings for a specific destination EC2 server (at the destination address) for a custom routing accelerator.

```
aws globalaccelerator list-custom-routing-port-mappings-by-destination \  
  --endpoint-id subnet-abcd123example \  
  --destination-address 198.51.100.52
```

Output:

```
{  
  "DestinationPortMappings": [  
    {  
      "AcceleratorArn":  
        "arn:aws:globalaccelerator::402092451327:accelerator/24ea29b8-  
d750-4489-8919-3095f3c4b0a7",  
      "AcceleratorSocketAddresses": [  
        {  
          "IpAddress": "192.0.2.250",  
          "Port": 65514  
        },  
        {  
          "IpAddress": "192.10.100.99",  
          "Port": 65514  
        }  
      ],  
      "EndpointGroupArn":  
        "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-  
abcd-1234abcdefgh/listener/0123vxyz/endpoint-group/ab88888example",  
      "EndpointId": "subnet-abcd123example",  
      "EndpointGroupRegion": "us-west-2",  
      "DestinationSocketAddress": {  
        "IpAddress": "198.51.100.52",
```

```

        "Port": 80
      },
      "IpAddressType": "IPv4",
      "DestinationTrafficState": "ALLOW"
    }
  ]
}

```

For more information, see [How custom routing accelerators work in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [ListCustomRoutingPortMappingsByDestination](#) in *AWS CLI Command Reference*.

list-custom-routing-port-mappings

The following code example shows how to use `list-custom-routing-port-mappings`.

AWS CLI

To list the port mappings in a custom routing accelerator

The following `list-custom-routing-port-mappings` example provides a partial list of the port mappings in a custom routing accelerator.

```

aws globalaccelerator list-custom-routing-port-mappings \
  --accelerator-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh

```

Output:

```

{
  "PortMappings": [
    {
      "AcceleratorPort": 40480,
      "EndpointGroupArn":
"arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/0123vxyz/endpoint-group/098765zyxwvu",
      "EndpointId": "subnet-1234567890abcdef0",
      "DestinationSocketAddress": {
        "IpAddress": "192.0.2.250",
        "Port": 80
      }
    }
  ]
}

```

```

    },
    "Protocols": [
        "TCP",
        "UDP"
    ],
    "DestinationTrafficState": "ALLOW"
}
{
    "AcceleratorPort": 40481,
    "EndpointGroupArn":
"arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/0123vxyz/endpoint-group/098765zyxwvu",
    "EndpointId": "subnet-1234567890abcdef0",
    "DestinationSocketAddress": {
        "IpAddress": "192.0.2.251",
        "Port": 80
    },
    "Protocols": [
        "TCP",
        "UDP"
    ],
    "DestinationTrafficState": "ALLOW"
}
]
}

```

For more information, see [How custom routing accelerators work in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [ListCustomRoutingPortMappings](#) in *AWS CLI Command Reference*.

list-endpoint-groups

The following code example shows how to use `list-endpoint-groups`.

AWS CLI

To list endpoint groups

The following `list-endpoint-groups` example lists the endpoint groups for a listener. This listener has two endpoint groups.

```
aws globalaccelerator --region us-west-2 list-endpoint-groups \
```



```
--listener-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-  
abcd-1234-abcd-1234abcdefgh/listener/abcdef1234
```

Output:

```
{  
  "EndpointGroups": [  
    {  
      "EndpointGroupArn":  
"arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-  
abcd-1234abcdefgh/listener/abcdef1234/endpoint-group/ab88888example",  
      "EndpointGroupRegion": "eu-central-1",  
      "EndpointDescriptions": [],  
      "TrafficDialPercentage": 100.0,  
      "HealthCheckPort": 80,  
      "HealthCheckProtocol": "TCP",  
      "HealthCheckIntervalSeconds": 30,  
      "ThresholdCount": 3  
    }  
    {  
      "EndpointGroupArn":  
"arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-  
abcd-1234abcdefgh/listener/abcdef1234/endpoint-group/ab99999example",  
      "EndpointGroupRegion": "us-east-1",  
      "EndpointDescriptions": [],  
      "TrafficDialPercentage": 50.0,  
      "HealthCheckPort": 80,  
      "HealthCheckProtocol": "TCP",  
      "HealthCheckIntervalSeconds": 30,  
      "ThresholdCount": 3  
    }  
  ]  
}
```

For more information, see [Endpoint Groups in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [ListEndpointGroups](#) in *AWS CLI Command Reference*.

list-listeners

The following code example shows how to use `list-listeners`.

AWS CLI

To list listeners

The following `list-listeners` example lists the listeners for an accelerator.

```
aws globalaccelerator list-listeners \  
  --accelerator-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-  
abcd-1234-abcd-1234abcdefgh
```

Output:

```
{  
  "Listeners": [  
    {  
      "ListenerArn":  
"arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-  
abcd-1234abcdefgh/listener/abcdef1234",  
      "PortRanges": [  
        {  
          "FromPort": 80,  
          "ToPort": 80  
        }  
      ],  
      "Protocol": "TCP",  
      "ClientAffinity": "NONE"  
    }  
  ]  
}
```

For more information, see [Listeners in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [ListListeners](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags for an accelerator

The following `list-tags-for-resource` example lists the tags for a specific accelerator.

```
aws globalaccelerator list-tags-for-resource \  
  --accelerator-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-  
abcd-1234-abcd-1234abcdefgh
```

Output:

```
{  
  "Tags": [  
    {  
      "Key": "Project",  
      "Value": "A123456"  
    }  
  ]  
}
```

For more information, see [Tagging in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

provision-byoip-cidr

The following code example shows how to use `provision-byoip-cidr`.

AWS CLI

To provision an address range

The following `provision-byoip-cidr` example provisions the specified address range to use with your AWS resources.

```
aws globalaccelerator provision-byoip-cidr \  
  --cidr 192.0.2.250/24 \  
  --cidr-authorization-context Message="$text_message",Signature="$signed_message"
```

Output:

```
{
```

```
"ByoipCidr": {
  "Cidr": "192.0.2.250/24",
  "State": "PENDING_PROVISIONING"
}
```

For more information, see [Bring your own IP address in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [ProvisionByoipCidr](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To tag an accelerator

The following `tag-resource` example adds tags `Name` and `Project` to an accelerator, along with corresponding values for each.

```
aws globalaccelerator tag-resource \
  --resource-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh \
  --tags Key="Name",Value="Example Name" Key="Project",Value="Example Project"
```

This command produces no output.

For more information, see [Tagging in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag from an accelerator

The following `untag-resource` example removes the tags `Name` and `Project` from an accelerator.

```
aws globalaccelerator untag-resource \  
  --resource-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-  
abcd-1234-abcd-1234abcdefgh \  
  --tag-keys Key="Name" Key="Project"
```

This command produces no output.

For more information, see [Tagging in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-accelerator-attributes

The following code example shows how to use `update-accelerator-attributes`.

AWS CLI

To update an accelerator's attributes

The following `update-accelerator-attributes` example updates an accelerator to enable flow logs. You must specify the `US-West-2` (Oregon) Region to create or update accelerator attributes.

```
aws globalaccelerator update-accelerator-attributes \  
  --accelerator-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-  
abcd-1234-abcd-1234abcdefgh \  
  --flow-logs-enabled \  
  --flow-logs-s3-bucket flowlogs-abc \  
  --flow-logs-s3-prefix bucketprefix-abc
```

Output:

```
{  
  "AcceleratorAttributes": {  
    "FlowLogsEnabled": true  
    "FlowLogsS3Bucket": flowlogs-abc  
    "FlowLogsS3Prefix": bucketprefix-abc  
  }  
}
```

```
}
```

For more information, see [Accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [UpdateAcceleratorAttributes](#) in *AWS CLI Command Reference*.

update-accelerator

The following code example shows how to use `update-accelerator`.

AWS CLI

To update an accelerator

The following `update-accelerator` example modifies an accelerator to change the accelerator name to `ExampleAcceleratorNew`. You must specify the `US-West-2` (Oregon) Region to create or update accelerators.

```
aws globalaccelerator update-accelerator \  
  --accelerator-arn arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-  
abcd-1234-abcd-1234abcdefgh \  
  --name ExampleAcceleratorNew
```

Output:

```
{  
  "Accelerator": {  
    "AcceleratorArn":  
    "arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-abcd-1234-  
abcd-1234abcdefgh",  
    "IpAddressType": "IPV4",  
    "Name": "ExampleAcceleratorNew",  
    "Enabled": true,  
    "Status": "IN_PROGRESS",  
    "IpSets": [  
      {  
        "IpAddresses": [  
          "192.0.2.250",  
          "198.51.100.52"  
        ],  
        "IpFamily": "IPv4"  
      }  
    ]  
  }  
}
```

```
    }
  ],
  "DnsName": "a1234567890abcdef.awsglobalaccelerator.com",
  "CreatedTime": 1232394847,
  "LastModifiedTime": 1232395654
}
}
```

For more information, see [Accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [UpdateAccelerator](#) in *AWS CLI Command Reference*.

update-custom-routing-accelerator-attributes

The following code example shows how to use `update-custom-routing-accelerator-attributes`.

AWS CLI

To update a custom routing accelerator's attributes

The following `update-custom-routing-accelerator-attributes` example updates a custom routing accelerator to enable flow logs.

```
aws globalaccelerator update-custom-routing-accelerator-attributes \
  --accelerator-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh \
  --flow-logs-enabled \
  --flow-logs-s3-bucket flowlogs-abc \
  --flow-logs-s3-prefix bucketprefix-abc
```

Output:

```
{
  "AcceleratorAttributes": {
    "FlowLogsEnabled": true
    "FlowLogsS3Bucket": flowlogs-abc
    "FlowLogsS3Prefix": bucketprefix-abc
  }
}
```

For more information, see [Custom routing accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [UpdateCustomRoutingAcceleratorAttributes](#) in *AWS CLI Command Reference*.

update-custom-routing-accelerator

The following code example shows how to use `update-custom-routing-accelerator`.

AWS CLI

To update a custom routing accelerator

The following `update-custom-routing-accelerator` example modifies a custom routing accelerator to change the accelerator name.

```
aws globalaccelerator --region us-west-2 update-custom-routing-accelerator \
  --accelerator-arn arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh \
  --name ExampleCustomRoutingAcceleratorNew
```

Output:

```
{
  "Accelerator": {
    "AcceleratorArn":
"arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh",
    "IpAddressType": "IPV4",
    "Name": "ExampleCustomRoutingAcceleratorNew",
    "Enabled": true,
    "Status": "IN_PROGRESS",
    "IpSets": [
      {
        "IpAddresses": [
          "192.0.2.250",
          "198.51.100.52"
        ],
        "IpFamily": "IPv4"
      }
    ],
    "DnsName": "a1234567890abcdef.awsglobalaccelerator.com",
```



```
    "CreatedTime": 1232394847,  
    "LastModifiedTime": 1232395654  
  }  
}
```

For more information, see [Custom routing accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [UpdateCustomRoutingAccelerator](#) in *AWS CLI Command Reference*.

update-custom-routing-listener

The following code example shows how to use `update-custom-routing-listener`.

AWS CLI

To update a listener for a custom routing accelerator

The following `update-custom-routing-listener` example updates a listener to change the port range.

```
aws globalaccelerator update-custom-routing-listener \  
  --listener-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-  
abcd-1234-abcd-1234abcdefgh/listener/0123vxyz \  
  --port-ranges FromPort=10000,ToPort=20000
```

Output:

```
{  
  "Listener": {  
    "ListenerArn":  
    "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-  
abcd-1234abcdefgh/listener/0123vxyz"  
    "PortRanges": [  
      {  
        "FromPort": 10000,  
        "ToPort": 20000  
      }  
    ],  
    "Protocol": "TCP"  
  }  
}
```

For more information, see [Listeners for custom routing accelerators in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [UpdateCustomRoutingListener](#) in *AWS CLI Command Reference*.

update-endpoint-group

The following code example shows how to use `update-endpoint-group`.

AWS CLI

To update an endpoint group

The following `update-endpoint-group` example adds three endpoints to an endpoint group: an Elastic IP address, an ALB, and an NLB.

```
aws globalaccelerator update-endpoint-group \
  --endpoint-group-arn
  arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/6789vxyz-vxyz-6789-vxyz-6789lmnopqrs/endpoint-group/
ab88888example \
  --endpoint-configurations \
    EndpointId=eipalloc-eip01234567890abc,Weight=128 \
    EndpointId=arn:aws:elasticloadbalancing:us-east-1:000123456789:loadbalancer/
app/ALBTesting/alb01234567890xyz,Weight=128 \
    EndpointId=arn:aws:elasticloadbalancing:us-east-1:000123456789:loadbalancer/
net/NLBTesting/alb01234567890qrs,Weight=128
```

Output:

```
{
  "EndpointGroup": {
    "TrafficDialPercentage": 100,
    "EndpointDescriptions": [
      {
        "Weight": 128,
        "EndpointId": "eip01234567890abc"
      },
      {
        "Weight": 128,
        "EndpointId": "arn:aws:elasticloadbalancing:us-
east-1:000123456789:loadbalancer/app/ALBTesting/alb01234567890xyz"
      },
    ]
  }
}
```

```

    {
      "Weight": 128,
      "EndpointId": "arn:aws:elasticloadbalancing:us-
east-1:000123456789:loadbalancer/net/NLBTesting/alb01234567890qrs"
    }
  ],
  "EndpointGroupArn":
  "arn:aws:globalaccelerator::123456789012:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/6789vxyz-vxyz-6789-vxyz-6789lmnopqrs/endpoint-
group/4321abcd-abcd-4321-abcd-4321abcdefg",
  "EndpointGroupRegion": "us-east-1"
}
}

```

For more information, see [Endpoint groups in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [UpdateEndpointGroup](#) in *AWS CLI Command Reference*.

update-listener

The following code example shows how to use `update-listener`.

AWS CLI

To update a listener

The following `update-listener` example updates a listener to change the port to 100.

```

aws globalaccelerator update-listener \
  --listener-arn arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-
abcd-1234-abcd-1234abcdefgh/listener/0123vxyz \
  --port-ranges FromPort=100,ToPort=100

```

Output:

```

{
  "Listener": {
    "ListenerArn":
    "arn:aws:globalaccelerator::012345678901:accelerator/1234abcd-abcd-1234-
abcd-1234abcdefgh/listener/0123vxyz
    "PortRanges": [

```

```
    {
      "FromPort": 100,
      "ToPort": 100
    }
  ],
  "Protocol": "TCP",
  "ClientAffinity": "NONE"
}
```

For more information, see [Listeners in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [UpdateListener](#) in *AWS CLI Command Reference*.

withdraw-byoip-cidr

The following code example shows how to use `withdraw-byoip-cidr`.

AWS CLI

To withdraw an address range

The following `withdraw-byoip-cidr` example withdraws an address range from AWS Global Accelerator that you previously advertised for use with your AWS resources.

```
aws globalaccelerator withdraw-byoip-cidr \
  --cidr 192.0.2.250/24
```

Output:

```
{
  "ByoipCidr": {
    "Cidr": "192.0.2.250/24",
    "State": "PENDING_WITHDRAWING"
  }
}
```

For more information, see [Bring your own IP address in AWS Global Accelerator](#) in the *AWS Global Accelerator Developer Guide*.

- For API details, see [WithdrawByoipCidr](#) in *AWS CLI Command Reference*.

AWS Glue examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Glue.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

batch-stop-job-run

The following code example shows how to use `batch-stop-job-run`.

AWS CLI

To stop job runs

The following `batch-stop-job-run` example stops a job runs.

```
aws glue batch-stop-job-run \  
  --job-name "my-testing-job" \  
  --job-run-id jr_852f1de1f29fb62e0ba4166c33970803935d87f14f96cfdee5089d5274a61d3f
```

Output:

```
{  
  "SuccessfulSubmissions": [  
    {  
      "JobName": "my-testing-job",  
      "JobRunId":  
        "jr_852f1de1f29fb62e0ba4166c33970803935d87f14f96cfdee5089d5274a61d3f"    }  
  ]  
}
```

```

    }
  ],
  "Errors": [],
  "ResponseMetadata": {
    "RequestId": "66bd6b90-01db-44ab-95b9-6aeff0e73d88",
    "HTTPStatusCode": 200,
    "HTTPHeaders": {
      "date": "Fri, 16 Oct 2020 20:54:51 GMT",
      "content-type": "application/x-amz-json-1.1",
      "content-length": "148",
      "connection": "keep-alive",
      "x-amzn-requestid": "66bd6b90-01db-44ab-95b9-6aeff0e73d88"
    },
    "RetryAttempts": 0
  }
}

```

For more information, see [Job Runs](#) in the *AWS Glue Developer Guide*.

- For API details, see [BatchStopJobRun](#) in *AWS CLI Command Reference*.

create-connection

The following code example shows how to use `create-connection`.

AWS CLI

To create a connection for AWS Glue data stores

The following `create-connection` example creates a connection in the AWS Glue Data Catalog that provides connection information for a Kafka data store.

```

aws glue create-connection \
  --connection-input '{ \
    "Name":"conn-kafka-custom", \
    "Description":"kafka connection with ssl to custom kafka", \
    "ConnectionType":"KAFKA", \
    "ConnectionProperties":{ \
      "KAFKA_BOOTSTRAP_SERVERS":"<Kafka-broker-server-url>:<SSL-Port>", \
      "KAFKA_SSL_ENABLED":"true", \
      "KAFKA_CUSTOM_CERT": "s3://bucket/prefix/cert-file.pem" \
    }, \
    "PhysicalConnectionRequirements":{ \

```

```
"SubnetId":"subnet-1234", \  
  "SecurityGroupIdList":["sg-1234"], \  
  "AvailabilityZone":"us-east-1a"} \  
}' \  
--region us-east-1 \  
--endpoint https://glue.us-east-1.amazonaws.com
```

This command produces no output.

For more information, see [Defining Connections in the AWS Glue Data Catalog](#) in the *AWS Glue Developer Guide*.

- For API details, see [CreateConnection](#) in *AWS CLI Command Reference*.

create-database

The following code example shows how to use create-database.

AWS CLI

To create a database

The following create-database example creates a database in the AWS Glue Data Catalog.

```
aws glue create-database \  
  --database-input "{\"Name\":\"tempdb\"}" \  
  --profile my_profile \  
  --endpoint https://glue.us-east-1.amazonaws.com
```

This command produces no output.

For more information, see [Defining a Database in Your Data Catalog](#) in the *AWS Glue Developer Guide*.

- For API details, see [CreateDatabase](#) in *AWS CLI Command Reference*.

create-job

The following code example shows how to use create-job.

AWS CLI

To create a job to transform data

The following `create-job` example creates a streaming job that runs a script stored in S3.

```
aws glue create-job \  
  --name my-testing-job \  
  --role AWSGlueServiceRoleDefault \  
  --command '{ \  
    "Name": "gluestreaming", \  
    "ScriptLocation": "s3://DOC-EXAMPLE-BUCKET/folder/" \  
  }' \  
  --region us-east-1 \  
  --output json \  
  --default-arguments '{ \  
    "--job-language":"scala", \  
    "--class":"GlueApp" \  
  }' \  
  --profile my-profile \  
  --endpoint https://glue.us-east-1.amazonaws.com
```

Contents of `test_script.scala`:

```
import com.amazonaws.services.glue.ChoiceOption  
import com.amazonaws.services.glue.GlueContext  
import com.amazonaws.services.glue.MappingSpec  
import com.amazonaws.services.glue.ResolveSpec  
import com.amazonaws.services.glue.errors.CallSite  
import com.amazonaws.services.glue.util.GlueArgParser  
import com.amazonaws.services.glue.util.Job  
import com.amazonaws.services.glue.util.JsonOptions  
import org.apache.spark.SparkContext  
import scala.collection.JavaConverters._  
  
object GlueApp {  
  def main(sysArgs: Array[String]) {  
    val spark: SparkContext = new SparkContext()  
    val glueContext: GlueContext = new GlueContext(spark)  
    // @params: [JOB_NAME]  
    val args = GlueArgParser.getResolvedOptions(sysArgs,  
Seq("JOB_NAME").toArray)  
    Job.init(args("JOB_NAME"), glueContext, args.asJava)  
    // @type: DataSource  
    // @args: [database = "tempdb", table_name = "s3-source", transformation_ctx  
= "datasource0"]  
    // @return: datasource0
```



```

    // @inputs: []
    val datasource0 = glueContext.getCatalogSource(database = "tempdb",
tableName = "s3-source", redshiftTmpDir = "", transformationContext =
"datasource0").getDynamicFrame()
    // @type: ApplyMapping
    // @args: [mapping = [("sensorid", "int", "sensorid", "int"),
("currenttemperature", "int", "currenttemperature", "int"), ("status", "string",
"status", "string")], transformation_ctx = "applymapping1"]
    // @return: applymapping1
    // @inputs: [frame = datasource0]
    val applymapping1 = datasource0.applyMapping(mappings = Seq(("sensorid",
"int", "sensorid", "int"), ("currenttemperature", "int", "currenttemperature",
"int"), ("status", "string", "status", "string")), caseSensitive = false,
transformationContext = "applymapping1")
    // @type: SelectFields
    // @args: [paths = ["sensorid", "currenttemperature", "status"],
transformation_ctx = "selectfields2"]
    // @return: selectfields2
    // @inputs: [frame = applymapping1]
    val selectfields2 = applymapping1.selectFields(paths = Seq("sensorid",
"currenttemperature", "status"), transformationContext = "selectfields2")
    // @type: ResolveChoice
    // @args: [choice = "MATCH_CATALOG", database = "tempdb", table_name = "my-
s3-sink", transformation_ctx = "resolvechoice3"]
    // @return: resolvechoice3
    // @inputs: [frame = selectfields2]
    val resolvechoice3 = selectfields2.resolveChoice(choiceOption =
Some(ChoiceOption("MATCH_CATALOG")), database = Some("tempdb"), tableName =
Some("my-s3-sink"), transformationContext = "resolvechoice3")
    // @type: DataSink
    // @args: [database = "tempdb", table_name = "my-s3-sink",
transformation_ctx = "datasink4"]
    // @return: datasink4
    // @inputs: [frame = resolvechoice3]
    val datasink4 = glueContext.getCatalogSink(database = "tempdb",
tableName = "my-s3-sink", redshiftTmpDir = "", transformationContext =
"datasink4").writeDynamicFrame(resolvechoice3)
    Job.commit()
}
}

```

Output:

```
{
  "Name": "my-testing-job"
}
```

For more information, see [Authoring Jobs in AWS Glue](#) in the *AWS Glue Developer Guide*.

- For API details, see [CreateJob](#) in *AWS CLI Command Reference*.

create-table

The following code example shows how to use `create-table`.

AWS CLI

Example 1: To create a table for a Kinesis data stream

The following `create-table` example creates a table in the AWS Glue Data Catalog that describes a Kinesis data stream.

```
aws glue create-table \
  --database-name tempdb \
  --table-input '{"Name":"test-kinesis-input", "StorageDescriptor":{ \
    "Columns":[ \
      {"Name":"sensorid", "Type":"int"}, \
      {"Name":"currenttemperature", "Type":"int"}, \
      {"Name":"status", "Type":"string"} \
    ], \
    "Location":"my-testing-stream", \
    "Parameters":{ \
      "typeOfData":"kinesis", "streamName":"my-testing-stream", \
      "kinesisUrl":"https://kinesis.us-east-1.amazonaws.com" \
    }, \
    "SerdeInfo":{ \
      "SerializationLibrary":"org.openx.data.jsonserde.JsonSerDe"} \
  }, \
  "Parameters":{ \
    "classification":"json"} \
}' \
  --profile my-profile \
  --endpoint https://glue.us-east-1.amazonaws.com
```

This command produces no output.

For more information, see [Defining Tables in the AWS Glue Data Catalog](#) in the *AWS Glue Developer Guide*.

Example 2: To create a table for a Kafka data store

The following `create-table` example creates a table in the AWS Glue Data Catalog that describes a Kafka data store.

```
aws glue create-table \
  --database-name tempdb \
  --table-input '{"Name":"test-kafka-input", "StorageDescriptor":{ \
    "Columns":[ \
      {"Name":"sensorid", "Type":"int"}, \
      {"Name":"currenttemperature", "Type":"int"}, \
      {"Name":"status", "Type":"string"} \
    ], \
    "Location":"glue-topic", \
    "Parameters":{ \
      "typeOfData":"kafka","topicName":"glue-topic", \
      "connectionName":"my-kafka-connection" \
    }, \
    "SerdeInfo":{ \
      "SerializationLibrary":"org.apache.hadoop.hive.serde2.OpenCSVSerde"} \
  }, \
  "Parameters":{ \
    "separatorChar":"," \
  }' \
  --profile my-profile \
  --endpoint https://glue.us-east-1.amazonaws.com
```

This command produces no output.

For more information, see [Defining Tables in the AWS Glue Data Catalog](#) in the *AWS Glue Developer Guide*.

Example 3: To create a table for a AWS S3 data store

The following `create-table` example creates a table in the AWS Glue Data Catalog that describes a AWS Simple Storage Service (AWS S3) data store.

```
aws glue create-table \
```

```
--database-name tempdb \  
--table-input '{"Name":"s3-output", "StorageDescriptor":{ \  
  "Columns":[ \  
    {"Name":"s1", "Type":"string"}, \  
    {"Name":"s2", "Type":"int"}, \  
    {"Name":"s3", "Type":"string"} \  
  ], \  
  "Location":"s3://bucket-path/", \  
  "SerdeInfo":{ \  
    "SerializationLibrary":"org.openx.data.jsonserde.JsonSerDe"} \  
  }, \  
  "Parameters":{ \  
    "classification":"json"} \  
}' \  
--profile my-profile \  
--endpoint https://glue.us-east-1.amazonaws.com
```

This command produces no output.

For more information, see [Defining Tables in the AWS Glue Data Catalog](#) in the *AWS Glue Developer Guide*.

- For API details, see [CreateTable](#) in *AWS CLI Command Reference*.

delete-job

The following code example shows how to use `delete-job`.

AWS CLI

To delete a job

The following `delete-job` example deletes a job that is no longer needed.

```
aws glue delete-job \  
  --job-name my-testing-job
```

Output:

```
{  
  "JobName": "my-testing-job"  
}
```

For more information, see [Working with Jobs on the AWS Glue Console](#) in the *AWS Glue Developer Guide*.

- For API details, see [DeleteJob](#) in *AWS CLI Command Reference*.

get-databases

The following code example shows how to use `get-databases`.

AWS CLI

To list the definitions of some or all of the databases in the AWS Glue Data Catalog

The following `get-databases` example returns information about the databases in the Data Catalog.

```
aws glue get-databases
```

Output:

```
{
  "DatabaseList": [
    {
      "Name": "default",
      "Description": "Default Hive database",
      "LocationUri": "file:/spark-warehouse",
      "CreateTime": 1602084052.0,
      "CreateTableDefaultPermissions": [
        {
          "Principal": {
            "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
          },
          "Permissions": [
            "ALL"
          ]
        }
      ],
      "CatalogId": "111122223333"
    },
    {
      "Name": "flights-db",
      "CreateTime": 1587072847.0,
      "CreateTableDefaultPermissions": [
```

```
        {
          "Principal": {
            "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
          },
          "Permissions": [
            "ALL"
          ]
        }
      ],
      "CatalogId": "111122223333"
    },
    {
      "Name": "legislators",
      "CreateTime": 1601415625.0,
      "CreateTableDefaultPermissions": [
        {
          "Principal": {
            "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
          },
          "Permissions": [
            "ALL"
          ]
        }
      ],
      "CatalogId": "111122223333"
    },
    {
      "Name": "tempdb",
      "CreateTime": 1601498566.0,
      "CreateTableDefaultPermissions": [
        {
          "Principal": {
            "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
          },
          "Permissions": [
            "ALL"
          ]
        }
      ],
      "CatalogId": "111122223333"
    }
  ]
}
```

For more information, see [Defining a Database in Your Data Catalog](#) in the *AWS Glue Developer Guide*.

- For API details, see [GetDatabases](#) in *AWS CLI Command Reference*.

get-job-run

The following code example shows how to use `get-job-run`.

AWS CLI

To get information about a job run

The following `get-job-run` example retrieves information about a job run.

```
aws glue get-job-run \  
  --job-name "Combine legislators data" \  
  --run-id jr_012e176506505074d94d761755e5c62538ee1aad6f17d39f527e9140cf0c9a5e
```

Output:

```
{  
  "JobRun": {  
    "Id": "jr_012e176506505074d94d761755e5c62538ee1aad6f17d39f527e9140cf0c9a5e",  
    "Attempt": 0,  
    "JobName": "Combine legislators data",  
    "StartedOn": 1602873931.255,  
    "LastModifiedOn": 1602874075.985,  
    "CompletedOn": 1602874075.985,  
    "JobRunState": "SUCCEEDED",  
    "Arguments": {  
      "--enable-continuous-cloudwatch-log": "true",  
      "--enable-metrics": "",  
      "--enable-spark-ui": "true",  
      "--job-bookmark-option": "job-bookmark-enable",  
      "--spark-event-logs-path": "s3://aws-glue-assets-111122223333-us-east-1/  
sparkHistoryLogs/"  
    },  
    "PredecessorRuns": [],  
    "AllocatedCapacity": 10,  
    "ExecutionTime": 117,  
    "Timeout": 2880,  
    "MaxCapacity": 10.0,  
  }  
}
```

```
    "WorkerType": "G.1X",
    "NumberOfWorkers": 10,
    "LogGroupName": "/aws-glue/jobs",
    "GlueVersion": "2.0"
  }
}
```

For more information, see [Job Runs](#) in the *AWS Glue Developer Guide*.

- For API details, see [GetJobRun](#) in *AWS CLI Command Reference*.

get-job-runs

The following code example shows how to use `get-job-runs`.

AWS CLI

To get information about all job runs for a job

The following `get-job-runs` example retrieves information about job runs for a job.

```
aws glue get-job-runs \
  --job-name "my-testing-job"
```

Output:

```
{
  "JobRuns": [
    {
      "Id":
        "jr_012e176506505074d94d761755e5c62538ee1aad6f17d39f527e9140cf0c9a5e",
      "Attempt": 0,
      "JobName": "my-testing-job",
      "StartedOn": 1602873931.255,
      "LastModifiedOn": 1602874075.985,
      "CompletedOn": 1602874075.985,
      "JobRunState": "SUCCEEDED",
      "Arguments": {
        "--enable-continuous-cloudwatch-log": "true",
        "--enable-metrics": "",
        "--enable-spark-ui": "true",
        "--job-bookmark-option": "job-bookmark-enable",
      }
    }
  ]
}
```



```

        "--spark-event-logs-path": "s3://aws-glue-assets-111122223333-us-
east-1/sparkHistoryLogs/"
    },
    "PredecessorRuns": [],
    "AllocatedCapacity": 10,
    "ExecutionTime": 117,
    "Timeout": 2880,
    "MaxCapacity": 10.0,
    "WorkerType": "G.1X",
    "NumberOfWorkers": 10,
    "LogGroupName": "/aws-glue/jobs",
    "GlueVersion": "2.0"
},
{
    "Id":
"jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f_attempt_2",
    "Attempt": 2,
    "PreviousRunId":
"jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f_attempt_1",
    "JobName": "my-testing-job",
    "StartedOn": 1602811168.496,
    "LastModifiedOn": 1602811282.39,
    "CompletedOn": 1602811282.39,
    "JobRunState": "FAILED",
    "ErrorMessage": "An error occurred while calling
o122.pyWriteDynamicFrame.
        Access Denied (Service: Amazon S3; Status Code: 403; Error Code:
AccessDenied;
        Request ID: 021AAB703DB20A2D;
        S3 Extended Request ID: teZk24Y09TkXzBvMPG502L5VJBhe9DJuWA9/
TXtuG0qfByajkFL/Tlqt5JBGdEGpigAqzdMDM/U=)",
    "PredecessorRuns": [],
    "AllocatedCapacity": 10,
    "ExecutionTime": 110,
    "Timeout": 2880,
    "MaxCapacity": 10.0,
    "WorkerType": "G.1X",
    "NumberOfWorkers": 10,
    "LogGroupName": "/aws-glue/jobs",
    "GlueVersion": "2.0"
},
{
    "Id":
"jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f_attempt_1",

```

```

        "Attempt": 1,
        "PreviousRunId":
"jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f",
        "JobName": "my-testing-job",
        "StartedOn": 1602811020.518,
        "LastModifiedOn": 1602811138.364,
        "CompletedOn": 1602811138.364,
        "JobRunState": "FAILED",
        "ErrorMessage": "An error occurred while calling
o122.pyWriteDynamicFrame.
        Access Denied (Service: Amazon S3; Status Code: 403; Error Code:
AccessDenied;
        Request ID: 2671D37856AE7ABB;
        S3 Extended Request ID: RLJCJw20brV
+PpC6Gp0RahyF2fp9f1B5SSb2bTGPNUSPVizLXR11PN3QZ1db+v1o9qRVktNYbW8=)",
        "PredecessorRuns": [],
        "AllocatedCapacity": 10,
        "ExecutionTime": 113,
        "Timeout": 2880,
        "MaxCapacity": 10.0,
        "WorkerType": "G.1X",
        "NumberOfWorkers": 10,
        "LogGroupName": "/aws-glue/jobs",
        "GlueVersion": "2.0"
    }
]
}

```

For more information, see [Job Runs](#) in the *AWS Glue Developer Guide*.

- For API details, see [GetJobRuns](#) in *AWS CLI Command Reference*.

get-job

The following code example shows how to use `get-job`.

AWS CLI

To retrieve information about a job

The following `get-job` example retrieves information about a job.

```
aws glue get-job \
```

```
--job-name my-testing-job
```

Output:

```
{
  "Job": {
    "Name": "my-testing-job",
    "Role": "Glue_DefaultRole",
    "CreatedOn": 1602805698.167,
    "LastModifiedOn": 1602805698.167,
    "ExecutionProperty": {
      "MaxConcurrentRuns": 1
    },
    "Command": {
      "Name": "gluestreaming",
      "ScriptLocation": "s3://janetst-bucket-01/Scripts/test_script.scala",
      "PythonVersion": "2"
    },
    "DefaultArguments": {
      "--class": "GlueApp",
      "--job-language": "scala"
    },
    "MaxRetries": 0,
    "AllocatedCapacity": 10,
    "MaxCapacity": 10.0,
    "GlueVersion": "1.0"
  }
}
```

For more information, see [Jobs](#) in the *AWS Glue Developer Guide*.

- For API details, see [GetJob](#) in *AWS CLI Command Reference*.

get-plan

The following code example shows how to use `get-plan`.

AWS CLI

To get the generated code for mapping data from source tables to target tables

The following `get-plan` retrieves the generated code for mapping columns from the data source to the data target.

```

aws glue get-plan --mapping '[ \
  { \
    "SourcePath":"sensorid", \
    "SourceTable":"anything", \
    "SourceType":"int", \
    "TargetPath":"sensorid", \
    "TargetTable":"anything", \
    "TargetType":"int" \
  }, \
  { \
    "SourcePath":"currenttemperature", \
    "SourceTable":"anything", \
    "SourceType":"int", \
    "TargetPath":"currenttemperature", \
    "TargetTable":"anything", \
    "TargetType":"int" \
  }, \
  { \
    "SourcePath":"status", \
    "SourceTable":"anything", \
    "SourceType":"string", \
    "TargetPath":"status", \
    "TargetTable":"anything", \
    "TargetType":"string" \
  }]' \
--source '{ \
  "DatabaseName":"tempdb", \
  "TableName":"s3-source" \
}' \
--sinks '[ \
  { \
    "DatabaseName":"tempdb", \
    "TableName":"my-s3-sink" \
  }]' \
--language "scala"
--endpoint https://glue.us-east-1.amazonaws.com
--output "text"

```

Output:

```

import com.amazonaws.services.glue.ChoiceOption
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec

```

```
import com.amazonaws.services.glue.ResolveSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._

object GlueApp {
  def main(sysArgs: Array[String]) {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    // @params: [JOB_NAME]
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)
    // @type: DataSource
    // @args: [database = "tempdb", table_name = "s3-source", transformation_ctx =
"datasource0"]
    // @return: datasource0
    // @inputs: []
    val datasource0 = glueContext.getCatalogSource(database = "tempdb",
tableName = "s3-source", redshiftTmpDir = "", transformationContext =
"datasource0").getDynamicFrame()
    // @type: ApplyMapping
    // @args: [mapping = [("sensorid", "int", "sensorid", "int"),
("currenttemperature", "int", "currenttemperature", "int"), ("status", "string",
"status", "string")], transformation_ctx = "applymapping1"]
    // @return: applymapping1
    // @inputs: [frame = datasource0]
    val applymapping1 = datasource0.applyMapping(mappings = Seq(("sensorid",
"int", "sensorid", "int"), ("currenttemperature", "int", "currenttemperature",
"int"), ("status", "string", "status", "string")), caseSensitive = false,
transformationContext = "applymapping1")
    // @type: SelectFields
    // @args: [paths = ["sensorid", "currenttemperature", "status"],
transformation_ctx = "selectfields2"]
    // @return: selectfields2
    // @inputs: [frame = applymapping1]
    val selectfields2 = applymapping1.selectFields(paths = Seq("sensorid",
"currenttemperature", "status"), transformationContext = "selectfields2")
    // @type: ResolveChoice
    // @args: [choice = "MATCH_CATALOG", database = "tempdb", table_name = "my-s3-
sink", transformation_ctx = "resolvechoice3"]
    // @return: resolvechoice3
```

```

// @inputs: [frame = selectfields2]
val resolvechoice3 = selectfields2.resolveChoice(choiceOption =
Some(ChoiceOption("MATCH_CATALOG")), database = Some("tempdb"), tableName =
Some("my-s3-sink"), transformationContext = "resolvechoice3")
// @type: DataSink
// @args: [database = "tempdb", table_name = "my-s3-sink", transformation_ctx =
"datasink4"]
// @return: datasink4
// @inputs: [frame = resolvechoice3]
val datasink4 = glueContext.getCatalogSink(database = "tempdb",
tableName = "my-s3-sink", redshiftTmpDir = "", transformationContext =
"datasink4").writeDynamicFrame(resolvechoice3)
Job.commit()
}
}

```

For more information, see [Editing Scripts in AWS Glue](#) in the *AWS Glue Developer Guide*.

- For API details, see [GetPlan](#) in *AWS CLI Command Reference*.

get-tables

The following code example shows how to use `get-tables`.

AWS CLI

To list the definitions of some or all of the tables in the specified database

The following `get-tables` example returns information about the tables in the specified database.

```
aws glue get-tables --database-name 'tempdb'
```

Output:

```

{
  "TableList": [
    {
      "Name": "my-s3-sink",
      "DatabaseName": "tempdb",
      "CreateTime": 1602730539.0,
      "UpdateTime": 1602730539.0,
      "Retention": 0,
    }
  ]
}

```

```
"StorageDescriptor": {
  "Columns": [
    {
      "Name": "sensorid",
      "Type": "int"
    },
    {
      "Name": "currenttemperature",
      "Type": "int"
    },
    {
      "Name": "status",
      "Type": "string"
    }
  ],
  "Location": "s3://janetst-bucket-01/test-s3-output/",
  "Compressed": false,
  "NumberOfBuckets": 0,
  "SerdeInfo": {
    "SerializationLibrary": "org.openx.data.jsonserde.JsonSerDe"
  },
  "SortColumns": [],
  "StoredAsSubDirectories": false
},
"Parameters": {
  "classification": "json"
},
"CreatedBy": "arn:aws:iam::007436865787:user/JRSTERN",
"IsRegisteredWithLakeFormation": false,
"CatalogId": "007436865787"
},
{
  "Name": "s3-source",
  "DatabaseName": "tempdb",
  "CreateTime": 1602730658.0,
  "UpdateTime": 1602730658.0,
  "Retention": 0,
  "StorageDescriptor": {
    "Columns": [
      {
        "Name": "sensorid",
        "Type": "int"
      },
      {

```

```
        "Name": "currenttemperature",
        "Type": "int"
    },
    {
        "Name": "status",
        "Type": "string"
    }
],
"Location": "s3://janetst-bucket-01/",
"Compressed": false,
"NumberOfBuckets": 0,
"SortColumns": [],
"StoredAsSubDirectories": false
},
"Parameters": {
    "classification": "json"
},
"CreatedBy": "arn:aws:iam::007436865787:user/JRSTERN",
"IsRegisteredWithLakeFormation": false,
"CatalogId": "007436865787"
},
{
    "Name": "test-kinesis-input",
    "DatabaseName": "tempdb",
    "CreateTime": 1601507001.0,
    "UpdateTime": 1601507001.0,
    "Retention": 0,
    "StorageDescriptor": {
        "Columns": [
            {
                "Name": "sensorid",
                "Type": "int"
            },
            {
                "Name": "currenttemperature",
                "Type": "int"
            },
            {
                "Name": "status",
                "Type": "string"
            }
        ]
    },
    "Location": "my-testing-stream",
    "Compressed": false,
```



```
        "NumberOfBuckets": 0,
        "SerdeInfo": {
            "SerializationLibrary": "org.openx.data.jsonserde.JsonSerDe"
        },
        "SortColumns": [],
        "Parameters": {
            "kinesisUrl": "https://kinesis.us-east-1.amazonaws.com",
            "streamName": "my-testing-stream",
            "typeOfData": "kinesis"
        },
        "StoredAsSubDirectories": false
    },
    "Parameters": {
        "classification": "json"
    },
    "CreatedBy": "arn:aws:iam::007436865787:user/JRSTERN",
    "IsRegisteredWithLakeFormation": false,
    "CatalogId": "007436865787"
}
]
}
```

For more information, see [Defining Tables in the AWS Glue Data Catalog](#) in the *AWS Glue Developer Guide*.

- For API details, see [GetTables](#) in *AWS CLI Command Reference*.

start-crawler

The following code example shows how to use `start-crawler`.

AWS CLI

To start a crawler

The following `start-crawler` example starts a crawler.

```
aws glue start-crawler --name my-crawler
```

Output:

```
None
```

For more information, see [Defining Crawlers](#) in the *AWS Glue Developer Guide*.

- For API details, see [StartCrawler](#) in *AWS CLI Command Reference*.

start-job-run

The following code example shows how to use `start-job-run`.

AWS CLI

To start running a job

The following `start-job-run` example starts a job.

```
aws glue start-job-run \  
  --job-name my-job
```

Output:

```
{  
  "JobRunId":  
  "jr_22208b1f44eb5376a60569d4b21dd20fcb8621e1a366b4e7b2494af764b82ded"  
}
```

For more information, see [Authoring Jobs](#) in the *AWS Glue Developer Guide*.

- For API details, see [StartJobRun](#) in *AWS CLI Command Reference*.

GuardDuty examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with GuardDuty.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

accept-invitation

The following code example shows how to use `accept-invitation`.

AWS CLI

To accept an invitation to become a GuardDuty member account in the current region

The following `accept-invitation` example shows how to accept an invitation to become a GuardDuty member account in the current region.

```
aws guardduty accept-invitation \  
  --detector-id 12abc34d567e8fa901bc2d34eexample \  
  --master-id 123456789111 \  
  --invitation-id d6b94fb03a66ff665f7db8764example
```

This command produces no output.

For more information, see [Managing GuardDuty Accounts by Invitation](#) in the GuardDuty User Guide.

- For API details, see [AcceptInvitation](#) in *AWS CLI Command Reference*.

archive-findings

The following code example shows how to use `archive-findings`.

AWS CLI

To archive findings in the current region

This example shows how to archive findings in the current region.

```
aws guardduty archive-findings \  
  --detector-id 12abc34d567e8fa901bc2d34eexample \  
  --finding-ids d6b94fb03a66ff665f7db8764example 3eb970e0de00c16ec14e6910fexample
```

This command produces no output.

For more information, see [Managing GuardDuty Accounts by Invitation](#) in the *GuardDuty User Guide*.

- For API details, see [ArchiveFindings](#) in *AWS CLI Command Reference*.

create-detector

The following code example shows how to use `create-detector`.

AWS CLI

To enable GuardDuty in the current region

This example shows how to create a new detector, which enables GuardDuty, in the current region.:

```
aws guardduty create-detector \  
  --enable
```

Output:

```
{  
  "DetectorId": "b6b992d6d2f48e64bc59180bfexample"  
}
```

For more information, see [Enable Amazon GuardDuty](#) in the *GuardDuty User Guide*.

- For API details, see [CreateDetector](#) in *AWS CLI Command Reference*.

create-filter

The following code example shows how to use `create-filter`.

AWS CLI

To create a new filter for the current region

This example creates a filter that matches all portscan findings for instance created from a specific image.:

```
aws guardduty create-filter \  
  --enable
```

```
--detector-id b6b992d6d2f48e64bc59180bfexample \  
--action ARCHIVE \  
--name myFilter \  
--finding-criteria '{"Criterion": {"type": {"Eq": ["Recon:EC2/  
Portscan"]},"resource.instanceDetails.imageId": {"Eq": ["ami-0a7a207083example"]}}}'
```

Output:

```
{  
  "Name": "myFilter"  
}
```

For more information, see [Filtering findings](#) in the *GuardDuty User Guide*.

- For API details, see [CreateFilter](#) in *AWS CLI Command Reference*.

create-ip-set

The following code example shows how to use `create-ip-set`.

AWS CLI

To create a trusted IP set

The following `create-ip-set` example creates and activates a trusted IP set in the current region.

```
aws guardduty create-ip-set \  
  --detector-id 12abc34d567e8fa901bc2d34eexample \  
  --name new-ip-set \  
  --format TXT \  
  --location s3://AWSDOC-EXAMPLE-BUCKET/customtrustlist.csv \  
  --activate
```

Output:

```
{  
  "IpSetId": "d4b94fc952d6912b8f3060768example"  
}
```

For more information, see [Working with Trusted IP Lists and Threat Lists](#) in the *GuardDuty User Guide*.

- For API details, see [CreateIpSet](#) in *AWS CLI Command Reference*.

create-members

The following code example shows how to use create-members.

AWS CLI

To associate a new member with your GuardDuty master account in the current region.

This example shows how to associate member accounts to be managed by the current account as the GuardDuty master.

```
aws guardduty create-members
  --detector-id b6b992d6d2f48e64bc59180bfexample \
  --account-details AccountId=111122223333,Email=first+member@example.com
  AccountId=111111111111 ,Email=another+member@example.com
```

Output:

```
{
  "UnprocessedAccounts": []
}
```

For more information, see [Managing multiple accounts](#) in the GuardDuty User Guide.

- For API details, see [CreateMembers](#) in *AWS CLI Command Reference*.

create-publishing-destination

The following code example shows how to use create-publishing-destination.

AWS CLI

To create a publishing destination to export GuardDuty findings in the current region to.

This example shows how to create a publishing destination for GuardDuty findings.

```
aws guardduty create-publishing-destination \
  --detector-id b6b992d6d2f48e64bc59180bfexample \
  --destination-type S3 \
```

```
--destination-properties
DestinationArn=arn:aws:s3:::yourbucket,KmsKeyArn=arn:aws:kms:us-
west-1:111122223333:key/84cee9c5-dea1-401a-ab6d-e1de7example
```

Output:

```
{
  "DestinationId": "46b99823849e1bbc242dfbe3cexample"
}
```

For more information, see [Exporting findings](#) in the *GuardDuty User Guide*.

- For API details, see [CreatePublishingDestination](#) in *AWS CLI Command Reference*.

create-sample-findings

The following code example shows how to use `create-sample-findings`.

AWS CLI

To create sample GuardDuty findings in the current region.

This example shows how to create a sample finding of the provided types.

```
aws guardduty create-sample-findings \
  --detector-id b6b992d6d2f48e64bc59180bfexample \
  --finding-types UnauthorizedAccess:EC2/TorClient UnauthorizedAccess:EC2/TorRelay
```

This command produces no output.

For more information, see [Sample findings](#) in the *GuardDuty User Guide*.

- For API details, see [CreateSampleFindings](#) in *AWS CLI Command Reference*.

create-threat-intel-set

The following code example shows how to use `create-threat-intel-set`.

AWS CLI

To create a new threat intel set in the current region.

This example shows how to upload a threat intel set to GuardDuty and activate it immediately.

```
aws guardduty create-threat-intel-set \  
  --detector-id b6b992d6d2f48e64bc59180bfexample \  
  --name myThreatSet \  
  --format TXT \  
  --location s3://EXAMPLEBUCKET/threatlist.csv \  
  --activate
```

Output:

```
{  
  "ThreatIntelSetId": "20b9a4691aeb33506b808878cexample"  
}
```

For more information, see [Trusted IP and threat lists](#) in the *GuardDuty User Guide*.

- For API details, see [CreateThreatIntelSet](#) in *AWS CLI Command Reference*.

decline-invitations

The following code example shows how to use `decline-invitations`.

AWS CLI

To decline an invitation to have Guardduty managed by another account in the current region.

This example shows how to decline a membership invitation.

```
aws guardduty decline-invitations \  
  --account-ids 111122223333
```

Output:

```
{  
  "UnprocessedAccounts": []  
}
```

For more information, see [Managing GuardDuty accounts by invitation](#) in the *GuardDuty User Guide*.

- For API details, see [DeclineInvitations](#) in *AWS CLI Command Reference*.

delete-detector

The following code example shows how to use `delete-detector`.

AWS CLI

To delete a detector, and disable GuardDuty, in the current region.

This example shows how to delete a detector, if successful, this will disable GuardDuty in the region associated with that detector.

```
aws guardduty delete-detector \  
  --detector-id b6b992d6d2f48e64bc59180bfexample
```

This command produces no output.

For more information, see [Suspending or disabling GuardDuty](#) in the *GuardDuty User Guide*.

- For API details, see [DeleteDetector](#) in *AWS CLI Command Reference*.

delete-filter

The following code example shows how to use `delete-filter`.

AWS CLI

To delete an existing filter in the current region

This example shows how to create delete a filter.

```
aws guardduty delete-filter \  
  --detector-id b6b992d6d2f48e64bc59180bfexample \  
  --filter-name byebyeFilter
```

This command produces no output.

For more information, see [Filtering findings](#) in the *GuardDuty User Guide*.

- For API details, see [DeleteFilter](#) in *AWS CLI Command Reference*.

disable-organization-admin-account

The following code example shows how to use `disable-organization-admin-account`.

AWS CLI

To remove an account as the delegated administrator for GuardDuty within your organization

This example shows how to remove an account as the delegated administrator for GuardDuty.

```
aws guardduty disable-organization-admin-account \  
  --admin-account-id 111122223333
```

This command produces no output.

For more information, see [Managing accounts with AWS organizations](#) in the *GuardDuty User Guide*.

- For API details, see [DisableOrganizationAdminAccount](#) in *AWS CLI Command Reference*.

disassociate-from-master-account

The following code example shows how to use `disassociate-from-master-account`.

AWS CLI

To disassociate from your current master account in the current region

The following `disassociate-from-master-account` example disassociates your account from the current GuardDuty master account in the current AWS region.

```
aws guardduty disassociate-from-master-account \  
  --detector-id d4b040365221be2b54a6264dcexample
```

This command produces no output.

For more information, see [Understanding the Relationship between GuardDuty Master and Member Accounts](#) in the *GuardDuty User Guide*.

- For API details, see [DisassociateFromMasterAccount](#) in *AWS CLI Command Reference*.

get-detector

The following code example shows how to use `get-detector`.

AWS CLI

To retrieve details of a specific detector

The following `get-detector` example displays the configurations details of the specified detector.

```
aws guardduty get-detector \  
  --detector-id 12abc34d567e8fa901bc2d34eexample
```

Output:

```
{  
  "Status": "ENABLED",  
  "ServiceRole": "arn:aws:iam::111122223333:role/aws-service-role/  
guardduty.amazonaws.com/AWSServiceRoleForAmazonGuardDuty",  
  "Tags": {},  
  "FindingPublishingFrequency": "SIX_HOURS",  
  "UpdatedAt": "2018-11-07T03:24:22.938Z",  
  "CreatedAt": "2017-12-22T22:51:31.940Z"  
}
```

For more information, see [Concepts and Terminology](#) in the GuardDuty User Guide.

- For API details, see [GetDetector](#) in *AWS CLI Command Reference*.

get-findings

The following code example shows how to use `get-findings`.

AWS CLI

Example 1: To retrieve the details of a specific finding

The following `get-findings` example retrieves the full JSON finding details of the specified finding.

```
aws guardduty get-findings \  
  --detector-id 12abc34d567e8fa901bc2d34eexample \  
  --finding-id 1ab92989eaf0e742df4a014d5example
```

Output:

```
{
  "Findings": [
    {
      "Resource": {
        "ResourceType": "AccessKey",
        "AccessKeyDetails": {
          "UserName": "testuser",
          "UserType": "IAMUser",
          "PrincipalId": "AIDACKCEVSQ6C2EXAMPLE",
          "AccessKeyId": "ASIASZ4SI7REEEXAMPLE"
        }
      },
      "Description": "APIs commonly used to discover the users, groups,
policies and permissions in an account, was invoked by IAM principal testuser under
unusual circumstances. Such activity is not typically seen from this principal.",
      "Service": {
        "Count": 5,
        "Archived": false,
        "ServiceName": "guardduty",
        "EventFirstSeen": "2020-05-26T22:02:24Z",
        "ResourceRole": "TARGET",
        "EventLastSeen": "2020-05-26T22:33:55Z",
        "DetectorId": "d4b040365221be2b54a6264dcexample",
        "Action": {
          "ActionType": "AWS_API_CALL",
          "AwsApiCallAction": {
            "RemoteIpDetails": {
              "GeoLocation": {
                "Lat": 51.5164,
                "Lon": -0.093
              },
              "City": {
                "CityName": "London"
              },
              "IpAddressV4": "52.94.36.7",
              "Organization": {
                "Org": "Amazon.com",
                "Isp": "Amazon.com",
                "Asn": "16509",
                "AsnOrg": "AMAZON-02"
              },
              "Country": {
```

```

        "CountryName": "United Kingdom"
      }
    },
    "Api": "ListPolicyVersions",
    "ServiceName": "iam.amazonaws.com",
    "CallerType": "Remote IP"
  }
}
},
"Title": "Unusual user permission reconnaissance activity by testuser.",
"Type": "Recon:IAMUser/UserPermissions",
"Region": "us-east-1",
"Partition": "aws",
"Arn": "arn:aws:guardduty:us-east-1:111122223333:detector/
d4b040365221be2b54a6264dcexample/finding/1ab92989eaf0e742df4a014d5example",
"UpdatedAt": "2020-05-26T22:55:21.703Z",
"SchemaVersion": "2.0",
"Severity": 5,
"Id": "1ab92989eaf0e742df4a014d5example",
"CreatedAt": "2020-05-26T22:21:48.385Z",
"AccountId": "111122223333"
}
]
}

```

For more information, see [Findings](#) in the GuardDuty User Guide.

- For API details, see [GetFindings](#) in *AWS CLI Command Reference*.

get-ip-set

The following code example shows how to use `get-ip-set`.

AWS CLI

To list get details on a specified trusted IP set

The following `get-ip-set` example shows the status and details of the specified trusted IP set.

```

aws guardduty get-ip-set \
  --detector-id 12abc34d567e8fa901bc2d34eexample \
  --ip-set-id d4b94fc952d6912b8f3060768example

```

Output:

```
{
  "Status": "ACTIVE",
  "Location": "s3://AWSDOC-EXAMPLE-BUCKET.s3-us-west-2.amazonaws.com/
customlist.csv",
  "Tags": {},
  "Format": "TXT",
  "Name": "test-ip-set"
}
```

For more information, see [Working with Trusted IP Lists and Threat Lists](#) in the GuardDuty User Guide.

- For API details, see [GetIpSet](#) in *AWS CLI Command Reference*.

get-master-account

The following code example shows how to use `get-master-account`.

AWS CLI**To retrieve details about your master account in the current region**

The following `get-master-account` example displays the status and details of the master account associated with your detector in the current region.

```
aws guardduty get-master-account \
  --detector-id 12abc34d567e8fa901bc2d34eexample
```

Output:

```
{
  "Master": {
    "InvitationId": "04b94d9704854a73f94e061e8example",
    "InvitedAt": "2020-06-09T22:23:04.970Z",
    "RelationshipStatus": "Enabled",
    "AccountId": "123456789111"
  }
}
```

For more information, see [Understanding the Relationship between GuardDuty Master and Member Accounts](#) in the GuardDuty User Guide.

- For API details, see [GetMasterAccount](#) in *AWS CLI Command Reference*.

list-detectors

The following code example shows how to use `list-detectors`.

AWS CLI

To list the available detectors in the current region

The following `list-detectors` example lists the available detectors in your current AWS region.

```
aws guardduty list-detectors
```

Output:

```
{
  "DetectorIds": [
    "12abc34d567e8fa901bc2d34eexample"
  ]
}
```

For more information, see [Concepts and Terminology](#) in the GuardDuty User Guide.

- For API details, see [ListDetectors](#) in *AWS CLI Command Reference*.

list-findings

The following code example shows how to use `list-findings`.

AWS CLI

Example 1: To list all findings for the current region

The following `list-findings` example displays a list of all findingIds for the current region sorted by severity from highest to lowest.

```
aws guardduty list-findings \  
  --detector-id 12abc34d567e8fa901bc2d34eexample \  
  --sort-criteria '{"AttributeName": "severity", "OrderBy": "DESC"}'
```

Output:

```
{  
  "FindingIds": [  
    "04b8ab50fd29c64fc771b232dexample",  
    "5ab8ab50fd21373735c826d3aexample",  
    "90b93de7aba69107f05bbe60bexample",  
    ...  
  ]  
}
```

For more information, see [Findings](#) in the GuardDuty User Guide.

Example 2: To list findings for the current region matching a specific finding criteria

The following `list-findings` example displays a list of all `findingIds` that match a specified finding type.

```
aws guardduty list-findings \  
  --detector-id 12abc34d567e8fa901bc2d34eexample \  
  --finding-criteria '{"Criterion":{"type": {"Eq":["UnauthorizedAccess:EC2/  
SSHBruteForce"]}}}'
```

Output:

```
{  
  "FindingIds": [  
    "90b93de7aba69107f05bbe60bexample",  
    "6eb9430d7023d30774d6f05e3example",  
    "2eb91a2d060ac9a21963a5848example",  
    "44b8ab50fd2b0039a9e48f570example",  
    "9eb8ab4cd2b7e5b66ba4f5e96example",  
    "e0b8ab3a38e9b0312cc390ceeexample"  
  ]  
}
```

For more information, see [Findings](#) in the GuardDuty User Guide.

Example 3: To list findings for the current region matching a specific set of finding criteria defined within a JSON file

The following `list-findings` example displays a list of all findingIds that are not archived, and involve the IAM user named "testuser", as specified in a JSON file.

```
aws guardduty list-findings \  
  --detector-id 12abc34d567e8fa901bc2d34eexample \  
  --finding-criteria file://myfile.json
```

Contents of `myfile.json`:

```
{  
  "Criterion": {  
    "resource.accessKeyDetails.userName": {  
      "Eq": [  
        "testuser"  
      ]  
    },  
    "service.archived": {  
      "Eq": [  
        "false"  
      ]  
    }  
  }  
}
```

Output:

```
{  
  "FindingIds": [  
    "1ab92989eaf0e742df4a014d5example"  
  ]  
}
```

For more information, see [Findings](#) in the GuardDuty User Guide.

- For API details, see [ListFindings](#) in *AWS CLI Command Reference*.

list-invitations

The following code example shows how to use `list-invitations`.

AWS CLI

To list details on your invitations to become a member account in the current region

The following `list-invitations` example lists details and statuses on your invitations to become a GuardDuty member account in the current region.

```
aws guardduty list-invitations
```

Output:

```
{
  "Invitations": [
    {
      "InvitationId": "d6b94fb03a66ff665f7db8764example",
      "InvitedAt": "2020-06-10T17:56:38.221Z",
      "RelationshipStatus": "Invited",
      "AccountId": "123456789111"
    }
  ]
}
```

For more information, see [Managing GuardDuty Accounts by Invitation](#) in the GuardDuty User Guide.

- For API details, see [ListInvitations](#) in *AWS CLI Command Reference*.

list-ip-sets

The following code example shows how to use `list-ip-sets`.

AWS CLI

To list trusted IP sets in the current region

The following `list-ip-sets` example lists the trusted IP sets in your current AWS region.

```
aws guardduty list-ip-sets \
  --detector-id 12abc34d567e8fa901bc2d34eexample
```

Output:

```
{
  "IpSetIds": [
    "d4b94fc952d6912b8f3060768example"
  ]
}
```

For more information, see [Working with Trusted IP Lists and Threat Lists](#) in the GuardDuty User Guide.

- For API details, see [ListIpSets](#) in *AWS CLI Command Reference*.

list-members

The following code example shows how to use `list-members`.

AWS CLI

To list all members in the current region

The following `list-members` example lists all member accounts and their details for the current region.

```
aws guardduty list-members \
  --detector-id 12abc34d567e8fa901bc2d34eexample
```

Output:

```
{
  "Members": [
    {
      "RelationshipStatus": "Enabled",
      "InvitedAt": "2020-06-09T22:49:00.910Z",
      "MasterId": "123456789111",
      "DetectorId": "7ab8b2f61b256c87f793f6a86example",
      "UpdatedAt": "2020-06-09T23:08:22.512Z",
      "Email": "your+member@example.com",
      "AccountId": "123456789222"
    }
  ]
}
```

For more information, see [Understanding the Relationship between GuardDuty Master and Member Accounts](#) in the GuardDuty User Guide.

- For API details, see [ListMembers](#) in *AWS CLI Command Reference*.

update-ip-set

The following code example shows how to use `update-ip-set`.

AWS CLI

To update a trusted IP set

The following `update-ip-set` example shows how to update the details of a trusted IP set.

```
aws guardduty update-ip-set \  
  --detector-id 12abc34d567e8fa901bc2d34eexample \  
  --ip-set-id d4b94fc952d6912b8f3060768example \  
  --location https://AWSDOC-EXAMPLE-BUCKET.s3-us-west-2.amazonaws.com/  
  customtrustlist2.csv
```

This command produces no output.

For more information, see [Working with Trusted IP Lists and Threat Lists](#) in the GuardDuty User Guide.

- For API details, see [UpdateIpSet](#) in *AWS CLI Command Reference*.

AWS Health examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Health.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

describe-affected-entities

The following code example shows how to use `describe-affected-entities`.

AWS CLI

To list the entities that are affected by a specified AWS Health event

The following `describe-affected-entities` example lists the entities that are affected by the specified AWS Health event. This event is a billing notification for the AWS account.

```
aws health describe-affected-entities \
  --filter "eventArns=arn:aws:health:global::event/BILLING/
  AWS_BILLING_NOTIFICATION/AWS_BILLING_NOTIFICATION_6ce1d874-e995-40e2-99cd-
  EXAMPLE11145" \
  --region us-east-1
```

Output:

```
{
  "entities": [
    {
      "entityArn": "arn:aws:health:global:123456789012:entity/
  EXAMPLEimSMoULmWHpb",
      "eventArn": "arn:aws:health:global::event/BILLING/
  AWS_BILLING_NOTIFICATION/AWS_BILLING_NOTIFICATION_6ce1d874-e995-40e2-99cd-
  EXAMPLE11145",
      "entityValue": "AWS_ACCOUNT",
      "awsAccountId": "123456789012",
      "lastUpdatedTime": 1588356454.08
    }
  ]
}
```

For more information, see [Event log](#) in the *AWS Health User Guide*.

- For API details, see [DescribeAffectedEntities](#) in *AWS CLI Command Reference*.

describe-event-details

The following code example shows how to use describe-event-details.

AWS CLI

To list information about an AWS Health event

The following describe-event-details example lists information about the specified AWS Health event.

```
aws health describe-event-details \
  --event-arns "arn:aws:health:us-east-1::event/EC2/AWS_EC2_OPERATIONAL_ISSUE/
AWS_EC2_OPERATIONAL_ISSUE_VKTXI_EXAMPLE111" \
  --region us-east-1
```

Output:

```
{
  "successfulSet": [
    {
      "event": {
        "arn": "arn:aws:health:us-east-1::event/EC2/
AWS_EC2_OPERATIONAL_ISSUE/AWS_EC2_OPERATIONAL_ISSUE_VKTXI_EXAMPLE111",
        "service": "EC2",
        "eventTypeCode": "AWS_EC2_OPERATIONAL_ISSUE",
        "eventTypeCategory": "issue",
        "region": "us-east-1",
        "startTime": 1587462325.096,
        "endTime": 1587464204.774,
        "lastUpdatedTime": 1587464204.865,
        "statusCode": "closed"
      },
      "eventDescription": {
        "latestDescription": "[RESOLVED] Increased API Error Rates and
Latencies\n\n[02:45 AM PDT] We are investigating increased API error rates and
latencies in the US-EAST-1 Region.\n\n[03:16 AM PDT] Between 2:10 AM and 2:59 AM
PDT we experienced increased API error rates and latencies in the US-EAST-1 Region.
The issue has been resolved and the service is operating normally."
      }
    }
  ],
  "failedSet": []
}
```

```
}
```

For more information, see [Event details pane](#) in the *AWS Health User Guide*.

- For API details, see [DescribeEventDetails](#) in *AWS CLI Command Reference*.

describe-events

The following code example shows how to use `describe-events`.

AWS CLI

Example 1: To list AWS Health events

The following `describe-events` example lists recent AWS Health events.

```
aws health describe-events \  
  --region us-east-1
```

Output:

```
{  
  "events": [  
    {  
      "arn": "arn:aws:health:us-west-1::event/ECS/AWS_ECS_OPERATIONAL_ISSUE/  
AWS_ECS_OPERATIONAL_ISSUE_KWQPY_EXAMPLE111",  
      "service": "ECS",  
      "eventTypeCode": "AWS_ECS_OPERATIONAL_ISSUE",  
      "eventTypeCategory": "issue",  
      "region": "us-west-1",  
      "startTime": 1589077890.53,  
      "endTime": 1589086345.597,  
      "lastUpdatedTime": 1589086345.905,  
      "statusCode": "closed",  
      "eventScopeCode": "PUBLIC"  
    },  
    {  
      "arn": "arn:aws:health:global::event/BILLING/AWS_BILLING_NOTIFICATION/  
AWS_BILLING_NOTIFICATION_6ce1d874-e995-40e2-99cd-EXAMPLE1118b",  
      "service": "BILLING",  
      "eventTypeCode": "AWS_BILLING_NOTIFICATION",  
      "eventTypeCategory": "accountNotification",  
      "region": "global",
```

```
    "startTime": 1588356000.0,
    "lastUpdatedTime": 1588356524.358,
    "statusCode": "open",
    "eventScopeCode": "ACCOUNT_SPECIFIC"
  },
  {
    "arn": "arn:aws:health:us-west-2::event/
    CLOUDFORMATION/AWS_CLOUDFORMATION_OPERATIONAL_ISSUE/
    AWS_CLOUDFORMATION_OPERATIONAL_ISSUE_OHTWY_EXAMPLE111",
    "service": "CLOUDFORMATION",
    "eventTypeCode": "AWS_CLOUDFORMATION_OPERATIONAL_ISSUE",
    "eventTypeCategory": "issue",
    "region": "us-west-2",
    "startTime": 1588279630.761,
    "endTime": 1588284650.0,
    "lastUpdatedTime": 1588284691.941,
    "statusCode": "closed",
    "eventScopeCode": "PUBLIC"
  },
  {
    "arn": "arn:aws:health:ap-northeast-1::event/LAMBDA/
    AWS_LAMBDA_OPERATIONAL_ISSUE/AWS_LAMBDA_OPERATIONAL_ISSUE_JZDND_EXAMPLE111",
    "service": "LAMBDA",
    "eventTypeCode": "AWS_LAMBDA_OPERATIONAL_ISSUE",
    "eventTypeCategory": "issue",
    "region": "ap-northeast-1",
    "startTime": 1587379534.08,
    "endTime": 1587391771.0,
    "lastUpdatedTime": 1587395689.316,
    "statusCode": "closed",
    "eventScopeCode": "PUBLIC"
  },
  {
    "arn": "arn:aws:health:us-east-1::event/EC2/AWS_EC2_OPERATIONAL_ISSUE/
    AWS_EC2_OPERATIONAL_ISSUE_COBJX_EXAMPLE111",
    "service": "EC2",
    "eventTypeCode": "AWS_EC2_OPERATIONAL_ISSUE",
    "eventTypeCategory": "issue",
    "region": "us-east-1",
    "startTime": 1586473044.284,
    "endTime": 1586479706.091,
    "lastUpdatedTime": 1586479706.153,
    "statusCode": "closed",
    "eventScopeCode": "PUBLIC"
  }
}
```



```
    },
    {
      "arn": "arn:aws:health:global::event/SECURITY/AWS_SECURITY_NOTIFICATION/
AWS_SECURITY_NOTIFICATION_42007387-8129-42da-8c88-EXAMPLE11139",
      "service": "SECURITY",
      "eventTypeCode": "AWS_SECURITY_NOTIFICATION",
      "eventTypeCategory": "accountNotification",
      "region": "global",
      "startTime": 1585674000.0,
      "lastUpdatedTime": 1585674004.132,
      "statusCode": "open",
      "eventScopeCode": "PUBLIC"
    },
    {
      "arn": "arn:aws:health:global::event/CLOUDFRONT/
AWS_CLOUDFRONT_OPERATIONAL_ISSUE/AWS_CLOUDFRONT_OPERATIONAL_ISSUE_FRQXG_EXAMPLE111",
      "service": "CLOUDFRONT",
      "eventTypeCode": "AWS_CLOUDFRONT_OPERATIONAL_ISSUE",
      "eventTypeCategory": "issue",
      "region": "global",
      "startTime": 1585610898.589,
      "endTime": 1585617671.0,
      "lastUpdatedTime": 1585620638.869,
      "statusCode": "closed",
      "eventScopeCode": "PUBLIC"
    },
    {
      "arn": "arn:aws:health:us-east-1::event/SES/AWS_SES_OPERATIONAL_ISSUE/
AWS_SES_OPERATIONAL_ISSUE_URNDF_EXAMPLE111",
      "service": "SES",
      "eventTypeCode": "AWS_SES_OPERATIONAL_ISSUE",
      "eventTypeCategory": "issue",
      "region": "us-east-1",
      "startTime": 1585342008.46,
      "endTime": 1585344017.0,
      "lastUpdatedTime": 1585344355.989,
      "statusCode": "closed",
      "eventScopeCode": "PUBLIC"
    },
    {
      "arn": "arn:aws:health:global::event/IAM/
AWS_IAM_OPERATIONAL_NOTIFICATION/
AWS_IAM_OPERATIONAL_NOTIFICATION_b6771c34-6ecd-4aea-9d3e-EXAMPLE1117e",
      "service": "IAM",
```

```

        "eventTypeCode": "AWS_IAM_OPERATIONAL_NOTIFICATION",
        "eventTypeCategory": "accountNotification",
        "region": "global",
        "startTime": 1584978300.0,
        "lastUpdatedTime": 1584978553.572,
        "statusCode": "open",
        "eventScopeCode": "ACCOUNT_SPECIFIC"
    },
    {
        "arn": "arn:aws:health:ap-southeast-2::event/EC2/
AWS_EC2_OPERATIONAL_ISSUE/AWS_EC2_OPERATIONAL_ISSUE_HNGHE_EXAMPLE111",
        "service": "EC2",
        "eventTypeCode": "AWS_EC2_OPERATIONAL_ISSUE",
        "eventTypeCategory": "issue",
        "region": "ap-southeast-2",
        "startTime": 1583881487.483,
        "endTime": 1583885056.785,
        "lastUpdatedTime": 1583885057.052,
        "statusCode": "closed",
        "eventScopeCode": "PUBLIC"
    }
]
}

```

For more information, see [Getting started with the AWS Personal Health Dashboard](#) in the *AWS Health User Guide*.

Example 2: To list AWS Health events by service and event status code

The following `describe-events` example lists AWS Health events for Amazon Elastic Compute Cloud (Amazon EC2) where the event status is closed.

```

aws health describe-events \
  --filter "services=EC2,eventStatusCodes=closed"

```

Output:

```

{
  "events": [
    {
      "arn": "arn:aws:health:us-east-1::event/EC2/AWS_EC2_OPERATIONAL_ISSUE/
AWS_EC2_OPERATIONAL_ISSUE_VKTXI_EXAMPLE111",

```

```
    "service": "EC2",
    "eventTypeCode": "AWS_EC2_OPERATIONAL_ISSUE",
    "eventTypeCategory": "issue",
    "region": "us-east-1",
    "startTime": 1587462325.096,
    "endTime": 1587464204.774,
    "lastUpdatedTime": 1587464204.865,
    "statusCode": "closed",
    "eventScopeCode": "PUBLIC"
  },
  {
    "arn": "arn:aws:health:us-east-1::event/EC2/AWS_EC2_OPERATIONAL_ISSUE/
AWS_EC2_OPERATIONAL_ISSUE_COBXJ_EXAMPLE111",
    "service": "EC2",
    "eventTypeCode": "AWS_EC2_OPERATIONAL_ISSUE",
    "eventTypeCategory": "issue",
    "region": "us-east-1",
    "startTime": 1586473044.284,
    "endTime": 1586479706.091,
    "lastUpdatedTime": 1586479706.153,
    "statusCode": "closed",
    "eventScopeCode": "PUBLIC"
  },
  {
    "arn": "arn:aws:health:ap-southeast-2::event/EC2/
AWS_EC2_OPERATIONAL_ISSUE/AWS_EC2_OPERATIONAL_ISSUE_HNGHE_EXAMPLE111",
    "service": "EC2",
    "eventTypeCode": "AWS_EC2_OPERATIONAL_ISSUE",
    "eventTypeCategory": "issue",
    "region": "ap-southeast-2",
    "startTime": 1583881487.483,
    "endTime": 1583885056.785,
    "lastUpdatedTime": 1583885057.052,
    "statusCode": "closed",
    "eventScopeCode": "PUBLIC"
  }
]
}
```

For more information, see [Getting started with the AWS Personal Health Dashboard](#) in the *AWS Health User Guide*.

- For API details, see [DescribeEvents](#) in *AWS CLI Command Reference*.

HealthImaging examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with HealthImaging.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

copy-image-set

The following code example shows how to use `copy-image-set`.

AWS CLI

Example 1: To copy an image set without a destination.

The following `copy-image-set` code example makes a duplicate copy of an image set without a destination.

```
aws medical-imaging copy-image-set \  
  --datastore-id 12345678901234567890123456789012 \  
  --source-image-set-id ea92b0d8838c72a3f25d00d13616f87e \  
  --copy-image-set-information '{"sourceImageSet": {"latestVersionId": "1" } }'
```

Output:

```
{  
  "destinationImageSetProperties": {
```

```

    "latestVersionId": "2",
    "imageSetWorkflowStatus": "COPYING",
    "updatedAt": 1680042357.432,
    "imageSetId": "b9a06fef182a5f992842f77f8e0868e5",
    "imageSetState": "LOCKED",
    "createdAt": 1680042357.432
  },
  "sourceImageSetProperties": {
    "latestVersionId": "1",
    "imageSetWorkflowStatus": "COPYING_WITH_READ_ONLY_ACCESS",
    "updatedAt": 1680042357.432,
    "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",
    "imageSetState": "LOCKED",
    "createdAt": 1680027126.436
  },
  "datastoreId": "12345678901234567890123456789012"
}

```

Example 2: To copy an image set with a destination.

The following `copy-image-set` code example makes a duplicate copy of an image set with a destination.

```

aws medical-imaging copy-image-set \
  --datastore-id 12345678901234567890123456789012 \
  --source-image-set-id ea92b0d8838c72a3f25d00d13616f87e \
  --copy-image-set-information '{"sourceImageSet": {"latestVersionId": "1" },
"destinationImageSet": { "imageSetId": "b9a06fef182a5f992842f77f8e0868e5",
"latestVersionId": "1"} }'

```

Output:

```

{
  "destinationImageSetProperties": {
    "latestVersionId": "2",
    "imageSetWorkflowStatus": "COPYING",
    "updatedAt": 1680042505.135,
    "imageSetId": "b9a06fef182a5f992842f77f8e0868e5",
    "imageSetState": "LOCKED",
    "createdAt": 1680042357.432
  },
  "sourceImageSetProperties": {
    "latestVersionId": "1",

```

```
    "imageSetWorkflowStatus": "COPYING_WITH_READ_ONLY_ACCESS",
    "updatedAt": 1680042505.135,
    "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",
    "imageSetState": "LOCKED",
    "createdAt": 1680027126.436
  },
  "datastoreId": "12345678901234567890123456789012"
}
```

For more information, see [Copying an image set](#) in the *AWS HealthImaging Developer Guide*.

- For API details, see [CopyImageSet](#) in *AWS CLI Command Reference*.

create-datastore

The following code example shows how to use create-datastore.

AWS CLI

To create a data store

The following create-datastore code example creates a data store with the name my-datastore.

```
aws medical-imaging create-datastore \
  --datastore-name "my-datastore"
```

Output:

```
{
  "datastoreId": "12345678901234567890123456789012",
  "datastoreStatus": "CREATING"
}
```

For more information, see [Creating a data store](#) in the *AWS HealthImaging Developer Guide*.

- For API details, see [CreateDatastore](#) in *AWS CLI Command Reference*.

delete-datastore

The following code example shows how to use delete-datastore.

AWS CLI

To delete a data store

The following `delete-datastore` code example deletes a data store.

```
aws medical-imaging delete-datastore \  
  --datastore-id "12345678901234567890123456789012"
```

Output:

```
{  
  "datastoreId": "12345678901234567890123456789012",  
  "datastoreStatus": "DELETING"  
}
```

For more information, see [Deleting a data store](#) in the *AWS HealthImaging Developer Guide*.

- For API details, see [DeleteDatastore](#) in *AWS CLI Command Reference*.

delete-image-set

The following code example shows how to use `delete-image-set`.

AWS CLI

To delete an image set

The following `delete-image-set` code example deletes an image set.

```
aws medical-imaging delete-image-set \  
  --datastore-id 12345678901234567890123456789012 \  
  --image-set-id ea92b0d8838c72a3f25d00d13616f87e
```

Output:

```
{  
  "imageSetWorkflowStatus": "DELETING",  
  "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",  
  "imageSetState": "LOCKED",  
  "datastoreId": "12345678901234567890123456789012"
```

```
}
```

For more information, see [Deleting an image set](#) in the *AWS HealthImaging Developer Guide*.

- For API details, see [DeleteImageSet](#) in *AWS CLI Command Reference*.

get-datastore

The following code example shows how to use `get-datastore`.

AWS CLI

To get a data store's properties

The following `get-datastore` code example gets a data store's properties.

```
aws medical-imaging get-datastore \
  --datastore-id 12345678901234567890123456789012
```

Output:

```
{
  "datastoreProperties": {
    "datastoreId": "12345678901234567890123456789012",
    "datastoreName": "TestDatastore123",
    "datastoreStatus": "ACTIVE",
    "datastoreArn": "arn:aws:medical-imaging:us-
east-1:123456789012:datastore/12345678901234567890123456789012",
    "createdAt": "2022-11-15T23:33:09.643000+00:00",
    "updatedAt": "2022-11-15T23:33:09.643000+00:00"
  }
}
```

For more information, see [Getting data store properties](#) in the *AWS HealthImaging Developer Guide*.

- For API details, see [GetDatastore](#) in *AWS CLI Command Reference*.

get-dicom-import-job

The following code example shows how to use `get-dicom-import-job`.

AWS CLI

To get a dicom import job's properties

The following `get-dicom-import-job` code example gets a dicom import job's properties.

```
aws medical-imaging get-dicom-import-job \  
  --datastore-id "12345678901234567890123456789012" \  
  --job-id "09876543210987654321098765432109"
```

Output:

```
{  
  "jobProperties": {  
    "jobId": "09876543210987654321098765432109",  
    "jobName": "my-job",  
    "jobStatus": "COMPLETED",  
    "datastoreId": "12345678901234567890123456789012",  
    "dataAccessRoleArn": "arn:aws:iam::123456789012:role/  
ImportJobDataAccessRole",  
    "endedAt": "2022-08-12T11:29:42.285000+00:00",  
    "submittedAt": "2022-08-12T11:28:11.152000+00:00",  
    "inputS3Uri": "s3://medical-imaging-dicom-input/dicom_input/",  
    "outputS3Uri": "s3://medical-imaging-output/  
job_output/12345678901234567890123456789012-  
DicomImport-09876543210987654321098765432109/"  
  }  
}
```

For more information, see [Getting import job properties](#) in the *AWS HealthImaging Developer Guide*.

- For API details, see [GetDICOMImportJob](#) in *AWS CLI Command Reference*.

get-image-frame

The following code example shows how to use `get-image-frame`.

AWS CLI

To get image set pixel data

The following `get-image-frame` code example gets an image frame.

```
aws medical-imaging get-image-frame \  
  --datastore-id "12345678901234567890123456789012" \  
  --image-set-id "98765412345612345678907890789012" \  
  --image-frame-information imageFrameId=3abf5d5d7ae72f80a0ec81b2c0de3ef4 \  
  imageframe.jph
```

Note: This code example does not include output because the `GetImageFrame` action returns a stream of pixel data to the `imageframe.jph` file. For information about decoding and viewing image frames, see HTJ2K decoding libraries.

For more information, see [Getting image set pixel data](#) in the *AWS HealthImaging Developer Guide*.

- For API details, see [GetImageFrame](#) in *AWS CLI Command Reference*.

get-image-set-metadata

The following code example shows how to use `get-image-set-metadata`.

AWS CLI

Example 1: To get image set metadata without version

The following `get-image-set-metadata` code example gets metadata for an image set without specifying a version.

Note: `outfile` is a required parameter

```
aws medical-imaging get-image-set-metadata \  
  --datastore-id 12345678901234567890123456789012 \  
  --image-set-id ea92b0d8838c72a3f25d00d13616f87e \  
  studymetadata.json.gz
```

The returned metadata is compressed with gzip and stored in the `studymetadata.json.gz` file. To view the contents of the returned JSON object, you must first decompress it.

Output:

```
{  
  "contentType": "application/json",
```

```
"contentEncoding": "gzip"  
}
```

Example 2: To get image set metadata with version

The following `get-image-set-metadata` code example gets metadata for an image set with a specified version.

Note: `outfile` is a required parameter

```
aws medical-imaging get-image-set-metadata \  
  --datastore-id 12345678901234567890123456789012 \  
  --image-set-id ea92b0d8838c72a3f25d00d13616f87e \  
  --version-id 1 \  
  studymetadata.json.gz
```

The returned metadata is compressed with gzip and stored in the `studymetadata.json.gz` file. To view the contents of the returned JSON object, you must first decompress it.

Output:

```
{  
  "contentType": "application/json",  
  "contentEncoding": "gzip"  
}
```

For more information, see [Getting image set metadata](#) in the *AWS HealthImaging Developer Guide*.

- For API details, see [GetImageSetMetadata](#) in *AWS CLI Command Reference*.

get-image-set

The following code example shows how to use `get-image-set`.

AWS CLI

To get image set properties

The following `get-image-set` code example gets the properties for an image set.

```
aws medical-imaging get-image-set \  
  --datastore-id 12345678901234567890123456789012 \  
  --image-set-id 18f88ac7870584f58d56256646b4d92b \  
  --version-id 1
```

Output:

```
{  
  "versionId": "1",  
  "imageSetWorkflowStatus": "COPIED",  
  "updatedAt": 1680027253.471,  
  "imageSetId": "18f88ac7870584f58d56256646b4d92b",  
  "imageSetState": "ACTIVE",  
  "createdAt": 1679592510.753,  
  "datastoreId": "12345678901234567890123456789012"  
}
```

For more information, see [Getting image set properties](#) in the *AWS HealthImaging Developer Guide*.

- For API details, see [GetImageSet](#) in *AWS CLI Command Reference*.

list-datastores

The following code example shows how to use `list-datastores`.

AWS CLI

To list data stores

The following `list-datastores` code example lists available data stores.

```
aws medical-imaging list-datastores
```

Output:

```
{  
  "datastoreSummaries": [  
    {  
      "datastoreId": "12345678901234567890123456789012",
```

```
        "datastoreName": "TestDatastore123",
        "datastoreStatus": "ACTIVE",
        "datastoreArn": "arn:aws:medical-imaging:us-
east-1:123456789012:datastore/12345678901234567890123456789012",
        "createdAt": "2022-11-15T23:33:09.643000+00:00",
        "updatedAt": "2022-11-15T23:33:09.643000+00:00"
    }
]
}
```

For more information, see [Listing data stores](#) in the *AWS HealthImaging Developer Guide*.

- For API details, see [ListDatastores](#) in *AWS CLI Command Reference*.

list-dicom-import-jobs

The following code example shows how to use `list-dicom-import-jobs`.

AWS CLI

To list dicom import jobs

The following `list-dicom-import-jobs` code example lists dicom import jobs.

```
aws medical-imaging list-dicom-import-jobs \
  --datastore-id "12345678901234567890123456789012"
```

Output:

```
{
  "jobSummaries": [
    {
      "jobId": "09876543210987654321098765432109",
      "jobName": "my-job",
      "jobStatus": "COMPLETED",
      "datastoreId": "12345678901234567890123456789012",
      "dataAccessRoleArn": "arn:aws:iam::123456789012:role/
ImportJobDataAccessRole",
      "endedAt": "2022-08-12T11:21:56.504000+00:00",
      "submittedAt": "2022-08-12T11:20:21.734000+00:00"
    }
  ]
}
```

```
}
```

For more information, see [Listing import jobs](#) in the *AWS HealthImaging Developer Guide*.

- For API details, see [ListDICOMImportJobs](#) in *AWS CLI Command Reference*.

list-image-set-versions

The following code example shows how to use `list-image-set-versions`.

AWS CLI

To list image set versions

The following `list-image-set-versions` code example lists the version history for an image set.

```
aws medical-imaging list-image-set-versions \  
  --datastore-id 12345678901234567890123456789012 \  
  --image-set-id ea92b0d8838c72a3f25d00d13616f87e
```

Output:

```
{  
  "imageSetPropertiesList": [  
    {  
      "ImageSetWorkflowStatus": "UPDATED",  
      "versionId": "4",  
      "updatedAt": 1680029436.304,  
      "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",  
      "imageSetState": "ACTIVE",  
      "createdAt": 1680027126.436  
    },  
    {  
      "ImageSetWorkflowStatus": "UPDATED",  
      "versionId": "3",  
      "updatedAt": 1680029163.325,  
      "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",  
      "imageSetState": "ACTIVE",  
      "createdAt": 1680027126.436  
    },  
    {
```

```

    "ImageSetWorkflowStatus": "COPY_FAILED",
    "versionId": "2",
    "updatedAt": 1680027455.944,
    "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",
    "imageSetState": "ACTIVE",
    "message": "INVALID_REQUEST: Series of SourceImageSet and
DestinationImageSet don't match.",
    "createdAt": 1680027126.436
  },
  {
    "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",
    "imageSetState": "ACTIVE",
    "versionId": "1",
    "ImageSetWorkflowStatus": "COPIED",
    "createdAt": 1680027126.436
  }
]
}

```

For more information, see [Listing image set versions](#) in the *AWS HealthImaging Developer Guide*.

- For API details, see [ListImageSetVersions](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

Example 1: To list resource tags for a data store

The following `list-tags-for-resource` code example lists tags for a data store.

```

aws medical-imaging list-tags-for-resource \
  --resource-arn "arn:aws:medical-imaging:us-
east-1:123456789012: datastore/12345678901234567890123456789012"

```

Output:

```

{
  "tags":{
    "Deployment":"Development"
  }
}

```

```
}  
}
```

Example 2: To list resource tags for an image set

The following `list-tags-for-resource` code example lists tags for an image set.

```
aws medical-imaging list-tags-for-resource \  
  --resource-arn "arn:aws:medical-imaging:us-  
east-1:123456789012:datastore/12345678901234567890123456789012/  
imageset/18f88ac7870584f58d56256646b4d92b"
```

Output:

```
{  
  "tags":{  
    "Deployment":"Development"  
  }  
}
```

For more information, see [Tagging resources with AWS HealthImaging](#) in the *AWS HealthImaging Developer Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

search-image-sets

The following code example shows how to use `search-image-sets`.

AWS CLI

Example 1: To search image sets with an EQUAL operator

The following `search-image-sets` code example uses the EQUAL operator to search image sets based on a specific value.

```
aws medical-imaging search-image-sets \  
  --datastore-id 12345678901234567890123456789012 \  
  --search-criteria file://search-criteria.json
```

Contents of `search-criteria.json`


```
{
  "filters": [{
    "values": [{"DICOMPatientId" : "SUBJECT08701"}],
    "operator": "EQUAL"
  }]
}
```

Output:

```
{
  "imageSetsMetadataSummaries": [{
    "imageSetId": "09876543210987654321098765432109",
    "createdAt": "2022-12-06T21:40:59.429000+00:00",
    "version": 1,
    "DICOMTags": {
      "DICOMStudyId": "2011201407",
      "DICOMStudyDate": "19991122",
      "DICOMPatientSex": "F",
      "DICOMStudyInstanceUID": "1.2.840.99999999.84710745.943275268089",
      "DICOMPatientBirthDate": "19201120",
      "DICOMStudyDescription": "UNKNOWN",
      "DICOMPatientId": "SUBJECT08701",
      "DICOMPatientName": "Melissa844 Huel628",
      "DICOMNumberOfStudyRelatedInstances": 1,
      "DICOMStudyTime": "140728",
      "DICOMNumberOfStudyRelatedSeries": 1
    },
    "updatedAt": "2022-12-06T21:40:59.429000+00:00"
  }]
}
```

Example 2: To search image sets with a BETWEEN operator using DICOMStudyDate and DICOMStudyTime

The following `search-image-sets` code example searches for image sets with DICOM Studies generated between January 1, 1990 (12:00 AM) and January 1, 2023 (12:00 AM).

Note: `DICOMStudyTime` is optional. If it is not present, 12:00 AM (start of the day) is the time value for the dates provided for filtering.

```
aws medical-imaging search-image-sets \
  --datastore-id 12345678901234567890123456789012 \
```

```
--search-criteria file://search-criteria.json
```

Contents of search-criteria.json

```
{
  "filters": [{
    "values": [{
      "DICOMStudyDateAndTime": {
        "DICOMStudyDate": "19900101",
        "DICOMStudyTime": "000000"
      }
    },
    {
      "DICOMStudyDateAndTime": {
        "DICOMStudyDate": "20230101",
        "DICOMStudyTime": "000000"
      }
    }
  ]},
  "operator": "BETWEEN"
}]
}
```

Output:

```
{
  "imageSetsMetadataSummaries": [{
    "imageSetId": "09876543210987654321098765432109",
    "createdAt": "2022-12-06T21:40:59.429000+00:00",
    "version": 1,
    "DICOMTags": {
      "DICOMStudyId": "2011201407",
      "DICOMStudyDate": "19991122",
      "DICOMPatientSex": "F",
      "DICOMStudyInstanceUID": "1.2.840.99999999.84710745.943275268089",
      "DICOMPatientBirthDate": "19201120",
      "DICOMStudyDescription": "UNKNOWN",
      "DICOMPatientId": "SUBJECT08701",
      "DICOMPatientName": "Melissa844 Huel628",
      "DICOMNumberOfStudyRelatedInstances": 1,
      "DICOMStudyTime": "140728",
      "DICOMNumberOfStudyRelatedSeries": 1
    },
    "updatedAt": "2022-12-06T21:40:59.429000+00:00"
  ]
}
```

```
    ]]  
  }
```

Example 3: To search image sets with a BETWEEN operator using createdAt (time studies were previously persisted)

The following `search-image-sets` code example searches for image sets with DICOM Studies persisted in HealthImaging between the time ranges in UTC time zone.

Note: Provide `createdAt` in example format ("1985-04-12T23:20:50.52Z").

```
aws medical-imaging search-image-sets \  
  --datastore-id 12345678901234567890123456789012 \  
  --search-criteria file://search-criteria.json
```

Contents of `search-criteria.json`

```
{  
  "filters": [{  
    "values": [{  
      "createdAt": "1985-04-12T23:20:50.52Z"  
    },  
    {  
      "createdAt": "2022-04-12T23:20:50.52Z"  
    }],  
    "operator": "BETWEEN"  
  }]  
}
```

Output:

```
{  
  "imageSetsMetadataSummaries": [{  
    "imageSetId": "09876543210987654321098765432109",  
    "createdAt": "2022-12-06T21:40:59.429000+00:00",  
    "version": 1,  
    "DICOMTags": {  
      "DICOMStudyId": "2011201407",  
      "DICOMStudyDate": "19991122",  
      "DICOMPatientSex": "F",  
      "DICOMStudyInstanceUID": "1.2.840.99999999.84710745.943275268089",  
    }  
  }  
}
```

```

        "DICOMPatientBirthDate": "19201120",
        "DICOMStudyDescription": "UNKNOWN",
        "DICOMPatientId": "SUBJECT08701",
        "DICOMPatientName": "Melissa844 Huel628",
        "DICOMNumberOfStudyRelatedInstances": 1,
        "DICOMStudyTime": "140728",
        "DICOMNumberOfStudyRelatedSeries": 1
    },
    "lastUpdatedAt": "2022-12-06T21:40:59.429000+00:00"
  ]
}

```

Example 4: To search image sets with an EQUAL operator on DICOMSeriesInstanceUID and BETWEEN on updatedAt and sort response in ASC order on updatedAt field

The following `search-image-sets` code example searches for image sets with an EQUAL operator on DICOMSeriesInstanceUID and BETWEEN on updatedAt and sort response in ASC order on updatedAt field.

Note: Provide updatedAt in example format ("1985-04-12T23:20:50.52Z").

```

aws medical-imaging search-image-sets \
  --datastore-id 12345678901234567890123456789012 \
  --search-criteria file://search-criteria.json

```

Contents of search-criteria.json

```

{
  "filters": [{
    "values": [{
      "updatedAt": "2024-03-11T15:00:05.074000-07:00"
    }, {
      "updatedAt": "2024-03-11T16:00:05.074000-07:00"
    }],
    "operator": "BETWEEN"
  }, {
    "values": [{
      "DICOMSeriesInstanceUID": "1.2.840.99999999.84710745.943275268089"
    }],
    "operator": "EQUAL"
  }],
  "sort": {

```

```
    "sortField": "updatedAt",
    "sortOrder": "ASC"
  }
}
```

Output:

```
{
  "imageSetsMetadataSummaries": [{
    "imageSetId": "09876543210987654321098765432109",
    "createdAt": "2022-12-06T21:40:59.429000+00:00",
    "version": 1,
    "DICOMTags": {
      "DICOMStudyId": "2011201407",
      "DICOMStudyDate": "19991122",
      "DICOMPatientSex": "F",
      "DICOMStudyInstanceUID": "1.2.840.99999999.84710745.943275268089",
      "DICOMPatientBirthDate": "19201120",
      "DICOMStudyDescription": "UNKNOWN",
      "DICOMPatientId": "SUBJECT08701",
      "DICOMPatientName": "Melissa844 Huel628",
      "DICOMNumberOfStudyRelatedInstances": 1,
      "DICOMStudyTime": "140728",
      "DICOMNumberOfStudyRelatedSeries": 1
    },
    "lastUpdatedAt": "2022-12-06T21:40:59.429000+00:00"
  ]
}
```

For more information, see [Searching image sets](#) in the *AWS HealthImaging Developer Guide*.

- For API details, see [SearchImageSets](#) in *AWS CLI Command Reference*.

start-dicom-import-job

The following code example shows how to use `start-dicom-import-job`.

AWS CLI

To start a dicom import job

The following `start-dicom-import-job` code example starts a dicom import job.

```
aws medical-imaging start-dicom-import-job \  
  --job-name "my-job" \  
  --datastore-id "12345678901234567890123456789012" \  
  --input-s3-uri "s3://medical-imaging-dicom-input/dicom_input/" \  
  --output-s3-uri "s3://medical-imaging-output/job_output/" \  
  --data-access-role-arn "arn:aws:iam::123456789012:role/ImportJobDataAccessRole"
```

Output:

```
{  
  "datastoreId": "12345678901234567890123456789012",  
  "jobId": "09876543210987654321098765432109",  
  "jobStatus": "SUBMITTED",  
  "submittedAt": "2022-08-12T11:28:11.152000+00:00"  
}
```

For more information, see [Starting an import job](#) in the *AWS HealthImaging Developer Guide*.

- For API details, see [StartDICOMImportJob](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

Example 1: To tag a data store

The following tag-resource code examples tags a data store.

```
aws medical-imaging tag-resource \  
  --resource-arn "arn:aws:medical-imaging:us-  
east-1:123456789012:datastore/12345678901234567890123456789012" \  
  --tags '{"Deployment":"Development}"'
```

This command produces no output.

Example 2: To tag an image set

The following tag-resource code examples tags an image set.

```
aws medical-imaging tag-resource \  
  --resource-arn "arn:aws:medical-imaging:us-  
east-1:123456789012:datastore/12345678901234567890123456789012/  
imageset/18f88ac7870584f58d56256646b4d92b" \  
  --tags '{"Deployment":"Development"}'
```

This command produces no output.

For more information, see [Tagging resources with AWS HealthImaging](#) in the *AWS HealthImaging Developer Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

Example 1: To untag a data store

The following `untag-resource` code example untags a data store.

```
aws medical-imaging untag-resource \  
  --resource-arn "arn:aws:medical-imaging:us-  
east-1:123456789012:datastore/12345678901234567890123456789012" \  
  --tag-keys ["Deployment"]'
```

This command produces no output.

Example 2: To untag an image set

The following `untag-resource` code example untags an image set.

```
aws medical-imaging untag-resource \  
  --resource-arn "arn:aws:medical-imaging:us-  
east-1:123456789012:datastore/12345678901234567890123456789012/  
imageset/18f88ac7870584f58d56256646b4d92b" \  
  --tag-keys ["Deployment"]'
```

This command produces no output.

For more information, see [Tagging resources with AWS HealthImaging](#) in the *AWS HealthImaging Developer Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-image-set-metadata

The following code example shows how to use `update-image-set-metadata`.

AWS CLI

To insert or update an attribute in image set metadata

The following `update-image-set-metadata` code example inserts or updates an attribute in image set metadata.

```
aws medical-imaging update-image-set-metadata \
  --datastore-id 12345678901234567890123456789012 \
  --image-set-id ea92b0d8838c72a3f25d00d13616f87e \
  --latest-version-id 1 \
  --update-image-set-metadata-updates file://metadata-updates.json
```

Contents of `metadata-updates.json`

```
{
  "DICOMUpdates": {
    "updatableAttributes":
      "eyJTY2h1bWFWZXJzaW9uIjoxLjEsIlBhdGllbnQiOnsiRElDT00iOnsiUGF0aWVudE5hbWUiOiJNWf5NWCJ9fX0="
  }
}
```

Note: `updatableAttributes` is a Base64 encoded JSON string. Here is the unencoded JSON string.

```
{"SchemaVersion":1.1,"Patient":{"DICOM":{"PatientName":"MX^MX"}}
```

Output:

```
{
  "latestVersionId": "2",
  "imageSetWorkflowStatus": "UPDATING",
```



```

    "updatedAt": 1680042257.908,
    "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",
    "imageSetState": "LOCKED",
    "createdAt": 1680027126.436,
    "datastoreId": "12345678901234567890123456789012"
  }

```

To remove an attribute from image set metadata

The following `update-image-set-metadata` code example removes an attribute from image set metadata.

```

aws medical-imaging update-image-set-metadata \
  --datastore-id 12345678901234567890123456789012 \
  --image-set-id ea92b0d8838c72a3f25d00d13616f87e \
  --latest-version-id 1 \
  --update-image-set-metadata-updates file://metadata-updates.json

```

Contents of `metadata-updates.json`

```

{
  "DICOMUpdates": {
    "removableAttributes":
    "e1NjaGVtYVZlcnNpb246MS4xLFN0dWR50ntESUNPTTp7U3R1ZH1EZXNjcmlwdGlvbjpDSEVTVH19fQo="
  }
}

```

Note: `removableAttributes` is a Base64 encoded JSON string. Here is the unencoded JSON string. The key and value must match the attribute to be removed.

```
{"SchemaVersion":1.1,"Study":{"DICOM":{"StudyDescription":"CHEST"}}
```

Output:

```

{
  "latestVersionId": "2",
  "imageSetWorkflowStatus": "UPDATING",
  "updatedAt": 1680042257.908,
  "imageSetId": "ea92b0d8838c72a3f25d00d13616f87e",
  "imageSetState": "LOCKED",
  "createdAt": 1680027126.436,

```


For more information, see [Updating image set metadata](#) in the *AWS HealthImaging Developer Guide*.

- For API details, see [UpdateImageSetMetadata](#) in *AWS CLI Command Reference*.

HealthLake examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with HealthLake.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-fhir-datastore

The following code example shows how to use `create-fhir-datastore`.

AWS CLI

To create a FHIR Data Store.

The following `create-fhir-datastore` example demonstrates how to create a new Data Store in Amazon HealthLake.

```
aws healthlake create-fhir-datastore \  
  --region us-east-1 \  
  --datastore-type-version R4 \  
  --datastore-type-version R4 \  
  --datastore-type-version R4 \
```

```
--datastore-name "FhirTestDatastore"
```

Output:

```
{
  "DatastoreEndpoint": "https://healthlake.us-east-1.amazonaws.com/datastore/
(Datastore ID)/r4/",
  "DatastoreArn": "arn:aws:healthlake:us-east-1:(AWS Account ID):datastore/
(Datastore ID)",
  "DatastoreStatus": "CREATING",
  "DatastoreId": "(Datastore ID)"
}
```

For more information, see [Creating and monitoring a FHIR Data Store](#) in the *Amazon HealthLake Developer Guide*.

- For API details, see [CreateFhirDatastore](#) in *AWS CLI Command Reference*.

delete-fhir-datastore

The following code example shows how to use `delete-fhir-datastore`.

AWS CLI

To delete a FHIR Data Store

The following `delete-fhir-datastore` example demonstrates how to delete a Data Store and all of its contents in Amazon HealthLake.

```
aws healthlake delete-fhir-datastore \
  --datastore-id (Data Store ID) \
  --region us-east-1
```

Output:

```
{
  "DatastoreEndpoint": "https://healthlake.us-east-1.amazonaws.com/datastore/
(Datastore ID)/r4/",
  "DatastoreArn": "arn:aws:healthlake:us-east-1:(AWS Account ID):datastore/
(Datastore ID)",
  "DatastoreStatus": "DELETING",
}
```

```
"DatastoreId": "(Datastore ID)"
}
```

For more information, see [Creating and monitoring a FHIR Data Store](https://docs.aws.amazon.com/healthlake/latest/devguide/working-with-FHIR-healthlake.html) <<https://docs.aws.amazon.com/healthlake/latest/devguide/working-with-FHIR-healthlake.html>> in the *Amazon HealthLake Developer Guide*.

- For API details, see [DeleteFhirDatastore](#) in *AWS CLI Command Reference*.

describe-fhir-datastore

The following code example shows how to use `describe-fhir-datastore`.

AWS CLI

To describe a FHIR Data Store

The following `describe-fhir-datastore` example demonstrates how to find the properties of a Data Store in Amazon HealthLake.

```
aws healthlake describe-fhir-datastore \
  --datastore-id "1f2f459836ac6c513ce899f9e4f66a59" \
  --region us-east-1
```

Output:

```
{
  "DatastoreProperties": {
    "PreloadDataConfig": {
      "PreloadDataType": "SYNTHEA"
    },
    "DatastoreName": "FhirTestDatastore",
    "DatastoreArn": "arn:aws:healthlake:us-east-1:(AWS Account ID):datastore/
(Datastore ID)",
    "DatastoreEndpoint": "https://healthlake.us-east-1.amazonaws.com/datastore/
(Datastore ID)/r4/",
    "DatastoreStatus": "CREATING",
    "DatastoreTypeVersion": "R4",
    "DatastoreId": "(Datastore ID)"
  }
}
```

For more information, see [Creating and monitoring a FHIR Data Stores](#) in the *Amazon HealthLake Developer Guide*.

- For API details, see [DescribeFhirDatastore](#) in *AWS CLI Command Reference*.

describe-fhir-export-job

The following code example shows how to use `describe-fhir-export-job`.

AWS CLI

To describe a FHIR export job

The following `describe-fhir-export-job` example shows how to find the properties of a FHIR export job in Amazon HealthLake.

```
aws healthlake describe-fhir-export-job \  
  --datastore-id (Datastore ID) \  
  --job-id 9b9a51943afaedd0a8c0c26c49135a31
```

Output:

```
{  
  "ExportJobProperties": {  
    "DataAccessRoleArn": "arn:aws:iam::(AWS Account ID):role/(Role Name)",  
    "JobStatus": "IN_PROGRESS",  
    "JobId": "9009813e9d69ba7cf79bcb3468780f16",  
    "SubmitTime": 1609175692.715,  
    "OutputDataConfig": {  
      "S3Uri": "s3://(Bucket Name)/(Prefix  
Name)/59593b2d0367ce252b5e66bf5fd6b574-  
FHIR_EXPORT-9009813e9d69ba7cf79bcb3468780f16/"  
    },  
    "DatastoreId": "(Datastore ID)"  
  }  
}
```

For more information, see [Exporting files from a FHIR Data Store](#) in the *Amazon HealthLake Developer Guide*.

- For API details, see [DescribeFhirExportJob](#) in *AWS CLI Command Reference*.

describe-fhir-import-job

The following code example shows how to use `describe-fhir-import-job`.

AWS CLI

To describe a FHIR import job

The following `describe-fhir-import-job` example shows how to learn the properties of a FHIR import job using Amazon HealthLake.

```
aws healthlake describe-fhir-import-job \  
  --datastore-id (Datastore ID) \  
  --job-id c145fbb27b192af392f8ce6e7838e34f \  
  --region us-east-1
```

Output:

```
{  
  "ImportJobProperties": {  
    "InputDataConfig": {  
      "S3Uri": "s3://(Bucket Name)/(Prefix Name)/"  
      { "arrayitem2": 2 }  
    },  
    "DataAccessRoleArn": "arn:aws:iam::(AWS Account ID):role/(Role Name)",  
    "JobStatus": "COMPLETED",  
    "JobId": "c145fbb27b192af392f8ce6e7838e34f",  
    "SubmitTime": 1606272542.161,  
    "EndTime": 1606272609.497,  
    "DatastoreId": "(Datastore ID)"  
  }  
}
```

For more information, see [Importing files to a FHIR Data Store](#) in the *Amazon HealthLake Developer Guide*.

- For API details, see [DescribeFhirImportJob](#) in *AWS CLI Command Reference*.

list-fhir-datastores

The following code example shows how to use `list-fhir-datastores`.

AWS CLI

To list FHIR Data Stores

The following `list-fhir-datastores` example shows to how to use the command and how users can filter results based on Data Store status in Amazon HealthLake.

```
aws healthlake list-fhir-datastores \
  --region us-east-1 \
  --filter DatastoreStatus=ACTIVE
```

Output:

```
{
  "DatastorePropertiesList": [
    {
      "PreloadDataConfig": {
        "PreloadDataType": "SYNTHEA"
      },
      "DatastoreName": "FhirTestDatastore",
      "DatastoreArn": "arn:aws:healthlake:us-east-1:<AWS Account ID>:datastore/
<Datastore ID>",
      "DatastoreEndpoint": "https://healthlake.us-east-1.amazonaws.com/datastore/
<Datastore ID>/r4/",
      "DatastoreStatus": "ACTIVE",
      "DatastoreTypeVersion": "R4",
      "CreatedAt": 1605574003.209,
      "DatastoreId": "<Datastore ID>"
    },
    {
      "DatastoreName": "Demo",
      "DatastoreArn": "arn:aws:healthlake:us-east-1:<AWS Account ID>:datastore/
<Datastore ID>",
      "DatastoreEndpoint": "https://healthlake.us-east-1.amazonaws.com/datastore/
<Datastore ID>/r4/",
      "DatastoreStatus": "ACTIVE",
      "DatastoreTypeVersion": "R4",
      "CreatedAt": 1603761064.881,
      "DatastoreId": "<Datastore ID>"
    }
  ]
}
```


For more information, see [Creating and monitoring a FHIR Data Store](#) in the *Amazon HealthLake Developer Guide*.

- For API details, see [ListFhirDatastores](#) in *AWS CLI Command Reference*.

list-fhir-export-jobs

The following code example shows how to use `list-fhir-export-jobs`.

AWS CLI

To list all FHIR export jobs

The following `list-fhir-export-jobs` example shows how to use the command to view a list of export jobs associated with an account.

```
aws healthlake list-fhir-export-jobs \  
  --datastore-id (Datastore ID) \  
  --submitted-before (DATE like 2024-10-13T19:00:00Z)\ \  
  --submitted-after (DATE like 2020-10-13T19:00:00Z) \  
  --job-name "FHIR-EXPORT" \  
  --job-status SUBMITTED \  
  --max-results (Integer between 1 and 500)
```

Output:

```
{  
  "ExportJobProperties": {  
    "OutputDataConfig": {  
      "S3Uri": "s3://(Bucket Name)/(Prefix Name)/"  
      "S3Configuration": {  
        "S3Uri": "s3://(Bucket Name)/(Prefix Name)/",  
        "KmsKeyId" : "(KmsKey Id)"  
      },  
    },  
  },  
  "DataAccessRoleArn": "arn:aws:iam::(AWS Account ID):role/(Role Name)",  
  "JobStatus": "COMPLETED",  
  "JobId": "c145fbb27b192af392f8ce6e7838e34f",  
  "JobName": "FHIR-EXPORT",  
  "SubmitTime": 1606272542.161,  
  "EndTime": 1606272609.497,  
  "DatastoreId": "(Datastore ID)"
```

```

    }
  }
  "NextToken": String

```

For more information, see [Exporting files from a FHIR Data Store](#) in the Amazon HealthLake Developer Guide.

- For API details, see [ListFhirExportJobs](#) in *AWS CLI Command Reference*.

list-fhir-import-jobs

The following code example shows how to use `list-fhir-import-jobs`.

AWS CLI

To list all FHIR import jobs

The following `list-fhir-import-jobs` example shows how to use the command to view a list of all import jobs associated with an account.

```

aws healthlake list-fhir-import-jobs \
  --datastore-id (Datastore ID) \
  --submitted-before (DATE like 2024-10-13T19:00:00Z) \
  --submitted-after (DATE like 2020-10-13T19:00:00Z ) \
  --job-name "FHIR-IMPORT" \
  --job-status SUBMITTED \
  -max-results (Integer between 1 and 500)

```

Output:

```

{
  "ImportJobProperties": {
    "OutputDataConfig": {
      "S3Uri": "s3://(Bucket Name)/(Prefix Name)/",
      "S3Configuration": {
        "S3Uri": "s3://(Bucket Name)/(Prefix Name)/",
        "KmsKeyId" : "(KmsKey Id)"
      }
    },
    "DataAccessRoleArn": "arn:aws:iam::(AWS Account ID):role/(Role Name)",
    "JobStatus": "COMPLETED",
  }
}

```

```
"JobId": "c145fbb27b192af392f8ce6e7838e34f",
"JobName" "FHIR-IMPORT",
"SubmitTime": 1606272542.161,
"EndTime": 1606272609.497,
"DatastoreId": "(Datastore ID)"
}
}
"NextToken": String
```

For more information, see [Importing files to FHIR Data Store](#) in the Amazon HealthLake Developer Guide.

- For API details, see [ListFhirImportJobs](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags for a Data Store

The following `list-tags-for-resource` example lists the tags associated with the specified Data Store.:

```
aws healthlake list-tags-for-resource \
  --resource-arn "arn:aws:healthlake:us-east-1:674914422125:datastore/
  fhir/0725c83f4307f263e16fd56b6d8ebdbe" \
  --region us-east-1
```

Output:

```
{
  "tags": {
    "key": "value",
    "key1": "value1"
  }
}
```

For more information, see [Tagging resources in Amazon HealthLake](#) in the Amazon HealthLake Developer Guide.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

start-fhir-export-job

The following code example shows how to use `start-fhir-export-job`.

AWS CLI

To start a FHIR export job

The following `start-fhir-export-job` example shows how to start a FHIR export job using Amazon HealthLake.

```
aws healthlake start-fhir-export-job \  
  --output-data-config S3Uri="s3://(Bucket Name)/(Prefix Name)/" \  
  --datastore-id (Datastore ID) \  
  --data-access-role-arn arn:aws:iam::(AWS Account ID):role/(Role Name)
```

Output:

```
{  
  "DatastoreId": "(Datastore ID)",  
  "JobStatus": "SUBMITTED",  
  "JobId": "9b9a51943afaedd0a8c0c26c49135a31"  
}
```

For more information, see [Exporting files from a FHIR Data Store](#) in the *Amazon HealthLake Developer Guide*.

- For API details, see [StartFhirExportJob](#) in *AWS CLI Command Reference*.

start-fhir-import-job

The following code example shows how to use `start-fhir-import-job`.

AWS CLI

To start a FHIR import job

The following `start-fhir-import-job` example shows how to start a FHIR import job using Amazon HealthLake.

```
aws healthlake start-fhir-import-job \  
  --input-data-config S3Uri="s3://(Bucket Name)/(Prefix Name)/" \  
  --datastore-id (Datastore ID) \  
  --data-access-role-arn "arn:aws:iam::(AWS Account ID):role/(Role Name)" \  
  --region us-east-1
```

Output:

```
{  
  "DatastoreId": "(Datastore ID)",  
  "JobStatus": "SUBMITTED",  
  "JobId": "c145fbb27b192af392f8ce6e7838e34f"  
}
```

For more information, see Importing files to a FHIR Data Store <https://docs.aws.amazon.com/healthlake/latest/devguide/import-datastore.html> in the *Amazon HealthLake Developer Guide*.

- For API details, see [StartFhirImportJob](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add a tag to Data Store

The following `tag-resource` example shows how to add a tag to a Data Store.

```
aws healthlake tag-resource \  
  --resource-arn "arn:aws:healthlake:us-east-1:691207106566:datastore/  
fhir/0725c83f4307f263e16fd56b6d8ebdbe" \  
  --tags '[{"Key": "key1", "Value": "value1"}]' \  
  --region us-east-1
```

This command produces no output.

For more information, see 'Adding a tag to a Data Store <https://docs.aws.amazon.com/healthlake/latest/devguide/add-a-tag.html>' in the *Amazon HealthLake Developer Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from a Data Store.

The following `untag-resource` example shows how to remove tags from a Data Store.

```
aws healthlake untag-resource \  
  --resource-arn "arn:aws:healthlake:us-east-1:674914422125:datastore/fhir/  
b91723d65c6fdeb1d26543a49d2ed1fa" \  
  --tag-keys '["key1"]' \  
  --region us-east-1
```

This command produces no output.

For more information, see [Removing tags from a Data Store](#) in the *Amazon HealthLake Developer Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

HealthOmics examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with HealthOmics.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

abort-multipart-read-set-upload

The following code example shows how to use `abort-multipart-read-set-upload`.

AWS CLI

To stop a multipart read set upload

The following `abort-multipart-read-set-upload` example stops a multipart read set upload into your HealthOmics sequence store.

```
aws omics abort-multipart-read-set-upload \  
  --sequence-store-id 0123456789 \  
  --upload-id 1122334455
```

This command produces no output.

For more information, see [Direct upload to a sequence store](#) in the *AWS HealthOmics User Guide*.

- For API details, see [AbortMultipartReadSetUpload](#) in *AWS CLI Command Reference*.

accept-share

The following code example shows how to use `accept-share`.

AWS CLI

To accept a share of analytics store data

The following `accept-share` example accepts a share of HealthOmics analytics store data.

```
aws omics accept-share \  
  ----share-id "495c21bedc889d07d0ab69d710a6841e-dd75ab7a1a9c384fa848b5bd8e5a7e0a"
```

Output:

```
{  
  "status": "ACTIVATING"  
}
```

For more information, see [Cross-account sharing](#) in the *AWS HealthOmics User Guide*.

- For API details, see [AcceptShare](#) in *AWS CLI Command Reference*.

batch-delete-read-set

The following code example shows how to use `batch-delete-read-set`.

AWS CLI

To delete multiple read sets

The following `batch-delete-read-set` example deletes two read sets.

```
aws omics batch-delete-read-set \  
  --sequence-store-id 1234567890 \  
  --ids 1234567890 0123456789
```

If there is an error deleting any of the specified read sets, the service returns an error list.

```
{  
  "errors": [  
    {  
      "code": "",  
      "id": "0123456789",  
      "message": "The specified readset does not exist."  
    }  
  ]  
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [BatchDeleteReadSet](#) in *AWS CLI Command Reference*.

cancel-annotation-import-job

The following code example shows how to use `cancel-annotation-import-job`.

AWS CLI

To cancel an annotation import job

The following `cancel-annotation-import-job` example cancels an annotation import job with ID `04f57618-xmpl-4fd0-9349-e5a85aefb997`.

```
aws omics cancel-annotation-import-job \  
  --job-id 04f57618-xmpl-4fd0-9349-e5a85aefb997
```

For more information, see [Omics Analytics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [CancelAnnotationImportJob](#) in *AWS CLI Command Reference*.

cancel-run

The following code example shows how to use `cancel-run`.

AWS CLI

To cancel a run

The following `cancel-run` example cancels a run with ID `1234567`.

```
aws omics cancel-run \  
  --id 1234567
```

For more information, see [Omics Workflows](#) in the *Amazon Omics Developer Guide*.

- For API details, see [CancelRun](#) in *AWS CLI Command Reference*.

cancel-variant-import-job

The following code example shows how to use `cancel-variant-import-job`.

AWS CLI

To cancel a variant import job

The following `cancel-variant-import-job` example cancels a variant import job with ID `69cb65d6-xmpl-4a4a-9025-4565794b684e`.

```
aws omics cancel-variant-import-job \  
  --job-id 69cb65d6-xmpl-4a4a-9025-4565794b684e
```

For more information, see [Omics Analytics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [CancelVariantImportJob](#) in *AWS CLI Command Reference*.

complete-multipart-read-set-upload

The following code example shows how to use `complete-multipart-read-set-upload`.

AWS CLI

To conclude a multipart upload once you have uploaded all of the components.

The following `complete-multipart-read-set-upload` example concludes a multipart upload into a sequence store once all of the components have been uploaded.

```
aws omics complete-multipart-read-set-upload \  
  --sequence-store-id 0123456789 \  
  --upload-id 1122334455 \  
  --parts '[{"checksum":"gaCBQMe+rpCFZxLpoP6gydBoXaKKDA/  
Vobh5zBD4W4=", "partNumber":1, "partSource":"SOURCE1"}]'
```

Output:

```
{  
  "readSetId": "0000000001"  
  "readSetId": "0000000002"  
  "readSetId": "0000000003"  
}
```

For more information, see [Direct upload to a sequence store](#) in the *AWS HealthOmics User Guide*.

- For API details, see [CompleteMultipartReadSetUpload](#) in *AWS CLI Command Reference*.

create-annotation-store-version

The following code example shows how to use `create-annotation-store-version`.

AWS CLI

To create a new version of an annotation store

The following `create-annotation-store-version` example creates a new version of an annotation store.

```
aws omics create-annotation-store-version \  
  --name my_annotation_store \  
  --version-name my_version
```

Output:

```
{  
  "creationTime": "2023-07-21T17:15:49.251040+00:00",  
  "id": "3b93cdef69d2",  
  "name": "my_annotation_store",  
  "reference": {  
    "referenceArn": "arn:aws:omics:us-  
west-2:555555555555:referenceStore/6505293348/reference/5987565360"  
  },  
  "status": "CREATING",  
  "versionName": "my_version"  
}
```

For more information, see [Creating new versions of annotation stores](#) in the *AWS HealthOmics User Guide*.

- For API details, see [CreateAnnotationStoreVersion](#) in *AWS CLI Command Reference*.

create-annotation-store

The following code example shows how to use `create-annotation-store`.

AWS CLI

Example 1: To create a VCF annotation store

The following `create-annotation-store` example creates a VCF format annotation store.

```
aws omics create-annotation-store \  
  --name my_ann_store \  
  --store-format VCF \  
  --reference referenceArn=arn:aws:omics:us-  
west-2:123456789012:referenceStore/1234567890/reference/1234567890
```

Output:

```
{
```

```
"creationTime": "2022-11-23T22:48:39.226492Z",
"id": "0a91xmplc71f",
"name": "my_ann_store",
"reference": {
  "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890"
},
"status": "CREATING",
"storeFormat": "VCF"
}
```

Example 2: To create a TSV annotation store

The following `create-annotation-store` example creates a TSV format annotation store.

```
aws omics create-annotation-store \
  --name tsv_ann_store \
  --store-format TSV \
  --reference referenceArn=arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890 \
  --store-options file://tsv-store-options.json
```

`tsv-store-options.json` configures format options for annotations.

```
{
  "tsvStoreOptions": {
    "annotationType": "CHR_START_END_ZERO_BASE",
    "formatToHeader": {
      "CHR": "chromosome",
      "START": "start",
      "END": "end"
    },
    "schema": [
      {
        "chromosome": "STRING"
      },
      {
        "start": "LONG"
      },
      {
        "end": "LONG"
      }
    ]
  }
}
```

```

        "name": "STRING"
      }
    ]
  }
}

```

Output:

```

{
  "creationTime": "2022-11-30T01:28:08.525586Z",
  "id": "861cxmpl96b0",
  "name": "tsv_ann_store",
  "reference": {
    "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890"
  },
  "status": "CREATING",
  "storeFormat": "TSV",
  "storeOptions": {
    "tsvStoreOptions": {
      "annotationType": "CHR_START_END_ZERO_BASE",
      "formatToHeader": {
        "CHR": "chromosome",
        "END": "end",
        "START": "start"
      },
      "schema": [
        {
          "chromosome": "STRING"
        },
        {
          "start": "LONG"
        },
        {
          "end": "LONG"
        },
        {
          "name": "STRING"
        }
      ]
    }
  }
}

```

For more information, see [Omics Analytics](#) in the Amazon Omics Developer Guide.

- For API details, see [CreateAnnotationStore](#) in *AWS CLI Command Reference*.

create-multipart-read-set-upload

The following code example shows how to use `create-multipart-read-set-upload`.

AWS CLI

To begin a multipart read set upload.

The following `create-multipart-read-set-upload` example initiates a multipart read set upload.

```
aws omics create-multipart-read-set-upload \  
  --sequence-store-id 0123456789 \  
  --name HG00146 \  
  --source-file-type FASTQ \  
  --subject-id mySubject\  
  --sample-id mySample\  
  --description "FASTQ for HG00146"\  
  --generated-from "1000 Genomes"
```

Output:

```
{  
  "creationTime": "2022-07-13T23:25:20Z",  
  "description": "FASTQ for HG00146",  
  "generatedFrom": "1000 Genomes",  
  "name": "HG00146",  
  "sampleId": "mySample",  
  "sequenceStoreId": "0123456789",  
  "sourceFileType": "FASTQ",  
  "subjectId": "mySubject",  
  "uploadId": "1122334455"  
}
```

For more information, see [Direct upload to a sequence store](#) in the *AWS HealthOmics User Guide*.

- For API details, see [CreateMultipartReadSetUpload](#) in *AWS CLI Command Reference*.

create-reference-store

The following code example shows how to use `create-reference-store`.

AWS CLI

To create a reference store

The following `create-reference-store` example creates a reference store `my-ref-store`.

```
aws omics create-reference-store \  
  --name my-ref-store
```

Output:

```
{  
  "arn": "arn:aws:omics:us-west-2:123456789012:referenceStore/1234567890",  
  "creationTime": "2022-11-22T22:13:25.947Z",  
  "id": "1234567890",  
  "name": "my-ref-store"  
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [CreateReferenceStore](#) in *AWS CLI Command Reference*.

create-run-group

The following code example shows how to use `create-run-group`.

AWS CLI

To create a run group

The following `create-run-group` example creates a run group named `cram-converter`.

```
aws omics create-run-group \  
  --name cram-converter \  
  --max-cpus 20 \  
  --max-duration 600
```

Output:

```
{
  "arn": "arn:aws:omics:us-west-2:123456789012:runGroup/1234567",
  "id": "1234567",
  "tags": {}
}
```

For more information, see [Omics Workflows](#) in the *Amazon Omics Developer Guide*.

- For API details, see [CreateRunGroup](#) in *AWS CLI Command Reference*.

create-sequence-store

The following code example shows how to use `create-sequence-store`.

AWS CLI**To create a sequence store**

The following `create-sequence-store` example creates a sequence store.

```
aws omics create-sequence-store \
  --name my-seq-store
```

Output:

```
{
  "arn": "arn:aws:omics:us-west-2:123456789012:sequenceStore/1234567890",
  "creationTime": "2022-11-23T01:24:33.629Z",
  "id": "1234567890",
  "name": "my-seq-store"
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [CreateSequenceStore](#) in *AWS CLI Command Reference*.

create-share

The following code example shows how to use `create-share`.

AWS CLI

To create a share of a HealthOmics analytics store

The following `create-share` example shows how to create a share of a HealthOmics analytics store that can be accepted by a subscriber outside the account.

```
aws omics create-share \  
  --resource-arn "arn:aws:omics:us-west-2:555555555555:variantStore/  
omics_dev_var_store" \  
  --principal-subscriber "123456789012" \  
  --name "my_Share-123"
```

Output:

```
{  
  "shareId": "495c21bedc889d07d0ab69d710a6841e-dd75ab7a1a9c384fa848b5bd8e5a7e0a",  
  "name": "my_Share-123",  
  "status": "PENDING"  
}
```

For more information, see [Cross-account sharing](#) in the *AWS HealthOmics User Guide*.

- For API details, see [CreateShare](#) in *AWS CLI Command Reference*.

create-variant-store

The following code example shows how to use `create-variant-store`.

AWS CLI

To create a variant store

The following `create-variant-store` example creates a variant store named `my_var_store`.

```
aws omics create-variant-store \  
  --name my_var_store \  
  --reference referenceArn=arn:aws:omics:us-  
west-2:123456789012:referenceStore/1234567890/reference/1234567890
```

Output:

```
{
  "creationTime": "2022-11-23T22:09:07.534499Z",
  "id": "02dexplcfdd",
  "name": "my_var_store",
  "reference": {
    "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890"
  },
  "status": "CREATING"
}
```

For more information, see [Omics Analytics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [CreateVariantStore](#) in *AWS CLI Command Reference*.

create-workflow

The following code example shows how to use `create-workflow`.

AWS CLI

To create a workflow

The following `create-workflow` example creates a WDL workflow.

```
aws omics create-workflow \
  --name cram-converter \
  --engine WDL \
  --definition-zip fileb://workflow-crambam.zip \
  --parameter-template file://workflow-params.json
```

`workflow-crambam.zip` is a ZIP archive containing a workflow definition. `workflow-params.json` defines runtime parameters for the workflow.

```
{
  "ref_fasta" : {
    "description": "Reference genome fasta file",
    "optional": false
  },
  "ref_fasta_index" : {
    "description": "Index of the reference genome fasta file",
```

```
    "optional": false
  },
  "ref_dict" : {
    "description": "dictionary file for 'ref_fasta'",
    "optional": false
  },
  "input_cram" : {
    "description": "The Cram file to convert to BAM",
    "optional": false
  },
  "sample_name" : {
    "description": "The name of the input sample, used to name the output BAM",
    "optional": false
  }
}
```

Output:

```
{
  "arn": "arn:aws:omics:us-west-2:123456789012:workflow/1234567",
  "id": "1234567",
  "status": "CREATING",
  "tags": {}
}
```

For more information, see [Omics Workflows](#) in the *Amazon Omics Developer Guide*.

- For API details, see [CreateWorkflow](#) in *AWS CLI Command Reference*.

delete-annotation-store-versions

The following code example shows how to use delete-annotation-store-versions.

AWS CLI

To delete an annotation store version

The following delete-annotation-store-versions example deletes an annotation store version.

```
aws omics delete-annotation-store-versions \
  --name my_annotation_store \
```

```
--versions my_version
```

Output:

```
{  
  "errors": []  
}
```

For more information, see [Creating new versions of annotation stores](#) in the *AWS HealthOmics User Guide*.

- For API details, see [DeleteAnnotationStoreVersions](#) in *AWS CLI Command Reference*.

delete-annotation-store

The following code example shows how to use `delete-annotation-store`.

AWS CLI**To delete an annotation store**

The following `delete-annotation-store` example deletes an annotation store named `my_vcf_store`.

```
aws omics delete-annotation-store \  
  --name my_vcf_store
```

Output:

```
{  
  "status": "DELETING"  
}
```

For more information, see [Omics Analytics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [DeleteAnnotationStore](#) in *AWS CLI Command Reference*.

delete-reference-store

The following code example shows how to use `delete-reference-store`.

AWS CLI

To delete a reference store

The following `delete-reference-store` example deletes a reference store with ID 1234567890.

```
aws omics delete-reference-store \  
  --id 1234567890
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [DeleteReferenceStore](#) in *AWS CLI Command Reference*.

`delete-reference`

The following code example shows how to use `delete-reference`.

AWS CLI

To delete a reference

The following `delete-reference` example deletes a reference.

```
aws omics delete-reference \  
  --reference-store-id 1234567890 \  
  --id 1234567890
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [DeleteReference](#) in *AWS CLI Command Reference*.

`delete-run-group`

The following code example shows how to use `delete-run-group`.

AWS CLI

To delete a run group

The following `delete-run-group` example deletes a run group with ID 1234567.

```
aws omics delete-run-group \  
  --id 1234567
```

For more information, see [Omics Workflows](#) in the *Amazon Omics Developer Guide*.

- For API details, see [DeleteRunGroup](#) in *AWS CLI Command Reference*.

delete-run

The following code example shows how to use `delete-run`.

AWS CLI

To delete a workflow run

The following `delete-run` example deletes a run with ID 1234567.

```
aws omics delete-run \  
  --id 1234567
```

For more information, see [Omics Workflows](#) in the *Amazon Omics Developer Guide*.

- For API details, see [DeleteRun](#) in *AWS CLI Command Reference*.

delete-sequence-store

The following code example shows how to use `delete-sequence-store`.

AWS CLI

To delete a sequence store

The following `delete-sequence-store` example deletes a sequence store with ID 1234567890.

```
aws omics delete-sequence-store \  
  --id 1234567890
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [DeleteSequenceStore](#) in *AWS CLI Command Reference*.

delete-share

The following code example shows how to use delete-share.

AWS CLI

To delete a share of HealthOmics analytics data

The following delete-share example deletes a cross-account share of analytics data.

```
aws omics delete-share \  
  --share-id "495c21bedc889d07d0ab69d710a6841e-dd75ab7a1a9c384fa848b5bd8e5a7e0a"
```

Output:

```
{  
  "status": "DELETING"  
}
```

For more information, see [Cross-account sharing](#) in the *AWS HealthOmics User Guide*.

- For API details, see [DeleteShare](#) in *AWS CLI Command Reference*.

delete-variant-store

The following code example shows how to use delete-variant-store.

AWS CLI

To delete a variant store

The following delete-variant-store example deletes a variant store named my_var_store.

```
aws omics delete-variant-store \  
  --name my_var_store
```

Output:

```
{  
  "status": "DELETING"  
}
```

```
}
```

For more information, see [Omics Analytics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [DeleteVariantStore](#) in *AWS CLI Command Reference*.

delete-workflow

The following code example shows how to use `delete-workflow`.

AWS CLI

To delete a workflow

The following `delete-workflow` example deletes a workflow with ID 1234567.

```
aws omics delete-workflow \  
  --id 1234567
```

For more information, see [Omics Workflows](#) in the *Amazon Omics Developer Guide*.

- For API details, see [DeleteWorkflow](#) in *AWS CLI Command Reference*.

get-annotation-import-job

The following code example shows how to use `get-annotation-import-job`.

AWS CLI

To view an annotation import job

The following `get-annotation-import-job` example gets details about an annotation import job.

```
aws omics get-annotation-import-job \  
  --job-id 984162c7-xmpl-4d23-ab47-286f7950bfbf
```

Output:

```
{  
  "creationTime": "2022-11-30T01:40:11.017746Z",
```



```

"destinationName": "tsv_ann_store",
"id": "984162c7-xmpl-4d23-ab47-286f7950bfbf",
"items": [
  {
    "jobStatus": "COMPLETED",
    "source": "s3://omics-artifacts-01d6xmpl4e72dd32/targetedregions.bed.gz"
  }
],
"roleArn": "arn:aws:iam::123456789012:role/omics-service-role-serviceRole-
W801XMPL7QZ",
"runLeftNormalization": false,
"status": "COMPLETED",
"updateTime": "2022-11-30T01:42:39.134009Z"
}

```

For more information, see [Omics Analytics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [GetAnnotationImportJob](#) in *AWS CLI Command Reference*.

get-annotation-store-version

The following code example shows how to use `get-annotation-store-version`.

AWS CLI

To retrieve the metadata for an annotation store version

The following `get-annotation-store-version` example retrieves the metadata for the requested annotation store version.

```

aws omics get-annotation-store-version \
  --name my_annotation_store \
  --version-name my_version

```

Output:

```

{
  "storeId": "4934045d1c6d",
  "id": "2a3f4a44aa7b",
  "status": "ACTIVE",
  "versionArn": "arn:aws:omics:us-west-2:555555555555:annotationStore/
my_annotation_store/version/my_version",

```

```
"name": "my_annotation_store",
"versionName": "my_version",
"creationTime": "2023-07-21T17:15:49.251040+00:00",
"updateTime": "2023-07-21T17:15:56.434223+00:00",
"statusMessage": "",
"versionSizeBytes": 0
}
```

For more information, see [Creating new versions of annotation stores](#) in the *AWS HealthOmics User Guide*.

- For API details, see [GetAnnotationStoreVersion](#) in *AWS CLI Command Reference*.

get-annotation-store

The following code example shows how to use `get-annotation-store`.

AWS CLI

To view an annotation store

The following `get-annotation-store` example gets details about an annotation store named `my_ann_store`.

```
aws omics get-annotation-store \
  --name my_ann_store
```

Output:

```
{
  "creationTime": "2022-11-23T22:48:39.226492Z",
  "id": "0a91xmplc71f",
  "name": "my_ann_store",
  "reference": {
    "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890"
  },
  "status": "CREATING",
  "storeArn": "arn:aws:omics:us-west-2:123456789012:annotationStore/my_ann_store",
  "storeFormat": "VCF",
  "storeSizeBytes": 0,
  "tags": {}
}
```

```
}
```

For more information, see [Omics Analytics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [GetAnnotationStore](#) in *AWS CLI Command Reference*.

get-read-set-activation-job

The following code example shows how to use `get-read-set-activation-job`.

AWS CLI

To view a read set activation job

The following `get-read-set-activation-job` example gets details about a read set activation job.

```
aws omics get-read-set-activation-job \  
  --sequence-store-id 1234567890 \  
  --id 1234567890
```

Output:

```
{  
  "completionTime": "2022-12-06T22:33:42.828Z",  
  "creationTime": "2022-12-06T22:32:45.213Z",  
  "id": "1234567890",  
  "sequenceStoreId": "1234567890",  
  "sources": [  
    {  
      "readSetId": "1234567890",  
      "status": "FINISHED",  
      "statusMessage": "No activation needed as read set is already in  
ACTIVATING or ACTIVE state."  
    }  
  ],  
  "status": "COMPLETED",  
  "statusMessage": "The job completed successfully."  
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [GetReadSetActivationJob](#) in *AWS CLI Command Reference*.

get-read-set-export-job

The following code example shows how to use `get-read-set-export-job`.

AWS CLI

To view a read set export job

The following `get-read-set-export-job` example gets details about a read set export job.

```
aws omics get-read-set-export-job \  
  --sequence-store-id 1234567890 \  
  --id 1234567890
```

Output:

```
{  
  "completionTime": "2022-12-06T22:39:14.491Z",  
  "creationTime": "2022-12-06T22:37:18.612Z",  
  "destination": "s3://omics-artifacts-01d6xmpl4e72dd32/read-set-export/",  
  "id": "1234567890",  
  "sequenceStoreId": "1234567890",  
  "status": "COMPLETED",  
  "statusMessage": "The job is submitted and will start soon."  
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [GetReadSetExportJob](#) in *AWS CLI Command Reference*.

get-read-set-import-job

The following code example shows how to use `get-read-set-import-job`.

AWS CLI

To view a read set import job

The following `get-read-set-import-job` example gets details about a read set import job.

```
aws omics get-read-set-import-job \  
  --sequence-store-id 1234567890 \  
  --id 1234567890
```

Output:

```

{
  "creationTime": "2022-11-23T01:36:38.158Z",
  "id": "1234567890",
  "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-serviceRole-
W801XMPL7QZ",
  "sequenceStoreId": "1234567890",
  "sources": [
    {
      "name": "HG00100",
      "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890",
      "sampleId": "bam-sample",
      "sourceFileType": "BAM",
      "sourceFiles": {
        "source1": "s3://omics-artifacts-01d6xmpl4e72dd32/
HG00100.chrom20.ILLUMINA.bwa.GBR.low_coverage.20101123.bam",
        "source2": ""
      },
      "status": "IN_PROGRESS",
      "statusMessage": "The source job is currently in progress.",
      "subjectId": "bam-subject",
      "tags": {
        "aws:omics:sampleId": "bam-sample",
        "aws:omics:subjectId": "bam-subject"
      }
    },
    {
      "name": "HG00146",
      "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890",
      "sampleId": "fastq-sample",
      "sourceFileType": "FASTQ",
      "sourceFiles": {
        "source1": "s3://omics-artifacts-01d6xmpl4e72dd32/
SRR233106_1.filt.fastq.gz",
        "source2": "s3://omics-artifacts-01d6xmpl4e72dd32/
SRR233106_2.filt.fastq.gz"
      },
      "status": "IN_PROGRESS",
      "statusMessage": "The source job is currently in progress.",
      "subjectId": "fastq-subject",
      "tags": {

```

```

        "aws:omics:sampleId": "fastq-sample",
        "aws:omics:subjectId": "fastq-subject"
    }
},
{
    "name": "HG00096",
    "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890",
    "sampleId": "cram-sample",
    "sourceFileType": "CRAM",
    "sourceFiles": {
        "source1": "s3://omics-artifacts-01d6xmpl4e72dd32/
HG00096.alt_bwamem_GRCh38DH.20150718.GBR.low_coverage.cram",
        "source2": ""
    },
    "status": "IN_PROGRESS",
    "statusMessage": "The source job is currently in progress.",
    "subjectId": "cram-subject",
    "tags": {
        "aws:omics:sampleId": "cram-sample",
        "aws:omics:subjectId": "cram-subject"
    }
}
],
"status": "IN_PROGRESS",
"statusMessage": "The job is currently in progress."
}

```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [GetReadSetImportJob](#) in *AWS CLI Command Reference*.

get-read-set-metadata

The following code example shows how to use `get-read-set-metadata`.

AWS CLI

To view a read set

The following `get-read-set-metadata` example gets details about a read set's files.

```

aws omics get-read-set-metadata \
  --sequence-store-id 1234567890 \

```

```
--id 1234567890
```

Output:

```
{
  "arn": "arn:aws:omics:us-west-2:123456789012:sequenceStore/1234567890/
readSet/1234567890",
  "creationTime": "2022-11-23T21:55:00.515Z",
  "fileType": "FASTQ",
  "files": {
    "source1": {
      "contentLength": 310054739,
      "partSize": 104857600,
      "totalParts": 3
    },
    "source2": {
      "contentLength": 307846621,
      "partSize": 104857600,
      "totalParts": 3
    }
  },
  "id": "1234567890",
  "name": "HG00146",
  "referenceArn": "arn:aws:omics:us-west-2:123456789012:referenceStore/1234567890/
reference/1234567890",
  "sampleId": "fastq-sample",
  "sequenceInformation": {
    "alignment": "UNALIGNED",
    "totalBaseCount": 677717384,
    "totalReadCount": 8917334
  },
  "sequenceStoreId": "1234567890",
  "status": "ACTIVE",
  "subjectId": "fastq-subject"
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [GetReadSetMetadata](#) in *AWS CLI Command Reference*.

get-read-set

The following code example shows how to use `get-read-set`.

AWS CLI

To download a read set

The following `get-read-set` example downloads part 3 of a read set as `1234567890.3.bam`.

```
aws omics get-read-set \  
  --sequence-store-id 1234567890 \  
  --id 1234567890 \  
  --part-number 3 1234567890.3.bam
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [GetReadSet](#) in *AWS CLI Command Reference*.

get-reference-import-job

The following code example shows how to use `get-reference-import-job`.

AWS CLI

To view a reference import job

The following `get-reference-import-job` example gets details about a reference import job.

```
aws omics get-reference-import-job \  
  --reference-store-id 1234567890 \  
  --id 1234567890
```

Output:

```
{  
  "creationTime": "2022-11-22T22:25:41.124Z",  
  "id": "1234567890",  
  "referenceStoreId": "1234567890",  
  "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-serviceRole-  
W801XMPL7QZ",  
  "sources": [  
    {  
      "name": "assembly-38",  
      "sourceFile": "s3://omics-artifacts-01d6xmpl4e72dd32/  
Homo_sapiens_assembly38.fasta",
```



```
        "status": "IN_PROGRESS",
        "statusMessage": "The source job is currently in progress."
    }
  ],
  "status": "IN_PROGRESS",
  "statusMessage": "The job is currently in progress."
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [GetReferenceImportJob](#) in *AWS CLI Command Reference*.

get-reference-metadata

The following code example shows how to use `get-reference-metadata`.

AWS CLI

To view a reference

The following `get-reference-metadata` example gets details about a reference.

```
aws omics get-reference-metadata \
  --reference-store-id 1234567890 \
  --id 1234567890
```

Output:

```
{
  "arn": "arn:aws:omics:us-west-2:123456789012:referenceStore/1234567890/
reference/1234567890",
  "creationTime": "2022-11-22T22:27:09.033Z",
  "files": {
    "index": {
      "contentLength": 160928,
      "partSize": 104857600,
      "totalParts": 1
    },
    "source": {
      "contentLength": 3249912778,
      "partSize": 104857600,
      "totalParts": 31
    }
  }
}
```

```
  },
  "id": "1234567890",
  "md5": "7ff134953dcca8c8997453bbb80b6b5e",
  "name": "assembly-38",
  "referenceStoreId": "1234567890",
  "status": "ACTIVE",
  "updateTime": "2022-11-22T22:27:09.033Z"
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [GetReferenceMetadata](#) in *AWS CLI Command Reference*.

get-reference-store

The following code example shows how to use `get-reference-store`.

AWS CLI

To view a reference store

The following `get-reference-store` example gets details about a reference store.

```
aws omics get-reference-store \
  --id 1234567890
```

Output:

```
{
  "arn": "arn:aws:omics:us-west-2:123456789012:referenceStore/1234567890",
  "creationTime": "2022-09-23T23:27:20.364Z",
  "id": "1234567890",
  "name": "my-rstore-0"
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [GetReferenceStore](#) in *AWS CLI Command Reference*.

get-reference

The following code example shows how to use `get-reference`.

AWS CLI

To download a genome reference

The following `get-reference` example downloads part 1 of a genome as `hg38.1.fa`.

```
aws omics get-reference \  
  --reference-store-id 1234567890 \  
  --id 1234567890 \  
  --part-number 1 hg38.1.fa
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [GetReference](#) in *AWS CLI Command Reference*.

get-run-group

The following code example shows how to use `get-run-group`.

AWS CLI

To view a run group

The following `get-run-group` example gets details about a run group.

```
aws omics get-run-group \  
  --id 1234567
```

Output:

```
{  
  "arn": "arn:aws:omics:us-west-2:123456789012:runGroup/1234567",  
  "creationTime": "2022-12-01T00:58:42.915219Z",  
  "id": "1234567",  
  "maxCpus": 20,  
  "maxDuration": 600,  
  "name": "cram-convert",  
  "tags": {}  
}
```

For more information, see [Omics Workflows](#) in the *Amazon Omics Developer Guide*.

- For API details, see [GetRunGroup](#) in *AWS CLI Command Reference*.

get-run-task

The following code example shows how to use `get-run-task`.

AWS CLI

To view a task

The following `get-run-task` example gets details about a workflow task.

```
aws omics get-run-task \  
  --id 1234567 \  
  --task-id 1234567
```

Output:

```
{  
  "cpus": 1,  
  "creationTime": "2022-11-30T23:13:00.718651Z",  
  "logStream": "arn:aws:logs:us-west-2:123456789012:log-group:/aws/omics/  
WorkflowLog:log-stream:run/1234567/task/1234567",  
  "memory": 15,  
  "name": "CramToBamTask",  
  "startTime": "2022-11-30T23:17:47.016Z",  
  "status": "COMPLETED",  
  "stopTime": "2022-11-30T23:18:21.503Z",  
  "taskId": "1234567"  
}
```

For more information, see [Omics Workflows](#) in the *Amazon Omics Developer Guide*.

- For API details, see [GetRunTask](#) in *AWS CLI Command Reference*.

get-run

The following code example shows how to use `get-run`.

AWS CLI

To view a workflow run

The following `get-run` example gets details about a workflow run.

```
aws omics get-run \  
  --id 1234567
```

Output:

```
{  
  "arn": "arn:aws:omics:us-west-2:123456789012:run/1234567",  
  "creationTime": "2022-11-30T22:58:22.615865Z",  
  "digest":  
    "sha256:c54bxmpl742dcc26f7fa1f10e37550ddd8f251f418277c0a58e895b801ed28cf",  
    "id": "1234567",  
    "name": "cram-to-bam",  
    "outputUri": "s3://omics-artifacts-01d6xmpl4e72dd32/workflow-output/",  
    "parameters": {  
      "ref_dict": "s3://omics-artifacts-01d6xmpl4e72dd32/  
Homo_sapiens_assembly38.dict",  
      "ref_fasta_index": "s3://omics-artifacts-01d6xmpl4e72dd32/  
Homo_sapiens_assembly38.fasta.fai",  
      "ref_fasta": "s3://omics-artifacts-01d6xmpl4e72dd32/  
Homo_sapiens_assembly38.fasta",  
      "sample_name": "NA12878",  
      "input_cram": "s3://omics-artifacts-01d6xmpl4e72dd32/NA12878.cram"  
    },  
    "resourceDigests": {  
      "s3://omics-artifacts-01d6xmpl4e72dd32/Homo_sapiens_assembly38.fasta.fai":  
"etag:f76371b113734a56cde236bc0372de0a",  
      "s3://omics-artifacts-01d6xmpl4e72dd32/Homo_sapiens_assembly38.dict":  
"etag:3884c62eb0e53fa92459ed9bfff133ae6",  
      "s3://omics-artifacts-01d6xmpl4e72dd32/Homo_sapiens_assembly38.fasta":  
"etag:e307d81c605fb91b7720a08f00276842-388",  
      "s3://omics-artifacts-01d6xmpl4e72dd32/NA12878.cram":  
"etag:a9f52976381286c6143b5cc681671ec6"  
    },  
    "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-serviceRole-  
w801XMPL7QZ",  
    "startedBy": "arn:aws:iam::123456789012:user/laptop-2020",  
    "status": "STARTING",  
    "tags": {},  
    "workflowId": "1234567",  
    "workflowType": "PRIVATE"  
}
```

For more information, see [Omics Workflows](#) in the *Amazon Omics Developer Guide*.

- For API details, see [GetRun](#) in *AWS CLI Command Reference*.

get-sequence-store

The following code example shows how to use `get-sequence-store`.

AWS CLI

To view a sequence store

The following `get-sequence-store` example gets details about a sequence store with ID 1234567890.

```
aws omics get-sequence-store \  
  --id 1234567890
```

Output:

```
{  
  "arn": "arn:aws:omics:us-east-1:123456789012:sequenceStore/1234567890",  
  "creationTime": "2022-11-23T19:55:48.376Z",  
  "id": "1234567890",  
  "name": "my-seq-store"  
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [GetSequenceStore](#) in *AWS CLI Command Reference*.

get-share

The following code example shows how to use `get-share`.

AWS CLI

To retrieve the metadata about a share of a HealthOmics analytics data

The following `get-share` example retrieves the metadata for a cross-account share of analytics data.

```
aws omics get-share \  
  --share-id "495c21bedc889d07d0ab69d710a6841e-dd75ab7a1a9c384fa848b5bd8e5a7e0a"
```

Output:

```
{  
  "share": {  
    "shareId": "495c21bedc889d07d0ab69d710a6841e-dd75ab7a1a9c384fa848b5bd8e5a7e0a",  
    "name": "my_Share-123",  
    "resourceArn": "arn:aws:omics:us-west-2:555555555555:variantStore/omics_dev_var_store",  
    "principalSubscriber": "123456789012",  
    "ownerId": "555555555555",  
    "status": "PENDING"  
  }  
}
```

For more information, see [Cross-account sharing](#) in the *AWS HealthOmics User Guide*.

- For API details, see [GetShare](#) in *AWS CLI Command Reference*.

get-variant-import-job

The following code example shows how to use `get-variant-import-job`.

AWS CLI

To view a variant import job

The following `get-variant-import-job` example gets details about a variant import job.

```
aws omics get-variant-import-job \  
  --job-id edd7b8ce-xmpl-47e2-bc99-258cac95a508
```

Output:

```
{  
  "creationTime": "2022-11-23T22:42:50.037812Z",  
  "destinationName": "my_var_store",  
  "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",  
  "items": [  
    {  
      "creationTime": "2022-11-23T22:42:50.037812Z",  
      "destinationName": "my_var_store",  
      "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",  
      "items": [  
        {  
          "creationTime": "2022-11-23T22:42:50.037812Z",  
          "destinationName": "my_var_store",  
          "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",  
          "items": [  
            {  
              "creationTime": "2022-11-23T22:42:50.037812Z",  
              "destinationName": "my_var_store",  
              "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",  
              "items": [  
                {  
                  "creationTime": "2022-11-23T22:42:50.037812Z",  
                  "destinationName": "my_var_store",  
                  "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",  
                  "items": [  
                    {  
                      "creationTime": "2022-11-23T22:42:50.037812Z",  
                      "destinationName": "my_var_store",  
                      "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",  
                      "items": [  
                        {  
                          "creationTime": "2022-11-23T22:42:50.037812Z",  
                          "destinationName": "my_var_store",  
                          "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",  
                          "items": [  
                            {  
                              "creationTime": "2022-11-23T22:42:50.037812Z",  
                              "destinationName": "my_var_store",  
                              "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",  
                              "items": [  
                                {  
                                  "creationTime": "2022-11-23T22:42:50.037812Z",  
                                  "destinationName": "my_var_store",  
                                  "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",  
                                  "items": [  
                                    {  
                                      "creationTime": "2022-11-23T22:42:50.037812Z",  
                                      "destinationName": "my_var_store",  
                                      "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",  
                                      "items": [  
                                        {  
                                          "creationTime": "2022-11-23T22:42:50.037812Z",  
                                          "destinationName": "my_var_store",  
                                          "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",  
                                          "items": [  
                                            {  
                                              "creationTime": "2022-11-23T22:42:50.037812Z",  
                                              "destinationName": "my_var_store",  
                                              "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",  
                                              "items": [  
                                                {  
                                                  "creationTime": "2022-11-23T22:42:50.037812Z",  
                                                  "destinationName": "my_var_store",  
                                                  "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",  
                                                  "items": [  
                                                    {  
                                                      "creationTime": "2022-11-23T22:42:50.037812Z",  
                                                      "destinationName": "my_var_store",  
                                                      "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",  
                                                      "items": [  
                                                        {  
                                                          "creationTime": "2022-11-23T22:42:50.037812Z",  
                                                          "destinationName": "my_var_store",  
                                                          "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",  
                                                          "items": [  
                                                            {  
                                                              "creationTime": "2022-11-23T22:42:50.037812Z",  
                                                              "destinationName": "my_var_store",  
                                                              "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",  
                                                              "items": [  
                                                                {  
                                                                  "creationTime": "2022-11-23T22:42:50.037812Z",  
                                                                  "destinationName": "my_var_store",  
                                                                  "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",  
                                                                  "items": [  
                                                                    {  
                                                                      "creationTime": "2022-11-23T22:42:50.037812Z",  
                                                                      "destinationName": "my_var_store",  
                                                                      "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",  
                                                                      "items": [  
                                                                      ]  
                                                                    ]  
                                                                  ]  
                                                                ]  
                                                              ]  
                                                            ]  
                                                          ]  
                                                        ]  
                                                      ]  
                                                    ]  
                                                  ]  
                                                ]  
                                              ]  
                                            ]  
                                          ]  
                                        ]  
                                      ]  
                                    ]  
                                  ]  
                                ]  
                              ]  
                            ]  
                          ]  
                        ]  
                      ]  
                    ]  
                  ]  
                ]  
              ]  
            ]  
          ]  
        ]  
      ]  
    ]  
  ]  
}
```

```

    {
      "jobStatus": "IN_PROGRESS",
      "source": "s3://omics-artifacts-01d6xmpl4e72dd32/
Homo_sapiens_assembly38.known_indels.vcf.gz"
    }
  ],
  "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-serviceRole-
W801XMPL7QZ",
  "runLeftNormalization": false,
  "status": "IN_PROGRESS",
  "updateTime": "2022-11-23T22:43:05.898309Z"
}

```

For more information, see [Omics Analytics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [GetVariantImportJob](#) in *AWS CLI Command Reference*.

get-variant-store

The following code example shows how to use `get-variant-store`.

AWS CLI

To view a variant store

The following `get-variant-store` example gets details about a variant store.

```
aws omics get-variant-store \
  --name my_var_store
```

Output:

```

{
  "creationTime": "2022-11-23T22:09:07.534499Z",
  "id": "02dexmplcfdd",
  "name": "my_var_store",
  "reference": {
    "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890"
  },
  "status": "CREATING",
  "storeArn": "arn:aws:omics:us-west-2:123456789012:variantStore/my_var_store",
  "storeSizeBytes": 0,

```



```
"tags": {},
"updateTime": "2022-11-23T22:09:24.931711Z"
}
```

For more information, see [Omics Analytics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [GetVariantStore](#) in *AWS CLI Command Reference*.

get-workflow

The following code example shows how to use `get-workflow`.

AWS CLI

To view a workflow

The following `get-workflow` example gets details about a workflow with ID 1234567.

```
aws omics get-workflow \
  --id 1234567
```

Output:

```
{
  "arn": "arn:aws:omics:us-west-2:123456789012:workflow/1234567",
  "creationTime": "2022-11-30T22:33:16.225368Z",
  "digest":
  "sha256:c54bxmpl742dcc26f7fa1f10e37550ddd8f251f418277c0a58e895b801ed28cf",
  "engine": "WDL",
  "id": "1234567",
  "main": "workflow-crambam.wdl",
  "name": "cram-converter",
  "parameterTemplate": {
    "ref_dict": {
      "description": "dictionary file for 'ref_fasta'"
    },
    "ref_fasta_index": {
      "description": "Index of the reference genome fasta file"
    },
    "ref_fasta": {
      "description": "Reference genome fasta file"
    },
    "input_cram": {
```

```

        "description": "The Cram file to convert to BAM"
    },
    "sample_name": {
        "description": "The name of the input sample, used to name the output
BAM"
    }
},
"status": "ACTIVE",
"statusMessage": "workflow-crambam.wdl\n    workflow CramToBamFlow\n
call CramToBamTask\n        call ValidateSamFile\n    task CramToBamTask\n    task
ValidateSamFile\n",
"tags": {},
"type": "PRIVATE"
}

```

For more information, see [Omics Workflows](#) in the *Amazon Omics Developer Guide*.

- For API details, see [GetWorkflow](#) in *AWS CLI Command Reference*.

list-annotation-import-jobs

The following code example shows how to use `list-annotation-import-jobs`.

AWS CLI

To get a list of annotation import jobs

The following `list-annotation-import-jobs` gets a list of annotation import jobs.

```
aws omics list-annotation-import-jobs
```

Output:

```

{
  "annotationImportJobs": [
    {
      "creationTime": "2022-11-30T01:39:41.478294Z",
      "destinationName": "gff_ann_store",
      "id": "18a9e792-xmpl-4869-a105-e5b602900444",
      "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-
serviceRole-W801XMPL7QZ",
      "runLeftNormalization": false,
      "status": "COMPLETED",
    }
  ]
}

```

```
        "updateTime": "2022-11-30T01:47:09.145178Z"
    },
    {
        "creationTime": "2022-11-30T00:45:58.007838Z",
        "destinationName": "my_ann_store",
        "id": "4e9eafc8-xmpl-431e-a0b2-3bda27cb600a",
        "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-
serviceRole-W801XMPL7QZ",
        "runLeftNormalization": false,
        "status": "FAILED",
        "updateTime": "2022-11-30T00:47:01.706325Z"
    }
]
}
```

For more information, see [Omics Analytics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [ListAnnotationImportJobs](#) in *AWS CLI Command Reference*.

list-annotation-store-versions

The following code example shows how to use `list-annotation-store-versions`.

AWS CLI

To list all the versions of an annotation store.

The following `list-annotation-store-versions` example lists all versions that exist of an annotation store.

```
aws omics list-annotation-store-versions \
  --name my_annotation_store
```

Output:

```
{
  "annotationStoreVersions": [
    {
      "storeId": "4934045d1c6d",
      "id": "2a3f4a44aa7b",
      "status": "CREATING",
      "versionArn": "arn:aws:omics:us-west-2:555555555555:annotationStore/
my_annotation_store/version/my_version_2",
```

```
    "name": "my_annotation_store",
    "versionName": "my_version_2",
    "creationTime": "2023-07-21T17:20:59.380043+00:00",
    "versionSizeBytes": 0
  },
  {
    "storeId": "4934045d1c6d",
    "id": "4934045d1c6d",
    "status": "ACTIVE",
    "versionArn": "arn:aws:omics:us-west-2:555555555555:annotationStore/
my_annotation_store/version/my_version_1",
    "name": "my_annotation_store",
    "versionName": "my_version_1",
    "creationTime": "2023-07-21T17:15:49.251040+00:00",
    "updateTime": "2023-07-21T17:15:56.434223+00:00",
    "statusMessage": "",
    "versionSizeBytes": 0
  }
}
```

For more information, see [Creating new versions of annotation stores](#) in the *AWS HealthOmics User Guide*.

- For API details, see [ListAnnotationStoreVersions](#) in *AWS CLI Command Reference*.

list-annotation-stores

The following code example shows how to use `list-annotation-stores`.

AWS CLI

To get a list of annotation stores

The following `list-annotation-stores` example gets a list of annotation stores.

```
aws omics list-annotation-stores
```

Output:

```
{
  "annotationStores": [
```

```
{
  "creationTime": "2022-11-23T22:48:39.226492Z",
  "id": "0a91xmplc71f",
  "name": "my_ann_store",
  "reference": {
    "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890"
  },
  "status": "ACTIVE",
  "statusMessage": "",
  "storeArn": "arn:aws:omics:us-west-2:123456789012:annotationStore/
my_ann_store",
  "storeFormat": "VCF",
  "storeSizeBytes": 0,
  "updateTime": "2022-11-23T22:53:27.372840Z"
}
]
```

For more information, see [Omics Analytics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [ListAnnotationStores](#) in *AWS CLI Command Reference*.

list-multipart-read-set-uploads

The following code example shows how to use `list-multipart-read-set-uploads`.

AWS CLI

To list all multipart read set uploads and their statuses.

The following `list-multipart-read-set-uploads` example lists all multipart read set uploads and their statuses.

```
aws omics list-multipart-read-set-uploads \
  --sequence-store-id 0123456789
```

Output:

```
{
  "uploads":
  [
```

```
{
  "sequenceStoreId": "0123456789",
  "uploadId": "8749584421",
  "sourceFileType": "FASTQ",
  "subjectId": "mySubject",
  "sampleId": "mySample",
  "generatedFrom": "1000 Genomes",
  "name": "HG00146",
  "description": "FASTQ for HG00146",
  "creationTime": "2023-11-29T19:22:51.349298+00:00"
},
{
  "sequenceStoreId": "0123456789",
  "uploadId": "5290538638",
  "sourceFileType": "BAM",
  "subjectId": "mySubject",
  "sampleId": "mySample",
  "generatedFrom": "1000 Genomes",
  "referenceArn": "arn:aws:omics:us-
west-2:845448930428:referenceStore/8168613728/reference/2190697383",
  "name": "HG00146",
  "description": "BAM for HG00146",
  "creationTime": "2023-11-29T19:23:33.116516+00:00"
},
{
  "sequenceStoreId": "0123456789",
  "uploadId": "4174220862",
  "sourceFileType": "BAM",
  "subjectId": "mySubject",
  "sampleId": "mySample",
  "generatedFrom": "1000 Genomes",
  "referenceArn": "arn:aws:omics:us-
west-2:845448930428:referenceStore/8168613728/reference/2190697383",
  "name": "HG00147",
  "description": "BAM for HG00147",
  "creationTime": "2023-11-29T19:23:47.007866+00:00"
}
]
```

For more information, see [Direct upload to a sequence store](#) in the *AWS HealthOmics User Guide*.

- For API details, see [ListMultipartReadSetUploads](#) in *AWS CLI Command Reference*.

list-read-set-activation-jobs

The following code example shows how to use `list-read-set-activation-jobs`.

AWS CLI

To get a list of read set activation jobs

The following `list-read-set-activation-jobs` example gets a list of activation jobs for a sequence store with id 1234567890.

```
aws omics list-read-set-activation-jobs \  
  --sequence-store-id 1234567890
```

Output:

```
{  
  "activationJobs": [  
    {  
      "completionTime": "2022-12-06T22:33:42.828Z",  
      "creationTime": "2022-12-06T22:32:45.213Z",  
      "id": "1234567890",  
      "sequenceStoreId": "1234567890",  
      "status": "COMPLETED"  
    },  
    {  
      "creationTime": "2022-12-06T22:35:10.100Z",  
      "id": "1234567890",  
      "sequenceStoreId": "1234567890",  
      "status": "IN_PROGRESS"  
    }  
  ]  
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [ListReadSetActivationJobs](#) in *AWS CLI Command Reference*.

list-read-set-export-jobs

The following code example shows how to use `list-read-set-export-jobs`.

AWS CLI

To gets a list of read set export jobs

The following `list-read-set-export-jobs` example gets a list of export jobs for a sequence store with id 1234567890.

```
aws omics list-read-set-export-jobs \  
  --sequence-store-id 1234567890
```

Output:

```
{  
  "exportJobs": [  
    {  
      "completionTime": "2022-12-06T22:39:14.491Z",  
      "creationTime": "2022-12-06T22:37:18.612Z",  
      "destination": "s3://omics-artifacts-01d6xmpl4e72dd32/read-set-export/",  
      "id": "1234567890",  
      "sequenceStoreId": "1234567890",  
      "status": "COMPLETED"  
    },  
    {  
      "creationTime": "2022-12-06T22:38:04.871Z",  
      "destination": "s3://omics-artifacts-01d6xmpl4e72dd32/read-set-export/",  
      "id": "1234567890",  
      "sequenceStoreId": "1234567890",  
      "status": "IN_PROGRESS"  
    }  
  ]  
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [ListReadSetExportJobs](#) in *AWS CLI Command Reference*.

list-read-set-import-jobs

The following code example shows how to use `list-read-set-import-jobs`.

AWS CLI

To get a list of read set import jobs

The following `list-read-set-import-jobs` example gets a list of import jobs for a sequence store with id 1234567890.

```
aws omics list-read-set-import-jobs \  
  --sequence-store-id 1234567890
```

Output:

```
{  
  "importJobs": [  
    {  
      "completionTime": "2022-11-29T18:17:49.244Z",  
      "creationTime": "2022-11-29T17:32:47.700Z",  
      "id": "1234567890",  
      "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-  
serviceRole-W801XMPL7QZ",  
      "sequenceStoreId": "1234567890",  
      "status": "COMPLETED"  
    },  
    {  
      "completionTime": "2022-11-23T22:01:34.090Z",  
      "creationTime": "2022-11-23T21:52:43.289Z",  
      "id": "1234567890",  
      "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-  
serviceRole-W801XMPL7QZ",  
      "sequenceStoreId": "1234567890",  
      "status": "COMPLETED_WITH_FAILURES"  
    }  
  ]  
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [ListReadSetImportJobs](#) in *AWS CLI Command Reference*.

list-read-set-upload-parts

The following code example shows how to use `list-read-set-upload-parts`.

AWS CLI

To list all parts in a requested multipart upload for a sequence store.

The following `list-read-set-upload-parts` example list all parts in a requested multipart upload for a sequence store.

```
aws omics list-read-set-upload-parts \  
  --sequence-store-id 0123456789 \  
  --upload-id 1122334455 \  
  --part-source SOURCE1
```

Output:

```
{  
  "parts": [  
    {  
      "partNumber": 1,  
      "partSize": 94371840,  
      "file": "SOURCE1",  
      "checksum":  
      "984979b9928ae8d8622286c4a9cd8e99d964a22d59ed0f5722e1733eb280e635",  
      "lastUpdatedTime": "2023-02-02T20:14:47.533000+00:00"  
    },  
    {  
      "partNumber": 2,  
      "partSize": 10471840,  
      "file": "SOURCE1",  
      "checksum":  
      "984979b9928ae8d8622286c4a9cd8e99d964a22d59ed0f5722e1733eb280e635",  
      "lastUpdatedTime": "2023-02-02T20:14:47.533000+00:00"  
    }  
  ]  
}
```

For more information, see [Direct upload to a sequence store](#) in the *AWS HealthOmics User Guide*.

- For API details, see [ListReadSetUploadParts](#) in *AWS CLI Command Reference*.

list-read-sets

The following code example shows how to use `list-read-sets`.

AWS CLI

To get a list of read sets

The following `list-read-sets` example gets a list of read sets for a sequence store with id 1234567890.

```
aws omics list-read-sets \  
  --sequence-store-id 1234567890
```

Output:

```
{  
  "readSets": [  
    {  
      "arn": "arn:aws:omics:us-west-2:123456789012:sequenceStore/1234567890/  
readSet/1234567890",  
      "creationTime": "2022-11-23T21:55:00.515Z",  
      "fileType": "FASTQ",  
      "id": "1234567890",  
      "name": "HG00146",  
      "referenceArn": "arn:aws:omics:us-  
west-2:123456789012:referenceStore/1234567890/reference/1234567890",  
      "sampleId": "fastq-sample",  
      "sequenceStoreId": "1234567890",  
      "status": "ACTIVE",  
      "subjectId": "fastq-subject"  
    }  
  ]  
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [ListReadSets](#) in *AWS CLI Command Reference*.

list-reference-import-jobs

The following code example shows how to use `list-reference-import-jobs`.

AWS CLI

To get a list of reference import jobs

The following `list-reference-import-jobs` example gets a list of reference import jobs for a reference store with id 1234567890.

```
aws omics list-reference-import-jobs \  
  --reference-store-id 1234567890
```

Output:

```
{  
  "importJobs": [  
    {  
      "completionTime": "2022-11-23T19:54:58.204Z",  
      "creationTime": "2022-11-23T19:53:20.729Z",  
      "id": "1234567890",  
      "referenceStoreId": "1234567890",  
      "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-  
serviceRole-W801XMPL7QZ",  
      "status": "COMPLETED"  
    },  
    {  
      "creationTime": "2022-11-23T20:34:03.250Z",  
      "id": "1234567890",  
      "referenceStoreId": "1234567890",  
      "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-  
serviceRole-W801XMPL7QZ",  
      "status": "IN_PROGRESS"  
    }  
  ]  
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [ListReferenceImportJobs](#) in *AWS CLI Command Reference*.

list-reference-stores

The following code example shows how to use `list-reference-stores`.

AWS CLI

To get a list of reference stores

The following `list-reference-stores` example gets a list of reference stores.

```
aws omics list-reference-stores
```

Output:

```
{
  "referenceStores": [
    {
      "arn": "arn:aws:omics:us-west-2:123456789012:referenceStore/1234567890",
      "creationTime": "2022-11-22T22:13:25.947Z",
      "id": "1234567890",
      "name": "my-ref-store"
    }
  ]
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [ListReferenceStores](#) in *AWS CLI Command Reference*.

list-references

The following code example shows how to use `list-references`.

AWS CLI

To get a list of references

The following `list-references` example gets a list of genome references for a reference store with id 1234567890.

```
aws omics list-references \
  --reference-store-id 1234567890
```

Output:

```
{
  "references": [
    {
      "arn": "arn:aws:omics:us-west-2:123456789012:referenceStore/1234567890/
reference/1234567890",
```

```
    "creationTime": "2022-11-22T22:27:09.033Z",
    "id": "1234567890",
    "md5": "7ff134953dcca8c8997453bbb80b6b5e",
    "name": "assembly-38",
    "referenceStoreId": "1234567890",
    "status": "ACTIVE",
    "updateTime": "2022-11-22T22:27:09.033Z"
  }
]
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [ListReferences](#) in *AWS CLI Command Reference*.

list-run-groups

The following code example shows how to use `list-run-groups`.

AWS CLI

To get a list of run groups

The following `list-run-groups` example gets a list of run groups.

```
aws omics list-run-groups
```

Output:

```
{
  "items": [
    {
      "arn": "arn:aws:omics:us-west-2:123456789012:runGroup/1234567",
      "creationTime": "2022-12-01T00:58:42.915219Z",
      "id": "1234567",
      "maxCpus": 20,
      "maxDuration": 600,
      "name": "cram-convert"
    }
  ]
}
```

For more information, see [Omics Workflows](#) in the *Amazon Omics Developer Guide*.

- For API details, see [ListRunGroups](#) in *AWS CLI Command Reference*.

list-run-tasks

The following code example shows how to use `list-run-tasks`.

AWS CLI

To get a list of tasks

The following `list-run-tasks` example gets a list of tasks for a workflow run.

```
aws omics list-run-tasks \  
  --id 1234567
```

Output:

```
{  
  "items": [  
    {  
      "cpus": 1,  
      "creationTime": "2022-11-30T23:13:00.718651Z",  
      "memory": 15,  
      "name": "CramToBamTask",  
      "startTime": "2022-11-30T23:17:47.016Z",  
      "status": "COMPLETED",  
      "stopTime": "2022-11-30T23:18:21.503Z",  
      "taskId": "1234567"  
    },  
    {  
      "cpus": 1,  
      "creationTime": "2022-11-30T23:18:32.315606Z",  
      "memory": 4,  
      "name": "ValidateSamFile",  
      "startTime": "2022-11-30T23:23:40.165Z",  
      "status": "COMPLETED",  
      "stopTime": "2022-11-30T23:24:14.766Z",  
      "taskId": "1234567"  
    }  
  ]  
}
```

```
}
```

For more information, see [Omics Workflows](#) in the *Amazon Omics Developer Guide*.

- For API details, see [ListRunTasks](#) in *AWS CLI Command Reference*.

list-runs

The following code example shows how to use `list-runs`.

AWS CLI

To get a list of workflow runs

The following `list-runs` example gets a list of workflow runs.

```
aws omics list-runs
```

Output:

```
{
  "items": [
    {
      "arn": "arn:aws:omics:us-west-2:123456789012:run/1234567",
      "creationTime": "2022-12-02T23:20:01.202074Z",
      "id": "1234567",
      "name": "cram-to-bam",
      "priority": 1,
      "startTime": "2022-12-02T23:29:18.115Z",
      "status": "COMPLETED",
      "stopTime": "2022-12-02T23:57:54.428812Z",
      "storageCapacity": 10,
      "workflowId": "1234567"
    },
    {
      "arn": "arn:aws:omics:us-west-2:123456789012:run/1234567",
      "creationTime": "2022-12-03T00:16:57.180066Z",
      "id": "1234567",
      "name": "cram-to-bam",
      "priority": 1,
      "startTime": "2022-12-03T00:26:50.233Z",
      "status": "FAILED",

```



```
    "stopTime": "2022-12-03T00:37:21.451340Z",
    "storageCapacity": 10,
    "workflowId": "1234567"
  },
  {
    "arn": "arn:aws:omics:us-west-2:123456789012:run/1234567",
    "creationTime": "2022-12-05T17:57:08.444817Z",
    "id": "1234567",
    "name": "cram-to-bam",
    "status": "STARTING",
    "workflowId": "1234567"
  }
]
```

For more information, see [Omics Workflows](#) in the *Amazon Omics Developer Guide*.

- For API details, see [ListRuns](#) in *AWS CLI Command Reference*.

list-sequence-stores

The following code example shows how to use `list-sequence-stores`.

AWS CLI

To get a list of sequence stores

The following `list-sequence-stores` example gets a list of sequence stores.

```
aws omics list-sequence-stores
```

Output:

```
{
  "sequenceStores": [
    {
      "arn": "arn:aws:omics:us-west-2:123456789012:sequenceStore/1234567890",
      "creationTime": "2022-11-23T01:24:33.629Z",
      "id": "1234567890",
      "name": "my-seq-store"
    }
  ]
}
```

```
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [ListSequenceStores](#) in *AWS CLI Command Reference*.

list-shares

The following code example shows how to use `list-shares`.

AWS CLI

To list the available shares of a HealthOmics analytics data

The following `list-shares` example lists all shares that have been created for a resource-owner.

```
aws omics list-shares \  
  --resource-owner SELF
```

Output:

```
{  
  "shares": [  
    {  
      "shareId": "595c1cbd-a008-4eca-a887-954d30c91c6e",  
      "name": "myShare",  
      "resourceArn": "arn:aws:omics:us-west-2:555555555555:variantStore/  
store_1",  
      "principalSubscriber": "123456789012",  
      "ownerId": "555555555555",  
      "status": "PENDING"  
    },  
    {  
      "shareId": "39b65d0d-4368-4a19-9814-b0e31d73c10a",  
      "name": "myShare3456",  
      "resourceArn": "arn:aws:omics:us-west-2:555555555555:variantStore/  
store_2",  
      "principalSubscriber": "123456789012",  
      "ownerId": "555555555555",  
      "status": "ACTIVE"  
    }  
  ],  
}
```

```
{
  "shareId": "203152f5-eef9-459d-a4e0-a691668d44ef",
  "name": "myShare4",
  "resourceArn": "arn:aws:omics:us-west-2:555555555555:variantStore/
store_3",
  "principalSubscriber": "123456789012",
  "ownerId": "555555555555",
  "status": "ACTIVE"
}
]
```

For more information, see [Cross-account sharing](#) in the *AWS HealthOmics User Guide*.

- For API details, see [ListShares](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To get a list of tags

The following `list-tags-for-resource` example gets a list of tags for a workflow with id 1234567.

```
aws omics list-tags-for-resource \
  --resource-arn arn:aws:omics:us-west-2:123456789012:workflow/1234567
```

Output:

```
{
  "tags": {
    "department": "analytics"
  }
}
```

For more information, see [Tagging resources in Amazon Omics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

list-variant-import-jobs

The following code example shows how to use `list-variant-import-jobs`.

AWS CLI

To get a list of variant import jobs

The following `list-variant-import-jobs` example gets a list of variant import jobs.

```
aws omics list-variant-import-jobs
```

Output:

```
{
  "variantImportJobs": [
    {
      "creationTime": "2022-11-23T22:47:02.514002Z",
      "destinationName": "my_var_store",
      "id": "69cb65d6-xmpl-4a4a-9025-4565794b684e",
      "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-
serviceRole-W801XMPL7QZ",
      "runLeftNormalization": false,
      "status": "COMPLETED",
      "updateTime": "2022-11-23T22:49:17.976597Z"
    },
    {
      "creationTime": "2022-11-23T22:42:50.037812Z",
      "destinationName": "my_var_store",
      "id": "edd7b8ce-xmpl-47e2-bc99-258cac95a508",
      "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-
serviceRole-W801XMPL7QZ",
      "runLeftNormalization": false,
      "status": "COMPLETED",
      "updateTime": "2022-11-23T22:45:26.009880Z"
    }
  ]
}
```

For more information, see [Omics Analytics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [ListVariantImportJobs](#) in *AWS CLI Command Reference*.

list-variant-stores

The following code example shows how to use `list-variant-stores`.

AWS CLI

To get a list of variant stores

The following `list-variant-stores` example gets a list of variant stores.

```
aws omics list-variant-stores
```

Output:

```
{
  "variantStores": [
    {
      "creationTime": "2022-11-23T22:09:07.534499Z",
      "id": "02dexplcfdd",
      "name": "my_var_store",
      "reference": {
        "referenceArn": "arn:aws:omics:us-west-2:123456789012:referenceStore/1234567890/reference/1234567890"
      },
      "status": "CREATING",
      "storeArn": "arn:aws:omics:us-west-2:123456789012:variantStore/my_var_store",
      "storeSizeBytes": 0,
      "updateTime": "2022-11-23T22:09:24.931711Z"
    },
    {
      "creationTime": "2022-09-23T23:00:09.140265Z",
      "id": "8777xmpl1a24",
      "name": "myvstore0",
      "status": "ACTIVE",
      "storeArn": "arn:aws:omics:us-west-2:123456789012:variantStore/myvstore0",
      "storeSizeBytes": 0,
      "updateTime": "2022-09-23T23:03:26.013220Z"
    }
  ]
}
```

For more information, see [Omics Analytics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [ListVariantStores](#) in *AWS CLI Command Reference*.

list-workflows

The following code example shows how to use `list-workflows`.

AWS CLI

To get a list of workflows

The following `list-workflows` example gets a list of workflows.

```
aws omics list-workflows
```

Output:

```
{
  "items": [
    {
      "arn": "arn:aws:omics:us-west-2:123456789012:workflow/1234567",
      "creationTime": "2022-09-23T23:08:22.041227Z",
      "digest": "nSCNo/qMWFxmp1XpUdokXJnwgne0axyyc2Y0xVxrJTE=",
      "id": "1234567",
      "name": "my-wkflow-0",
      "status": "ACTIVE",
      "type": "PRIVATE"
    },
    {
      "arn": "arn:aws:omics:us-west-2:123456789012:workflow/1234567",
      "creationTime": "2022-11-30T22:33:16.225368Z",
      "digest":
"sha256:c54bxmp1742dcc26f7fa1f10e37550ddd8f251f418277c0a58e895b801ed28cf",
      "id": "1234567",
      "name": "cram-converter",
      "status": "ACTIVE",
      "type": "PRIVATE"
    }
  ]
}
```

For more information, see [Omics Workflows](#) in the *Amazon Omics Developer Guide*.

- For API details, see [ListWorkflows](#) in *AWS CLI Command Reference*.

start-annotation-import-job

The following code example shows how to use `start-annotation-import-job`.

AWS CLI

To import annotations

The following `start-annotation-import-job` example imports annotations from Amazon S3.

```
aws omics start-annotation-import-job \  
  --destination-name tsv_ann_store \  
  --no-run-left-normalization \  
  --role-arn arn:aws:iam::123456789012:role/omics-service-role-serviceRole-  
W801XMPL7QZ \  
  --items source=s3://omics-artifacts-01d6xmpl4e72dd32/targetedregions.bed.gz
```

Output:

```
{  
  "jobId": "984162c7-xmpl-4d23-ab47-286f7950bfbf"  
}
```

For more information, see [Omics Analytics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [StartAnnotationImportJob](#) in *AWS CLI Command Reference*.

start-read-set-activation-job

The following code example shows how to use `start-read-set-activation-job`.

AWS CLI

To activate an archived read set

The following `start-read-set-activation-job` example activates two read sets.

```
aws omics start-read-set-activation-job \  
  --read-set-ids rs1 rs2
```

```
--sequence-store-id 1234567890 \  
--sources readSetId=1234567890 readSetId=1234567890
```

Output:

```
{  
  "creationTime": "2022-12-06T22:35:10.100Z",  
  "id": "1234567890",  
  "sequenceStoreId": "1234567890",  
  "status": "SUBMITTED"  
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [StartReadSetActivationJob](#) in *AWS CLI Command Reference*.

start-read-set-export-job

The following code example shows how to use `start-read-set-export-job`.

AWS CLI

To export a read set

The following `start-read-set-export-job` example exports two read sets to Amazon S3.

```
aws omics start-read-set-export-job \  
  --sequence-store-id 1234567890 \  
  --sources readSetId=1234567890 readSetId=1234567890 \  
  --role-arn arn:aws:iam::123456789012:role/omics-service-role-serviceRole-  
W801XMPL7QZ  
\  
  --destination s3://omics-artifacts-01d6xmpl4e72dd32/read-set-export/
```

Output:

```
{  
  "creationTime": "2022-12-06T22:37:18.612Z",  
  "destination": "s3://omics-artifacts-01d6xmpl4e72dd32/read-set-export/",  
  "id": "1234567890",  
  "sequenceStoreId": "1234567890",  
}
```



```
"status": "SUBMITTED"
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [StartReadSetExportJob](#) in *AWS CLI Command Reference*.

start-read-set-import-job

The following code example shows how to use `start-read-set-import-job`.

AWS CLI

To import a read set

The following `start-read-set-import-job` example imports a read set.

```
aws omics start-read-set-import-job \
  --sequence-store-id 1234567890 \
  --role-arn arn:aws:iam::123456789012:role/omics-service-role-serviceRole-
W801XMPL7QZ \
  --sources file://readset-sources.json
```

`readset-sources.json` is a JSON document with the following content.

```
[
  {
    "sourceFiles":
    {
      "source1": "s3://omics-artifacts-01d6xmpl4e72dd32/
HG00100.chrom20.ILLUMINA.bwa.GBR.low_coverage.20101123.bam"
    },
    "sourceFileType": "BAM",
    "subjectId": "bam-subject",
    "sampleId": "bam-sample",
    "referenceArn": "arn:aws:omics:us-
west-2:123456789012:referenceStore/1234567890/reference/1234567890",
    "name": "HG00100"
  }
]
```

Output:

```
{
  "creationTime": "2022-11-23T01:36:38.158Z",
  "id": "1234567890",
  "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-serviceRole-
W801XMPL7QZ",
  "sequenceStoreId": "1234567890",
  "status": "SUBMITTED"
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [StartReadSetImportJob](#) in *AWS CLI Command Reference*.

start-reference-import-job

The following code example shows how to use `start-reference-import-job`.

AWS CLI

To import a reference genome

The following `start-reference-import-job` example imports a reference genome from Amazon S3.

```
aws omics start-reference-import-job \
  --reference-store-id 1234567890 \
  --role-arn arn:aws:iam::123456789012:role/omics-service-role-serviceRole-
W801XMPL7QZ \
  --sources sourceFile=s3://omics-artifacts-01d6xmpl4e72dd32/
Homo_sapiens_assembly38.fasta,name=assembly-38
```

Output:

```
{
  "creationTime": "2022-11-22T22:25:41.124Z",
  "id": "1234567890",
  "referenceStoreId": "1234567890",
  "roleArn": "arn:aws:iam::123456789012:role/omics-service-role-serviceRole-
W801XMPL7QZ",
  "status": "SUBMITTED"
}
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [StartReferenceImportJob](#) in *AWS CLI Command Reference*.

start-run

The following code example shows how to use `start-run`.

AWS CLI

To run a workflow

The following `start-run` example runs a workflow with ID 1234567.

```
aws omics start-run \  
  --workflow-id 1234567 \  
  --role-arn arn:aws:iam::123456789012:role/omics-service-role-serviceRole-  
W801XMPL7QZ \  
  --name 'cram-to-bam' \  
  --output-uri s3://omics-artifacts-01d6xmpl4e72dd32/workflow-output/ \  
  --run-group-id 1234567 \  
  --priority 1 \  
  --storage-capacity 10 \  
  --log-level ALL \  
  --parameters file://workflow-inputs.json
```

`workflow-inputs.json` is a JSON document with the following content.

```
{  
  "sample_name": "NA12878",  
  "input_cram": "s3://omics-artifacts-01d6xmpl4e72dd32/NA12878.cram",  
  "ref_dict": "s3://omics-artifacts-01d6xmpl4e72dd32/  
Homo_sapiens_assembly38.dict",  
  "ref_fasta": "s3://omics-artifacts-01d6xmpl4e72dd32/  
Homo_sapiens_assembly38.fasta",  
  "ref_fasta_index": "omics-artifacts-01d6xmpl4e72dd32/  
Homo_sapiens_assembly38.fasta.fai"  
}
```

Output:

```
{
```

```
"arn": "arn:aws:omics:us-west-2:123456789012:run/1234567",
"id": "1234567",
"status": "PENDING",
"tags": {}
}
```

For more information, see [Omics Workflows](#) in the *Amazon Omics Developer Guide*.

To load source files from Amazon Omics

You can also load source files from Amazon Omics storage, by using service-specific URIs. The following example workflow-inputs.json file uses Amazon Omics URIs for read set and reference genome sources.

```
{
  "sample_name": "NA12878",
  "input_cram": "omics://123456789012.storage.us-west-2.amazonaws.com/1234567890/
readSet/1234567890/source1",
  "ref_dict": "s3://omics-artifacts-01d6xmpl4e72dd32/
Homo_sapiens_assembly38.dict",
  "ref_fasta": "omics://123456789012.storage.us-west-2.amazonaws.com/1234567890/
reference/1234567890",
  "ref_fasta_index": "omics://123456789012.storage.us-
west-2.amazonaws.com/1234567890/reference/1234567890/index"
}
```

For more information, see [Omics Workflows](#) in the *Amazon Omics Developer Guide*.

- For API details, see [StartRun](#) in *AWS CLI Command Reference*.

start-variant-import-job

The following code example shows how to use `start-variant-import-job`.

AWS CLI

To import a variant file

The following `start-variant-import-job` example imports a VCF format variant file.

```
aws omics start-variant-import-job \
  --destination-name my_var_store \
  --no-run-left-normalization \
```

```
--role-arn arn:aws:iam::123456789012:role/omics-service-role-serviceRole-
W801XMPL7QZ \
--items source=s3://omics-artifacts-01d6xmpl4e72dd32/
Homo_sapiens_assembly38.known_indels.vcf.gz
```

Output:

```
{
  "jobId": "edd7b8ce-xmpl-47e2-bc99-258cac95a508"
}
```

For more information, see [Omics Analytics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [StartVariantImportJob](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To tag a resource

The following `tag-resource` example adds a department tag to a workflow with id 1234567.

```
aws omics tag-resource \
--resource-arn arn:aws:omics:us-west-2:123456789012:workflow/1234567 \
--tags department=analytics
```

For more information, see [Tagging resources in Amazon Omics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag from a resource

The following `untag-resource` example removes the department tag from a workflow.

```
aws omics untag-resource \  
  --resource-arn arn:aws:omics:us-west-2:123456789012:workflow/1234567 \  
  --tag-keys department
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-annotation-store

The following code example shows how to use `update-annotation-store`.

AWS CLI

To update an annotation store

The following `update-annotation-store` example updates the description of an annotation store named `my_vcf_store`.

```
aws omics update-annotation-store \  
  --name my_vcf_store \  
  --description "VCF annotation store"
```

Output:

```
{  
  "creationTime": "2022-12-05T18:00:56.101860Z",  
  "description": "VCF annotation store",  
  "id": "bd6axmpl2444",  
  "name": "my_vcf_store",  
  "reference": {  
    "referenceArn": "arn:aws:omics:us-  
west-2:123456789012:referenceStore/1234567890/reference/1234567890"  
  },  
  "status": "ACTIVE",  
  "storeFormat": "VCF",  
  "updateTime": "2022-12-05T18:13:16.100051Z"  
}
```

For more information, see [Omics Analytics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [UpdateAnnotationStore](#) in *AWS CLI Command Reference*.

update-run-group

The following code example shows how to use `update-run-group`.

AWS CLI

To update a run group

The following `update-run-group` example updates the settings of a run group with id 1234567.

```
aws omics update-run-group \  
  --id 1234567 \  
  --max-cpus 10
```

Output:

```
{  
  "arn": "arn:aws:omics:us-west-2:123456789012:runGroup/1234567",  
  "creationTime": "2022-12-01T00:58:42.915219Z",  
  "id": "1234567",  
  "maxCpus": 10,  
  "maxDuration": 600,  
  "name": "cram-convert",  
  "tags": {}  
}
```

For more information, see [Omics Workflows](#) in the *Amazon Omics Developer Guide*.

- For API details, see [UpdateRunGroup](#) in *AWS CLI Command Reference*.

update-variant-store

The following code example shows how to use `update-variant-store`.

AWS CLI

To update a variant store

The following `update-variant-store` example updates the description of a variant store named `my_var_store`.

```
aws omics update-variant-store \  
  --name my_var_store \  
  --description "variant store"
```

Output:

```
{  
  "creationTime": "2022-11-23T22:09:07.534499Z",  
  "description": "variant store",  
  "id": "02dexplcfd",  
  "name": "my_var_store",  
  "reference": {  
    "referenceArn": "arn:aws:omics:us-  
west-2:123456789012:referenceStore/1234567890/reference/1234567890"  
  },  
  "status": "ACTIVE",  
  "updateTime": "2022-12-05T18:23:37.686402Z"  
}
```

For more information, see [Omics Analytics](#) in the *Amazon Omics Developer Guide*.

- For API details, see [UpdateVariantStore](#) in *AWS CLI Command Reference*.

update-workflow

The following code example shows how to use `update-workflow`.

AWS CLI

To update a workflow

The following `update-workflow` example updates the description of a workflow with ID `1234567`.

```
aws omics update-workflow \  
  --id 1234567 \  
  --description "copy workflow"
```

For more information, see [Omics Storage](#) in the *Amazon Omics Developer Guide*.

- For API details, see [UpdateWorkflow](#) in *AWS CLI Command Reference*.

upload-read-set-part

The following code example shows how to use `upload-read-set-part`.

AWS CLI

To upload a read set part.

The following `upload-read-set-part` example uploads a specified part of a read set.

```
aws omics upload-read-set-part \  
  --sequence-store-id 0123456789 \  
  --upload-id 1122334455 \  
  --part-source SOURCE1 \  
  --part-number 1 \  
  --payload /path/to/file/read_1_part_1.fastq.gz
```

Output:

```
{  
  "checksum": "984979b9928ae8d8622286c4a9cd8e99d964a22d59ed0f5722e1733eb280e635"  
}
```

For more information, see [Direct upload to a sequence store](#) in the *AWS HealthOmics User Guide*.

- For API details, see [UploadReadSetPart](#) in *AWS CLI Command Reference*.

IAM examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with IAM.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-client-id-to-open-id-connect-provider

The following code example shows how to use `add-client-id-to-open-id-connect-provider`.

AWS CLI

To add a client ID (audience) to an Open-ID Connect (OIDC) provider

The following `add-client-id-to-open-id-connect-provider` command adds the client ID `my-application-ID` to the OIDC provider named `server.example.com`.

```
aws iam add-client-id-to-open-id-connect-provider \  
  --client-id my-application-ID \  
  --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/  
server.example.com
```

This command produces no output.

To create an OIDC provider, use the `create-open-id-connect-provider` command.

For more information, see [Creating OpenID Connect \(OIDC\) identity providers](#) in the *AWS IAM User Guide*.

- For API details, see [AddClientIdToOpenIdConnectProvider](#) in *AWS CLI Command Reference*.

add-role-to-instance-profile

The following code example shows how to use `add-role-to-instance-profile`.

AWS CLI

To add a role to an instance profile

The following `add-role-to-instance-profile` command adds the role named `S3Access` to the instance profile named `Webserver`.

```
aws iam add-role-to-instance-profile \  
  --role-name S3Access \  
  --instance-profile-name Webserver
```

This command produces no output.

To create an instance profile, use the `create-instance-profile` command.

For more information, see [Using an IAM role to grant permissions to applications running on Amazon EC2 instances](#) in the *AWS IAM User Guide*.

- For API details, see [AddRoleToInstanceProfile](#) in *AWS CLI Command Reference*.

add-user-to-group

The following code example shows how to use `add-user-to-group`.

AWS CLI

To add a user to an IAM group

The following `add-user-to-group` command adds an IAM user named `Bob` to the IAM group named `Admins`.

```
aws iam add-user-to-group \  
  --user-name Bob \  
  --group-name Admins
```

This command produces no output.

For more information, see [Adding and removing users in an IAM user group](#) in the *AWS IAM User Guide*.

- For API details, see [AddUserToGroup](#) in *AWS CLI Command Reference*.

attach-group-policy

The following code example shows how to use `attach-group-policy`.

AWS CLI

To attach a managed policy to an IAM group

The following `attach-group-policy` command attaches the AWS managed policy named `ReadOnlyAccess` to the IAM group named `Finance`.

```
aws iam attach-group-policy \  
  --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \  
  --group-name Finance
```

This command produces no output.

For more information, see [Managed policies and inline policies](#) in the *AWS IAM User Guide*.

- For API details, see [AttachGroupPolicy](#) in *AWS CLI Command Reference*.

attach-role-policy

The following code example shows how to use `attach-role-policy`.

AWS CLI

To attach a managed policy to an IAM role

The following `attach-role-policy` command attaches the AWS managed policy named `ReadOnlyAccess` to the IAM role named `ReadOnlyRole`.

```
aws iam attach-role-policy \  
  --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \  
  --role-name ReadOnlyRole
```

This command produces no output.

For more information, see [Managed policies and inline policies](#) in the *AWS IAM User Guide*.

- For API details, see [AttachRolePolicy](#) in *AWS CLI Command Reference*.

attach-user-policy

The following code example shows how to use `attach-user-policy`.

AWS CLI

To attach a managed policy to an IAM user

The following `attach-user-policy` command attaches the AWS managed policy named `AdministratorAccess` to the IAM user named `Alice`.

```
aws iam attach-user-policy \  
  --policy-arn arn:aws:iam::aws:policy/AdministratorAccess \  
  --user-name Alice
```

This command produces no output.

For more information, see [Managed policies and inline policies](#) in the *AWS IAM User Guide*.

- For API details, see [AttachUserPolicy](#) in *AWS CLI Command Reference*.

change-password

The following code example shows how to use `change-password`.

AWS CLI

To change the password for your IAM user

To change the password for your IAM user, we recommend using the `--cli-input-json` parameter to pass a JSON file that contains your old and new passwords. Using this method, you can use strong passwords with non-alphanumeric characters. It can be difficult to use passwords with non-alphanumeric characters when you pass them as command line parameters. To use the `--cli-input-json` parameter, start by using the `change-password` command with the `--generate-cli-skeleton` parameter, as in the following example.

```
aws iam change-password \  
  --generate-cli-skeleton > change-password.json
```

The previous command creates a JSON file called `change-password.json` that you can use to fill in your old and new passwords. For example, the file might look like the following.

```
{  
  "OldPassword": "3s0K_;xh4~8XXI",
```

```
"NewPassword": "]35d/{pB9Fo9wJ"}
}
```

Next, to change your password, use the `change-password` command again, this time passing the `--cli-input-json` parameter to specify your JSON file. The following `change-password` command uses the `--cli-input-json` parameter with a JSON file called `change-password.json`.

```
aws iam change-password \
  --cli-input-json file://change-password.json
```

This command produces no output.

This command can be called by IAM users only. If this command is called using AWS account (root) credentials, the command returns an `InvalidUserType` error.

For more information, see [How an IAM user changes their own password](#) in the *AWS IAM User Guide*.

- For API details, see [ChangePassword](#) in *AWS CLI Command Reference*.

create-access-key

The following code example shows how to use `create-access-key`.

AWS CLI

To create an access key for an IAM user

The following `create-access-key` command creates an access key (access key ID and secret access key) for the IAM user named Bob.

```
aws iam create-access-key \
  --user-name Bob
```

Output:

```
{
  "AccessKey": {
    "UserName": "Bob",
```

```
"Status": "Active",  
"CreateDate": "2015-03-09T18:39:23.411Z",  
"SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxrFiCYzEXAMPLEKEY",  
"AccessKeyId": "AKIAIOSFODNN7EXAMPLE"  
}  
}
```

Store the secret access key in a secure location. If it is lost, it cannot be recovered, and you must create a new access key.

For more information, see [Managing access keys for IAM users](#) in the *AWS IAM User Guide*.

- For API details, see [CreateAccessKey](#) in *AWS CLI Command Reference*.

create-account-alias

The following code example shows how to use `create-account-alias`.

AWS CLI

To create an account alias

The following `create-account-alias` command creates the alias `examplecorp` for your AWS account.

```
aws iam create-account-alias \  
  --account-alias examplecorp
```

This command produces no output.

For more information, see [Your AWS account ID and its alias](#) in the *AWS IAM User Guide*.

- For API details, see [CreateAccountAlias](#) in *AWS CLI Command Reference*.

create-group

The following code example shows how to use `create-group`.

AWS CLI

To create an IAM group

The following `create-group` command creates an IAM group named Admins.

```
aws iam create-group \  
  --group-name Admins
```

Output:

```
{  
  "Group": {  
    "Path": "/",  
    "CreateDate": "2015-03-09T20:30:24.940Z",  
    "GroupId": "AIDGPMS9R04H3FEXAMPLE",  
    "Arn": "arn:aws:iam::123456789012:group/Admins",  
    "GroupName": "Admins"  
  }  
}
```

For more information, see [Creating IAM user groups](#) in the *AWS IAM User Guide*.

- For API details, see [CreateGroup](#) in *AWS CLI Command Reference*.

create-instance-profile

The following code example shows how to use `create-instance-profile`.

AWS CLI

To create an instance profile

The following `create-instance-profile` command creates an instance profile named Webserver.

```
aws iam create-instance-profile \  
  --instance-profile-name Webserver
```

Output:

```
{  
  "InstanceProfile": {  
    "InstanceProfileId": "AIPAJMBC7DLSPEXAMPLE",
```



```
    "Roles": [],
    "CreateDate": "2015-03-09T20:33:19.626Z",
    "InstanceProfileName": "Webserver",
    "Path": "/",
    "Arn": "arn:aws:iam::123456789012:instance-profile/Webserver"
  }
}
```

To add a role to an instance profile, use the `add-role-to-instance-profile` command.

For more information, see [Using an IAM role to grant permissions to applications running on Amazon EC2 instances](#) in the *AWS IAM User Guide*.

- For API details, see [CreateInstanceProfile](#) in *AWS CLI Command Reference*.

create-login-profile

The following code example shows how to use `create-login-profile`.

AWS CLI

To create a password for an IAM user

To create a password for an IAM user, we recommend using the `--cli-input-json` parameter to pass a JSON file that contains the password. Using this method, you can create a strong password with non-alphanumeric characters. It can be difficult to create a password with non-alphanumeric characters when you pass it as a command line parameter.

To use the `--cli-input-json` parameter, start by using the `create-login-profile` command with the `--generate-cli-skeleton` parameter, as in the following example.

```
aws iam create-login-profile \
  --generate-cli-skeleton > create-login-profile.json
```

The previous command creates a JSON file called `create-login-profile.json` that you can use to fill in the information for a subsequent `create-login-profile` command. For example:

```
{
  "UserName": "Bob",
  "Password": "&1-3a6u:RA0djs",
  "PasswordResetRequired": true
}
```

```
}
```

Next, to create a password for an IAM user, use the `create-login-profile` command again, this time passing the `--cli-input-json` parameter to specify your JSON file. The following `create-login-profile` command uses the `--cli-input-json` parameter with a JSON file called `create-login-profile.json`.

```
aws iam create-login-profile \  
  --cli-input-json file://create-login-profile.json
```

Output:

```
{  
  "LoginProfile": {  
    "UserName": "Bob",  
    "CreateDate": "2015-03-10T20:55:40.274Z",  
    "PasswordResetRequired": true  
  }  
}
```

If the new password violates the account password policy, the command returns a `PasswordPolicyViolation` error.

To change the password for a user that already has one, use `update-login-profile`. To set a password policy for the account, use the `update-account-password-policy` command.

If the account password policy allows them to, IAM users can change their own passwords using the `change-password` command.

For more information, see [Managing passwords for IAM users](#) in the *AWS IAM User Guide*.

- For API details, see [CreateLoginProfile](#) in *AWS CLI Command Reference*.

create-open-id-connect-provider

The following code example shows how to use `create-open-id-connect-provider`.

AWS CLI

To create an OpenID Connect (OIDC) provider

To create an OpenID Connect (OIDC) provider, we recommend using the `--cli-input-json` parameter to pass a JSON file that contains the required parameters. When you create an OIDC provider, you must pass the URL of the provider, and the URL must begin with `https://`. It can be difficult to pass the URL as a command line parameter, because the colon (`:`) and forward slash (`/`) characters have special meaning in some command line environments. Using the `--cli-input-json` parameter gets around this limitation.

To use the `--cli-input-json` parameter, start by using the `create-open-id-connect-provider` command with the `--generate-cli-skeleton` parameter, as in the following example.

```
aws iam create-open-id-connect-provider \  
  --generate-cli-skeleton > create-open-id-connect-provider.json
```

The previous command creates a JSON file called `create-open-id-connect-provider.json` that you can use to fill in the information for a subsequent `create-open-id-connect-provider` command. For example:

```
{  
  "Url": "https://server.example.com",  
  "ClientIDList": [  
    "example-application-ID"  
  ],  
  "ThumbprintList": [  
    "c3768084dfb3d2b68b7897bf5f565da8eEXAMPLE"  
  ]  
}
```

Next, to create the OpenID Connect (OIDC) provider, use the `create-open-id-connect-provider` command again, this time passing the `--cli-input-json` parameter to specify your JSON file. The following `create-open-id-connect-provider` command uses the `--cli-input-json` parameter with a JSON file called `create-open-id-connect-provider.json`.

```
aws iam create-open-id-connect-provider \  
  --cli-input-json file://create-open-id-connect-provider.json
```

Output:

```
{
```

```
"OpenIDConnectProviderArn": "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
}
```

For more information about OIDC providers, see [Creating OpenID Connect \(OIDC\) identity providers](#) in the *AWS IAM User Guide*.

For more information about obtaining thumbprints for an OIDC provider, see [Obtaining the thumbprint for an OpenID Connect Identity Provider](#) in the *AWS IAM User Guide*.

- For API details, see [CreateOpenIdConnectProvider](#) in *AWS CLI Command Reference*.

create-policy-version

The following code example shows how to use `create-policy-version`.

AWS CLI

To create a new version of a managed policy

This example creates a new v2 version of the IAM policy whose ARN is `arn:aws:iam::123456789012:policy/MyPolicy` and makes it the default version.

```
aws iam create-policy-version \
  --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
  --policy-document file://NewPolicyVersion.json \
  --set-as-default
```

Output:

```
{
  "PolicyVersion": {
    "CreateDate": "2015-06-16T18:56:03.721Z",
    "VersionId": "v2",
    "IsDefaultVersion": true
  }
}
```

For more information, see [Versioning IAM policies](#) in the *AWS IAM User Guide*.

- For API details, see [CreatePolicyVersion](#) in *AWS CLI Command Reference*.

create-policy

The following code example shows how to use create-policy.

AWS CLI

Example 1: To create a customer managed policy

The following command creates a customer managed policy named my-policy.

```
aws iam create-policy \  
  --policy-name my-policy \  
  --policy-document file://policy
```

The file policy is a JSON document in the current folder that grants read only access to the shared folder in an Amazon S3 bucket named my-bucket.

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "s3:Get*",  
        "s3:List*"  
      ],  
      "Resource": [  
        "arn:aws:s3:::my-bucket/shared/*"  
      ]  
    }  
  ]  
}
```

Output:

```
{  
  "Policy": {  
    "PolicyName": "my-policy",  
    "CreateDate": "2015-06-01T19:31:18.620Z",  
    "AttachmentCount": 0,  
    "IsAttachable": true,
```

```
"PolicyId": "ZXR6A36LTYANPAI7NJ5UV",
"DefaultVersionId": "v1",
"Path": "/",
"Arn": "arn:aws:iam::0123456789012:policy/my-policy",
"UpdateDate": "2015-06-01T19:31:18.620Z"
}
}
```

For more information on using files as input for string parameters, see [Specify parameter values for the AWS CLI](#) in the *AWS CLI User Guide*.

Example 2: To create a customer managed policy with a description

The following command creates a customer managed policy named `my-policy` with an immutable description:

```
aws iam create-policy \
  --policy-name my-policy \
  --policy-document file://policy.json \
  --description "This policy grants access to all Put, Get, and List actions for
my-bucket"
```

The file `policy.json` is a JSON document in the current folder that grants access to all Put, List, and Get actions for an Amazon S3 bucket named `my-bucket`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket*",
        "s3:PutBucket*",
        "s3:GetBucket*"
      ],
      "Resource": [
        "arn:aws:s3:::my-bucket"
      ]
    }
  ]
}
```

Output:

```
{
  "Policy": {
    "PolicyName": "my-policy",
    "PolicyId": "ANPAWGSUGIDPEXAMPLE",
    "Arn": "arn:aws:iam::123456789012:policy/my-policy",
    "Path": "/",
    "DefaultVersionId": "v1",
    "AttachmentCount": 0,
    "PermissionsBoundaryUsageCount": 0,
    "IsAttachable": true,
    "CreateDate": "2023-05-24T22:38:47+00:00",
    "UpdateDate": "2023-05-24T22:38:47+00:00"
  }
}
```

For more information on Identity-based Policies, see [Identity-based policies and resource-based policies](#) in the *AWS IAM User Guide*.

Example 3: To Create a customer managed policy with tags

The following command creates a customer managed policy named `my-policy` with tags. This example uses the `--tags` parameter flag with the following JSON-formatted tags: `'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. Alternatively, the `--tags` flag can be used with tags in the shorthand format: `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`.

```
aws iam create-policy \
  --policy-name my-policy \
  --policy-document file://policy.json \
  --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
  "Value": "Seattle"}'
```

The file `policy.json` is a JSON document in the current folder that grants access to all Put, List, and Get actions for an Amazon S3 bucket named `my-bucket`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
    "Action": [
      "s3:ListBucket*",
      "s3:PutBucket*",
      "s3:GetBucket*"
    ],
    "Resource": [
      "arn:aws:s3:::my-bucket"
    ]
  }
]
```

Output:

```
{
  "Policy": {
    "PolicyName": "my-policy",
    "PolicyId": "ANPAWGSUGIDPEXAMPLE",
    "Arn": "arn:aws:iam::12345678012:policy/my-policy",
    "Path": "/",
    "DefaultVersionId": "v1",
    "AttachmentCount": 0,
    "PermissionsBoundaryUsageCount": 0,
    "IsAttachable": true,
    "CreateDate": "2023-05-24T23:16:39+00:00",
    "UpdateDate": "2023-05-24T23:16:39+00:00",
    "Tags": [
      {
        "Key": "Department",
        "Value": "Accounting"
      },
      {
        "Key": "Location",
        "Value": "Seattle"
      }
    ]
  }
}
```

For more information on Tagging policies, see [Tagging customer managed policies](#) in the *AWS IAM User Guide*.

- For API details, see [CreatePolicy](#) in *AWS CLI Command Reference*.

create-role

The following code example shows how to use `create-role`.

AWS CLI

Example 1: To create an IAM role

The following `create-role` command creates a role named `Test-Role` and attaches a trust policy to it.

```
aws iam create-role \  
  --role-name Test-Role \  
  --assume-role-policy-document file://Test-Role-Trust-Policy.json
```

Output:

```
{  
  "Role": {  
    "AssumeRolePolicyDocument": "<URL-encoded-JSON>",  
    "RoleId": "AKIAIOSFODNN7EXAMPLE",  
    "CreateDate": "2013-06-07T20:43:32.821Z",  
    "RoleName": "Test-Role",  
    "Path": "/",  
    "Arn": "arn:aws:iam::123456789012:role/Test-Role"  
  }  
}
```

The trust policy is defined as a JSON document in the `Test-Role-Trust-Policy.json` file. (The file name and extension do not have significance.) The trust policy must specify a principal.

To attach a permissions policy to a role, use the `put-role-policy` command.

For more information, see [Creating IAM roles](#) in the *AWS IAM User Guide*.

Example 2: To create an IAM role with specified maximum session duration

The following `create-role` command creates a role named `Test-Role` and sets a maximum session duration of 7200 seconds (2 hours).

```
aws iam create-role \  
  --role-name Test-Role \  
  --assume-role-policy-document file://Test-Role-Trust-Policy.json
```

```
--role-name Test-Role \  
--assume-role-policy-document file://Test-Role-Trust-Policy.json \  
--max-session-duration 7200
```

Output:

```
{  
  "Role": {  
    "Path": "/",  
    "RoleName": "Test-Role",  
    "RoleId": "AKIAIOSFODNN7EXAMPLE",  
    "Arn": "arn:aws:iam::12345678012:role/Test-Role",  
    "CreateDate": "2023-05-24T23:50:25+00:00",  
    "AssumeRolePolicyDocument": {  
      "Version": "2012-10-17",  
      "Statement": [  
        {  
          "Sid": "Statement1",  
          "Effect": "Allow",  
          "Principal": {  
            "AWS": "arn:aws:iam::12345678012:root"  
          },  
          "Action": "sts:AssumeRole"  
        }  
      ]  
    }  
  }  
}
```

For more information, see [Modifying a role maximum session duration \(AWS API\)](#) in the *AWS IAM User Guide*.

Example 3: To create an IAM Role with tags

The following command creates an IAM Role Test-Role with tags. This example uses the `--tags` parameter flag with the following JSON-formatted tags: `'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. Alternatively, the `--tags` flag can be used with tags in the shorthand format: `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`.

```
aws iam create-role \  

```

```
--role-name Test-Role \  
--assume-role-policy-document file://Test-Role-Trust-Policy.json \  
--tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",  
"Value": "Seattle"}'
```

Output:

```
{  
  "Role": {  
    "Path": "/",  
    "RoleName": "Test-Role",  
    "RoleId": "AKIAIOSFODNN7EXAMPLE",  
    "Arn": "arn:aws:iam::123456789012:role/Test-Role",  
    "CreateDate": "2023-05-25T23:29:41+00:00",  
    "AssumeRolePolicyDocument": {  
      "Version": "2012-10-17",  
      "Statement": [  
        {  
          "Sid": "Statement1",  
          "Effect": "Allow",  
          "Principal": {  
            "AWS": "arn:aws:iam::123456789012:root"  
          },  
          "Action": "sts:AssumeRole"  
        }  
      ]  
    },  
    "Tags": [  
      {  
        "Key": "Department",  
        "Value": "Accounting"  
      },  
      {  
        "Key": "Location",  
        "Value": "Seattle"  
      }  
    ]  
  }  
}
```

For more information, see [Tagging IAM roles](#) in the *AWS IAM User Guide*.

- For API details, see [CreateRole](#) in *AWS CLI Command Reference*.

create-saml-provider

The following code example shows how to use `create-saml-provider`.

AWS CLI

To create a SAML provider

This example creates a new SAML provider in IAM named `MySAMLProvider`. It is described by the SAML metadata document found in the file `SAMLMetaData.xml`.

```
aws iam create-saml-provider \  
  --saml-metadata-document file://SAMLMetaData.xml \  
  --name MySAMLProvider
```

Output:

```
{  
  "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/MySAMLProvider"  
}
```

For more information, see [Creating IAM SAML identity providers](#) in the *AWS IAM User Guide*.

- For API details, see [CreateSAMLProvider](#) in *AWS CLI Command Reference*.

create-service-linked-role

The following code example shows how to use `create-service-linked-role`.

AWS CLI

To create a service-linked role

The following `create-service-linked-role` example creates a service-linked role for the specified AWS service and attaches the specified description.

```
aws iam create-service-linked-role \  
  --aws-service-name lex.amazonaws.com \  
  --description "My service-linked role to support Lex"
```

Output:

```
{
  "Role": {
    "Path": "/aws-service-role/lex.amazonaws.com/",
    "RoleName": "AWSServiceRoleForLexBots",
    "RoleId": "AROAI234567890EXAMPLE",
    "Arn": "arn:aws:iam::1234567890:role/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots",
    "CreateDate": "2019-04-17T20:34:14+00:00",
    "AssumeRolePolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Action": [
            "sts:AssumeRole"
          ],
          "Effect": "Allow",
          "Principal": {
            "Service": [
              "lex.amazonaws.com"
            ]
          }
        }
      ]
    }
  }
}
```

For more information, see [Using service-linked roles](#) in the *AWS IAM User Guide*.

- For API details, see [CreateServiceLinkedRole](#) in *AWS CLI Command Reference*.

create-service-specific-credential

The following code example shows how to use `create-service-specific-credential`.

AWS CLI

Create a set of service-specific credentials for a user

The following `create-service-specific-credential` example creates a username and password that can be used to access only the configured service.

```
aws iam create-service-specific-credential \
```

```
--user-name sofia \  
--service-name codecommit.amazonaws.com
```

Output:

```
{  
  "ServiceSpecificCredential": {  
    "CreateDate": "2019-04-18T20:45:36+00:00",  
    "ServiceName": "codecommit.amazonaws.com",  
    "ServiceUserName": "sofia-at-123456789012",  
    "ServicePassword": "k1zPZM6uVxMQ3oxqgoY1NuJPyRTZ1vREs76zTQE3eJk=",  
    "ServiceSpecificCredentialId": "ACCAEXAMPLE123EXAMPLE",  
    "UserName": "sofia",  
    "Status": "Active"  
  }  
}
```

For more information, see [Create Git credentials for HTTPS connections to CodeCommit](#) in the *AWS CodeCommit User Guide*.

- For API details, see [CreateServiceSpecificCredential](#) in *AWS CLI Command Reference*.

create-user

The following code example shows how to use `create-user`.

AWS CLI

Example 1: To create an IAM user

The following `create-user` command creates an IAM user named Bob in the current account.

```
aws iam create-user \  
--user-name Bob
```

Output:

```
{  
  "User": {  
    "UserName": "Bob",  
    "Path": "/",
```

```
    "CreateDate": "2023-06-08T03:20:41.270Z",
    "UserId": "AIDAIOSFODNN7EXAMPLE",
    "Arn": "arn:aws:iam::123456789012:user/Bob"
  }
}
```

For more information, see [Creating an IAM user in your AWS account](#) in the *AWS IAM User Guide*.

Example 2: To create an IAM user at a specified path

The following `create-user` command creates an IAM user named Bob at the specified path.

```
aws iam create-user \
  --user-name Bob \
  --path /division_abc/subdivision_xyz/
```

Output:

```
{
  "User": {
    "Path": "/division_abc/subdivision_xyz/",
    "UserName": "Bob",
    "UserId": "AIDAIOSFODNN7EXAMPLE",
    "Arn": "arn:aws:iam::12345678012:user/division_abc/subdivision_xyz/Bob",
    "CreateDate": "2023-05-24T18:20:17+00:00"
  }
}
```

For more information, see [IAM identifiers](#) in the *AWS IAM User Guide*.

Example 3: To Create an IAM User with tags

The following `create-user` command creates an IAM user named Bob with tags. This example uses the `--tags` parameter flag with the following JSON-formatted tags: `'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. Alternatively, the `--tags` flag can be used with tags in the shorthand format: `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`.

```
aws iam create-user \
  --user-name Bob \
```

```
--tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
"Value": "Seattle"}'
```

Output:

```
{
  "User": {
    "Path": "/",
    "UserName": "Bob",
    "UserId": "AIDAIOSFODNN7EXAMPLE",
    "Arn": "arn:aws:iam::12345678012:user/Bob",
    "CreateDate": "2023-05-25T17:14:21+00:00",
    "Tags": [
      {
        "Key": "Department",
        "Value": "Accounting"
      },
      {
        "Key": "Location",
        "Value": "Seattle"
      }
    ]
  }
}
```

For more information, see [Tagging IAM users](#) in the *AWS IAM User Guide*.

Example 3: To create an IAM user with a set permissions boundary

The following `create-user` command creates an IAM user named Bob with the permissions boundary of `AmazonS3FullAccess`.

```
aws iam create-user \  
  --user-name Bob \  
  --permissions-boundary arn:aws:iam::aws:policy/AmazonS3FullAccess
```

Output:

```
{
  "User": {
    "Path": "/",
    "UserName": "Bob",
```



```
"UserId": "AIDAIOSFODNN7EXAMPLE",
"Arn": "arn:aws:iam::12345678012:user/Bob",
"CreateDate": "2023-05-24T17:50:53+00:00",
"PermissionsBoundary": {
  "PermissionsBoundaryType": "Policy",
  "PermissionsBoundaryArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
}
}
```

For more information, see [Permissions boundaries for IAM entities](#) in the *AWS IAM User Guide*.

- For API details, see [CreateUser](#) in *AWS CLI Command Reference*.

create-virtual-mfa-device

The following code example shows how to use `create-virtual-mfa-device`.

AWS CLI

To create a virtual MFA device

This example creates a new virtual MFA device called `BobsMFADevice`. It creates a file that contains bootstrap information called `QRCode.png` and places it in the `C:/` directory. The bootstrap method used in this example is `QRCodePNG`.

```
aws iam create-virtual-mfa-device \
  --virtual-mfa-device-name BobsMFADevice \
  --outfile C:/QRCode.png \
  --bootstrap-method QRCodePNG
```

Output:

```
{
  "VirtualMFADevice": {
    "SerialNumber": "arn:aws:iam::210987654321:mfa/BobsMFADevice"
  }
}
```

For more information, see [Using multi-factor authentication \(MFA\) in AWS](#) in the *AWS IAM User Guide*.

- For API details, see [CreateVirtualMfaDevice](#) in *AWS CLI Command Reference*.

deactivate-mfa-device

The following code example shows how to use `deactivate-mfa-device`.

AWS CLI

To deactivate an MFA device

This command deactivates the virtual MFA device with the ARN `arn:aws:iam::210987654321:mfa/BobsMFADevice` that is associated with the user Bob.

```
aws iam deactivate-mfa-device \  
  --user-name Bob \  
  --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice
```

This command produces no output.

For more information, see [Using multi-factor authentication \(MFA\) in AWS](#) in the *AWS IAM User Guide*.

- For API details, see [DeactivateMfaDevice](#) in *AWS CLI Command Reference*.

decode-authorization-message

The following code example shows how to use `decode-authorization-message`.

AWS CLI

To decode a authorization failure message

The following `decode-authorization-message` example decodes the message returned by the EC2 console when attempting to launch an instance without the required permissions.

```
aws sts decode-authorization-message \  
  --encoded-message lxzA8VEjEvu-s0TTt3PgYCXik9Yak0qsrfJGRZR98xNcyWAxwRq14xIvd-  
npzbgTevuufCTbjeBAaDARg9cbTK1rJbg3awM33o-Vy3ebPErE2-  
mWR9hVYdvX-0zKgV0WF9pWjZaJSMqxB-aLXo-I_8TTvBq88x8IFPbMArNdpu0IjxDjzf22PF3S0E3XvIQ-  
_PE00aUqHCCcsSrFtvxm6yQD1nbm6VTIVrfa0Bzy8lsoMo7SjIaJ2r5vph6SY5vCCwg6o2JKe3hIHTa8zRrDbZSFMkcX  
Xx9AYAAIr6bhcis7C__bZh4d1AAWooHFGKgfoJcWGwgdzgbu9hWyVvKTpeot5hsb8qANYjJRCpXTKpi6PZfdijIkwb6g
```

The output is formatted as a single-line string of JSON text that you can parse with any JSON text processor.

```
{
  "DecodedMessage": "{\"allowed\":false,\"explicitDeny\":false,\"matchedStatements\
\":{\\"items\":[],\"failures\":{\\"items\":[],\"context\":{\\"principal\
\":{\\"id\":\\"AIDAV3ZUEFP6J7GY706L0\", \"name\":\\"chain-user\", \"arn\":\
\\\"arn:aws:iam:403299380220:user/chain-user\\\"}, \"action\":\\"ec2:RunInstances\\\",
\\\"resource\":\\"arn:aws:ec2:us-east-2:403299380220:instance/*\", \"conditions\":
{\\"items\":[{\\"key\":\\"ec2:InstanceMarketType\\\", \"values\":{\\"items\":[{\\"value\
\":\\"on-demand\\\"}]}], {\\"key\":\\"aws:Resource\\\", \"values\":{\\"items\":[{\\"value\
\":\\"instance/*\\\"}]}], {\\"key\":\\"aws:Account\\\", \"values\":{\\"items\":[{\\"value\
\":\\"403299380220\\\"}]}], {\\"key\":\\"ec2:AvailabilityZone\\\", \"values\":{\\"items\
\":[{\\"value\":\\"us-east-2b\\\"}]}], {\\"key\":\\"ec2:ebsoptimized\\\", \"values\":{\\"items\
\":[{\\"value\":\\"false\\\"}]}], {\\"key\":\\"ec2:IsLaunchTemplateResource\\\", \"values\
\":{\\"items\":[{\\"value\":\\"false\\\"}]}], {\\"key\":\\"ec2:InstanceType\\\", \"values\
\":{\\"items\":[{\\"value\":\\"t2.micro\\\"}]}], {\\"key\":\\"ec2:RootDeviceType\\\",
\\\"values\":{\\"items\":[{\\"value\":\\"efs\\\"}]}], {\\"key\":\\"aws:Region\\\", \"values\
\":{\\"items\":[{\\"value\":\\"us-east-2\\\"}]}], {\\"key\":\\"aws:Service\\\", \"values\
\":{\\"items\":[{\\"value\":\\"ec2\\\"}]}], {\\"key\":\\"ec2:InstanceID\\\", \"values\":
{\\"items\":[{\\"value\":\\"*\\\"}]}], {\\"key\":\\"aws:Type\\\", \"values\":{\\"items\":
[{\\"value\":\\"instance\\\"}]}], {\\"key\":\\"ec2:Tenancy\\\", \"values\":{\\"items\":
[{\\"value\":\\"default\\\"}]}], {\\"key\":\\"ec2:Region\\\", \"values\":{\\"items\":[{\\"value\
\":\\"us-east-2\\\"}]}], {\\"key\":\\"aws:ARN\\\", \"values\":{\\"items\":[{\\"value\":
\\\"arn:aws:ec2:us-east-2:403299380220:instance/*\\\"}]}]}]}]}\"
}
```

For more information, see [How can I decode an authorization failure message after receiving an "UnauthorizedOperation" error during an EC2 instance launch?](#) in *AWS re:Post*.

- For API details, see [DecodeAuthorizationMessage](#) in *AWS CLI Command Reference*.

delete-access-key

The following code example shows how to use delete-access-key.

AWS CLI

To delete an access key for an IAM user

The following delete-access-key command deletes the specified access key (access key ID and secret access key) for the IAM user named Bob.

```
aws iam delete-access-key \
  --access-key-id AKIDPMS9R04H3FEXAMPLE \
```

```
--user-name Bob
```

This command produces no output.

To list the access keys defined for an IAM user, use the `list-access-keys` command.

For more information, see [Managing access keys for IAM users](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteAccessKey](#) in *AWS CLI Command Reference*.

delete-account-alias

The following code example shows how to use `delete-account-alias`.

AWS CLI

To delete an account alias

The following `delete-account-alias` command removes the alias `mycompany` for the current account.

```
aws iam delete-account-alias \  
  --account-alias mycompany
```

This command produces no output.

For more information, see [Your AWS account ID and its alias](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteAccountAlias](#) in *AWS CLI Command Reference*.

delete-account-password-policy

The following code example shows how to use `delete-account-password-policy`.

AWS CLI

To delete the current account password policy

The following `delete-account-password-policy` command removes the password policy for the current account.

```
aws iam delete-account-password-policy
```

This command produces no output.

For more information, see [Setting an account password policy for IAM users](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteAccountPasswordPolicy](#) in *AWS CLI Command Reference*.

delete-group-policy

The following code example shows how to use `delete-group-policy`.

AWS CLI

To delete a policy from an IAM group

The following `delete-group-policy` command deletes the policy named `ExamplePolicy` from the group named `Admins`.

```
aws iam delete-group-policy \  
  --group-name Admins \  
  --policy-name ExamplePolicy
```

This command produces no output.

To see the policies attached to a group, use the `list-group-policies` command.

For more information, see [Managing IAM policies](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteGroupPolicy](#) in *AWS CLI Command Reference*.

delete-group

The following code example shows how to use `delete-group`.

AWS CLI

To delete an IAM group

The following `delete-group` command deletes an IAM group named `MyTestGroup`.

```
aws iam delete-group \  
  --group-name MyTestGroup
```

This command produces no output.

For more information, see [Deleting an IAM user group](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteGroup](#) in *AWS CLI Command Reference*.

delete-instance-profile

The following code example shows how to use delete-instance-profile.

AWS CLI

To delete an instance profile

The following delete-instance-profile command deletes the instance profile named ExampleInstanceProfile.

```
aws iam delete-instance-profile \  
  --instance-profile-name ExampleInstanceProfile
```

This command produces no output.

For more information, see [Using instance profiles](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteInstanceProfile](#) in *AWS CLI Command Reference*.

delete-login-profile

The following code example shows how to use delete-login-profile.

AWS CLI

To delete a password for an IAM user

The following delete-login-profile command deletes the password for the IAM user named Bob.

```
aws iam delete-login-profile \  
  --user-name Bob
```

This command produces no output.

For more information, see [Managing passwords for IAM users](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteLoginProfile](#) in *AWS CLI Command Reference*.

delete-open-id-connect-provider

The following code example shows how to use `delete-open-id-connect-provider`.

AWS CLI

To delete an IAM OpenID Connect identity provider

This example deletes the IAM OIDC provider that connects to the provider `example.oidcprovider.com`.

```
aws iam delete-open-id-connect-provider \  
  --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/  
  example.oidcprovider.com
```

This command produces no output.

For more information, see [Creating OpenID Connect \(OIDC\) identity providers](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteOpenIdConnectProvider](#) in *AWS CLI Command Reference*.

delete-policy-version

The following code example shows how to use `delete-policy-version`.

AWS CLI

To delete a version of a managed policy

This example deletes the version identified as `v2` from the policy whose ARN is `arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam delete-policy-version \  
  --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \  
  --version-id v2
```

This command produces no output.

For more information, see [Policies and permissions in IAM](#) in the *AWS IAM User Guide*.

- For API details, see [DeletePolicyVersion](#) in *AWS CLI Command Reference*.

delete-policy

The following code example shows how to use delete-policy.

AWS CLI

To delete an IAM policy

This example deletes the policy whose ARN is `arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam delete-policy \  
  --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

This command produces no output.

For more information, see [Policies and permissions in IAM](#) in the *AWS IAM User Guide*.

- For API details, see [DeletePolicy](#) in *AWS CLI Command Reference*.

delete-role-permissions-boundary

The following code example shows how to use delete-role-permissions-boundary.

AWS CLI

To delete a permissions boundary from an IAM role

The following delete-role-permissions-boundary example deletes the permissions boundary for the specified IAM role. To apply a permissions boundary to a role, use the `put-role-permissions-boundary` command.

```
aws iam delete-role-permissions-boundary \  
  --role-name lambda-application-role
```

This command produces no output.

For more information, see [Policies and permissions in IAM](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteRolePermissionsBoundary](#) in *AWS CLI Command Reference*.

delete-role-policy

The following code example shows how to use `delete-role-policy`.

AWS CLI

To remove a policy from an IAM role

The following `delete-role-policy` command removes the policy named `ExamplePolicy` from the role named `Test-Role`.

```
aws iam delete-role-policy \  
  --role-name Test-Role \  
  --policy-name ExamplePolicy
```

This command produces no output.

For more information, see [Modifying a role](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteRolePolicy](#) in *AWS CLI Command Reference*.

delete-role

The following code example shows how to use `delete-role`.

AWS CLI

To delete an IAM role

The following `delete-role` command removes the role named `Test-Role`.

```
aws iam delete-role \  
  --role-name Test-Role
```

This command produces no output.

Before you can delete a role, you must remove the role from any instance profile (`remove-role-from-instance-profile`), detach any managed policies (`detach-role-policy`) and delete any inline policies that are attached to the role (`delete-role-policy`).

For more information, see [Creating IAM roles](#) and [Using instance profiles](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteRole](#) in *AWS CLI Command Reference*.

delete-saml-provider

The following code example shows how to use `delete-saml-provider`.

AWS CLI

To delete a SAML provider

This example deletes the IAM SAML 2.0 provider whose ARN is `arn:aws:iam::123456789012:saml-provider/SAMLADFSPROVIDER`.

```
aws iam delete-saml-provider \  
--saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFSPROVIDER
```

This command produces no output.

For more information, see [Creating IAM SAML identity providers](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteSAMLProvider](#) in *AWS CLI Command Reference*.

delete-server-certificate

The following code example shows how to use `delete-server-certificate`.

AWS CLI

To delete a server certificate from your AWS account

The following `delete-server-certificate` command removes the specified server certificate from your AWS account.

```
aws iam delete-server-certificate \  
--server-certificate-name myUpdatedServerCertificate
```

This command produces no output.

To list the server certificates available in your AWS account, use the `list-server-certificates` command.

For more information, see [Managing server certificates in IAM](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteServerCertificate](#) in *AWS CLI Command Reference*.

delete-service-linked-role

The following code example shows how to use `delete-service-linked-role`.

AWS CLI

To delete a service-linked role

The following `delete-service-linked-role` example deletes the specified service-linked role that you no longer need. The deletion happens asynchronously. You can check the status of the deletion and confirm when it is done by using the `get-service-linked-role-deletion-status` command.

```
aws iam delete-service-linked-role \  
  --role-name AWSServiceRoleForLexBots
```

Output:

```
{  
  "DeletionTaskId": "task/aws-service-role/lex.amazonaws.com/  
  AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE"  
}
```

For more information, see [Using service-linked roles](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteServiceLinkedRole](#) in *AWS CLI Command Reference*.

delete-service-specific-credential

The following code example shows how to use `delete-service-specific-credential`.

AWS CLI

Example 1: Delete a service-specific credential for the requesting user

The following `delete-service-specific-credential` example deletes the specified service-specific credential for the user making the request. The `service-specific-credential-id` is provided when you create the credential and you can retrieve it by using the `list-service-specific-credentials` command.

```
aws iam delete-service-specific-credential \  
  --service-specific-credential-id ACCAEXAMPLE123EXAMPLE
```

This command produces no output.

Example 2: Delete a service-specific credential for a specified user

The following `delete-service-specific-credential` example deletes the specified service-specific credential for the specified user. The `service-specific-credential-id` is provided when you create the credential and you can retrieve it by using the `list-service-specific-credentials` command.

```
aws iam delete-service-specific-credential \  
  --user-name sofia \  
  --service-specific-credential-id ACCAEXAMPLE123EXAMPLE
```

This command produces no output.

For more information, see [Create Git credentials for HTTPS connections to CodeCommit](#) in the *AWS CodeCommit User Guide*.

- For API details, see [DeleteServiceSpecificCredential](#) in *AWS CLI Command Reference*.

delete-signing-certificate

The following code example shows how to use `delete-signing-certificate`.

AWS CLI

To delete a signing certificate for an IAM user

The following `delete-signing-certificate` command deletes the specified signing certificate for the IAM user named Bob.

```
aws iam delete-signing-certificate \  
  --user-name Bob \  
  --certificate-id TA7SMP42TDN5Z260BPJE7EXAMPLE
```

This command produces no output.

To get the ID for a signing certificate, use the `list-signing-certificates` command.

For more information, see [Manage signing certificates](#) in the *Amazon EC2 User Guide*.

- For API details, see [DeleteSigningCertificate](#) in *AWS CLI Command Reference*.

delete-ssh-public-key

The following code example shows how to use `delete-ssh-public-key`.

AWS CLI

To delete an SSH public keys attached to an IAM user

The following `delete-ssh-public-key` command deletes the specified SSH public key attached to the IAM user `sofia`.

```
aws iam delete-ssh-public-key \  
  --user-name sofia \  
  --ssh-public-key-id APKA123456789EXAMPLE
```

This command produces no output.

For more information, see [Use SSH keys and SSH with CodeCommit](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteSshPublicKey](#) in *AWS CLI Command Reference*.

delete-user-permissions-boundary

The following code example shows how to use `delete-user-permissions-boundary`.

AWS CLI

To delete a permissions boundary from an IAM user

The following `delete-user-permissions-boundary` example deletes the permissions boundary attached to the IAM user named `intern`. To apply a permissions boundary to a user, use the `put-user-permissions-boundary` command.

```
aws iam delete-user-permissions-boundary \  
  --user-name intern
```

This command produces no output.

For more information, see [Policies and permissions in IAM](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteUserPermissionsBoundary](#) in *AWS CLI Command Reference*.

delete-user-policy

The following code example shows how to use `delete-user-policy`.

AWS CLI

To remove a policy from an IAM user

The following `delete-user-policy` command removes the specified policy from the IAM user named Bob.

```
aws iam delete-user-policy \  
  --user-name Bob \  
  --policy-name ExamplePolicy
```

This command produces no output.

To get a list of policies for an IAM user, use the `list-user-policies` command.

For more information, see [Creating an IAM user in your AWS account](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteUserPolicy](#) in *AWS CLI Command Reference*.

delete-user

The following code example shows how to use `delete-user`.

AWS CLI

To delete an IAM user

The following `delete-user` command removes the IAM user named Bob from the current account.

```
aws iam delete-user \  
  --user-name Bob
```

This command produces no output.

For more information, see [Deleting an IAM user](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteUser](#) in *AWS CLI Command Reference*.

delete-virtual-mfa-device

The following code example shows how to use `delete-virtual-mfa-device`.

AWS CLI

To remove a virtual MFA device

The following `delete-virtual-mfa-device` command removes the specified MFA device from the current account.

```
aws iam delete-virtual-mfa-device \  
  --serial-number arn:aws:iam::123456789012:mfa/MFATest
```

This command produces no output.

For more information, see [Deactivating MFA devices](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteVirtualMfaDevice](#) in *AWS CLI Command Reference*.

detach-group-policy

The following code example shows how to use `detach-group-policy`.

AWS CLI

To detach a policy from a group

This example removes the managed policy with the ARN `arn:aws:iam::123456789012:policy/TesterAccessPolicy` from the group called `Testers`.

```
aws iam detach-group-policy \  
  --group-name Testers \  
  --policy-arn arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

This command produces no output.

For more information, see [Managing IAM user groups](#) in the *AWS IAM User Guide*.

- For API details, see [DetachGroupPolicy](#) in *AWS CLI Command Reference*.

detach-role-policy

The following code example shows how to use `detach-role-policy`.

AWS CLI

To detach a policy from a role

This example removes the managed policy with the ARN `arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy` from the role called `FedTesterRole`.

```
aws iam detach-role-policy \  
  --role-name FedTesterRole \  
  --policy-arn arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

This command produces no output.

For more information, see [Modifying a role](#) in the *AWS IAM User Guide*.

- For API details, see [DetachRolePolicy](#) in *AWS CLI Command Reference*.

detach-user-policy

The following code example shows how to use `detach-user-policy`.

AWS CLI

To detach a policy from a user

This example removes the managed policy with the ARN `arn:aws:iam::123456789012:policy/TesterPolicy` from the user `Bob`.

```
aws iam detach-user-policy \  
  --user-name Bob \  
  --policy-arn arn:aws:iam::123456789012:policy/TesterPolicy
```

This command produces no output.

For more information, see [Changing permissions for an IAM user](#) in the *AWS IAM User Guide*.

- For API details, see [DetachUserPolicy](#) in *AWS CLI Command Reference*.

enable-mfa-device

The following code example shows how to use `enable-mfa-device`.

AWS CLI

To enable an MFA device

After you use the `create-virtual-mfa-device` command to create a new virtual MFA device, you can assign the MFA device to a user. The following `enable-mfa-device` example assigns the MFA device with the serial number `arn:aws:iam::210987654321:mfa/BobsMFADevice` to the user Bob. The command also synchronizes the device with AWS by including the first two codes in sequence from the virtual MFA device.

```
aws iam enable-mfa-device \  
  --user-name Bob \  
  --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice \  
  --authentication-code1 123456 \  
  --authentication-code2 789012
```

This command produces no output.

For more information, see [Enabling a virtual multi-factor authentication \(MFA\) device](#) in the *AWS IAM User Guide*.

- For API details, see [EnableMfaDevice](#) in *AWS CLI Command Reference*.

generate-credential-report

The following code example shows how to use `generate-credential-report`.

AWS CLI

To generate a credential report

The following example attempts to generate a credential report for the AWS account.

```
aws iam generate-credential-report
```

Output:

```
{
  "State": "STARTED",
  "Description": "No report exists. Starting a new report generation task"
}
```

For more information, see [Getting credential reports for your AWS account](#) in the *AWS IAM User Guide*.

- For API details, see [GenerateCredentialReport](#) in *AWS CLI Command Reference*.

generate-organizations-access-report

The following code example shows how to use `generate-organizations-access-report`.

AWS CLI**Example 1: To generate an access report for a root in an organization**

The following `generate-organizations-access-report` example starts a background job to create an access report for the specified root in an organization. You can display the report after it's created by running the `get-organizations-access-report` command.

```
aws iam generate-organizations-access-report \
  --entity-path o-4fxmpl1t198/r-c3xb
```

Output:

```
{
  "JobId": "a8b6c06f-aaa4-8xmp-28bc-81da71836359"
}
```

Example 2: To generate an access report for an account in an organization

The following `generate-organizations-access-report` example starts a background job to create an access report for account ID 123456789012 in the organization o-4fxmpl1t198. You can display the report after it's created by running the `get-organizations-access-report` command.

```
aws iam generate-organizations-access-report \  
  --entity-path o-4fxmpl1t198/r-c3xb/123456789012
```

Output:

```
{  
  "JobId": "14b6c071-75f6-2xmp-fb77-faf6fb4201d2"  
}
```

Example 3: To generate an access report for an account in an organizational unit in an organization

The following `generate-organizations-access-report` example starts a background job to create an access report for account ID 234567890123 in organizational unit `ou-c3xb-lmu7j2yg` in the organization `o-4fxmpl1t198`. You can display the report after it's created by running the `get-organizations-access-report` command.

```
aws iam generate-organizations-access-report \  
  --entity-path o-4fxmpl1t198/r-c3xb/ou-c3xb-lmu7j2yg/234567890123
```

Output:

```
{  
  "JobId": "2eb6c2e6-0xmp-ec04-1425-c937916a64af"  
}
```

To get details about roots and organizational units in your organization, use the `organizations list-roots` and `organizations list-organizational-units-for-parent` commands.

For more information, see [Refining permissions in AWS using last accessed information](#) in the *AWS IAM User Guide*.

- For API details, see [GenerateOrganizationsAccessReport](#) in *AWS CLI Command Reference*.

generate-service-last-accessed-details

The following code example shows how to use `generate-service-last-accessed-details`.

AWS CLI

Example 1: To generate a service access report for a custom policy

The following `generate-service-last-accessed-details` example starts a background job to generate a report that lists the services accessed by IAM users and other entities with a custom policy named `intern-boundary`. You can display the report after it is created by running the `get-service-last-accessed-details` command.

```
aws iam generate-service-last-accessed-details \  
  --arn arn:aws:iam::123456789012:policy/intern-boundary
```

Output:

```
{  
  "JobId": "2eb6c2b8-7b4c-3xmp-3c13-03b72c8cdfdc"  
}
```

Example 2: To generate a service access report for the AWS managed AdministratorAccess policy

The following `generate-service-last-accessed-details` example starts a background job to generate a report that lists the services accessed by IAM users and other entities with the AWS managed `AdministratorAccess` policy. You can display the report after it is created by running the `get-service-last-accessed-details` command.

```
aws iam generate-service-last-accessed-details \  
  --arn arn:aws:iam::aws:policy/AdministratorAccess
```

Output:

```
{  
  "JobId": "78b6c2ba-d09e-6xmp-7039-ecde30b26916"  
}
```

For more information, see [Refining permissions in AWS using last accessed information](#) in the *AWS IAM User Guide*.

- For API details, see [GenerateServiceLastAccessedDetails](#) in *AWS CLI Command Reference*.

get-access-key-last-used

The following code example shows how to use `get-access-key-last-used`.

AWS CLI

To retrieve information about when the specified access key was last used

The following example retrieves information about when the access key ABCDEXAMPLE was last used.

```
aws iam get-access-key-last-used \
  --access-key-id ABCDEXAMPLE
```

Output:

```
{
  "UserName": "Bob",
  "AccessKeyLastUsed": {
    "Region": "us-east-1",
    "ServiceName": "iam",
    "LastUsedDate": "2015-06-16T22:45:00Z"
  }
}
```

For more information, see [Managing access keys for IAM users](#) in the *AWS IAM User Guide*.

- For API details, see [GetAccessKeyLastUsed](#) in *AWS CLI Command Reference*.

get-account-authorization-details

The following code example shows how to use `get-account-authorization-details`.

AWS CLI

To list an AWS accounts IAM users, groups, roles, and policies

The following `get-account-authorization-details` command returns information about all IAM users, groups, roles, and policies in the AWS account.

```
aws iam get-account-authorization-details
```

Output:

```
{
  "RoleDetailList": [
    {
      "AssumeRolePolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
              "Service": "ec2.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
          }
        ]
      },
      "RoleId": "AROA1234567890EXAMPLE",
      "CreateDate": "2014-07-30T17:09:20Z",
      "InstanceProfileList": [
        {
          "InstanceProfileId": "AIPA1234567890EXAMPLE",
          "Roles": [
            {
              "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                  {
                    "Sid": "",
                    "Effect": "Allow",
                    "Principal": {
                      "Service": "ec2.amazonaws.com"
                    },
                    "Action": "sts:AssumeRole"
                  }
                ]
              },
              "RoleId": "AROA1234567890EXAMPLE",
              "CreateDate": "2014-07-30T17:09:20Z",
              "RoleName": "EC2role",
              "Path": "/",
              "Arn": "arn:aws:iam::123456789012:role/EC2role"
            }
          ]
        }
      ]
    }
  ]
}
```

```

        ],
        "CreateDate": "2014-07-30T17:09:20Z",
        "InstanceProfileName": "EC2role",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:instance-profile/EC2role"
    }
],
"RoleName": "EC2role",
"Path": "/",
"AttachedManagedPolicies": [
    {
        "PolicyName": "AmazonS3FullAccess",
        "PolicyArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
    },
    {
        "PolicyName": "AmazonDynamoDBFullAccess",
        "PolicyArn": "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess"
    }
],
"RoleLastUsed": {
    "Region": "us-west-2",
    "LastUsedDate": "2019-11-13T17:30:00Z"
},
"RolePolicyList": [],
"Arn": "arn:aws:iam::123456789012:role/EC2role"
}
],
"GroupDetailList": [
    {
        "GroupId": "AIDA1234567890EXAMPLE",
        "AttachedManagedPolicies": {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        },
        "GroupName": "Admins",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:group/Admins",
        "CreateDate": "2013-10-14T18:32:24Z",
        "GroupPolicyList": []
    },
    {
        "GroupId": "AIDA1234567890EXAMPLE",
        "AttachedManagedPolicies": {
            "PolicyName": "PowerUserAccess",

```

```
        "PolicyArn": "arn:aws:iam::aws:policy/PowerUserAccess"
    },
    "GroupName": "Dev",
    "Path": "/",
    "Arn": "arn:aws:iam::123456789012:group/Dev",
    "CreateDate": "2013-10-14T18:33:55Z",
    "GroupPolicyList": []
},
{
    "GroupId": "AIDA1234567890EXAMPLE",
    "AttachedManagedPolicies": [],
    "GroupName": "Finance",
    "Path": "/",
    "Arn": "arn:aws:iam::123456789012:group/Finance",
    "CreateDate": "2013-10-14T18:57:48Z",
    "GroupPolicyList": [
        {
            "PolicyName": "policygen-201310141157",
            "PolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Action": "aws-portal:*",
                        "Sid": "Stmnt1381777017000",
                        "Resource": "*",
                        "Effect": "Allow"
                    }
                ]
            }
        }
    ]
}
],
"UserDetailList": [
    {
        "UserName": "Alice",
        "GroupList": [
            "Admins"
        ],
        "CreateDate": "2013-10-14T18:32:24Z",
        "UserId": "AIDA1234567890EXAMPLE",
        "UserPolicyList": [],
        "Path": "/",
        "AttachedManagedPolicies": [],
```



```
    "Arn": "arn:aws:iam::123456789012:user/Alice"
  },
  {
    "UserName": "Bob",
    "GroupList": [
      "Admins"
    ],
    "CreateDate": "2013-10-14T18:32:25Z",
    "UserId": "AIDA1234567890EXAMPLE",
    "UserPolicyList": [
      {
        "PolicyName": "DenyBillingAndIAMPolicy",
        "PolicyDocument": {
          "Version": "2012-10-17",
          "Statement": {
            "Effect": "Deny",
            "Action": [
              "aws-portal:*",
              "iam:*"
            ],
            "Resource": "*"
          }
        }
      }
    ],
    "Path": "/",
    "AttachedManagedPolicies": [],
    "Arn": "arn:aws:iam::123456789012:user/Bob"
  },
  {
    "UserName": "Charlie",
    "GroupList": [
      "Dev"
    ],
    "CreateDate": "2013-10-14T18:33:56Z",
    "UserId": "AIDA1234567890EXAMPLE",
    "UserPolicyList": [],
    "Path": "/",
    "AttachedManagedPolicies": [],
    "Arn": "arn:aws:iam::123456789012:user/Charlie"
  }
],
"Policies": [
  {
```

```

    "PolicyName": "create-update-delete-set-managed-policies",
    "CreateDate": "2015-02-06T19:58:34Z",
    "AttachmentCount": 1,
    "IsAttachable": true,
    "PolicyId": "ANPA1234567890EXAMPLE",
    "DefaultVersionId": "v1",
    "PolicyVersionList": [
      {
        "CreateDate": "2015-02-06T19:58:34Z",
        "VersionId": "v1",
        "Document": {
          "Version": "2012-10-17",
          "Statement": {
            "Effect": "Allow",
            "Action": [
              "iam:CreatePolicy",
              "iam:CreatePolicyVersion",
              "iam>DeletePolicy",
              "iam>DeletePolicyVersion",
              "iam:GetPolicy",
              "iam:GetPolicyVersion",
              "iam>ListPolicies",
              "iam>ListPolicyVersions",
              "iam:SetDefaultPolicyVersion"
            ],
            "Resource": "*"
          }
        },
        "IsDefaultVersion": true
      }
    ],
    "Path": "/",
    "Arn": "arn:aws:iam::123456789012:policy/create-update-delete-set-
managed-policies",
    "UpdateDate": "2015-02-06T19:58:34Z"
  },
  {
    "PolicyName": "S3-read-only-specific-bucket",
    "CreateDate": "2015-01-21T21:39:41Z",
    "AttachmentCount": 1,
    "IsAttachable": true,
    "PolicyId": "ANPA1234567890EXAMPLE",
    "DefaultVersionId": "v1",
    "PolicyVersionList": [

```

```
    {
      "CreateDate": "2015-01-21T21:39:41Z",
      "VersionId": "v1",
      "Document": {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Action": [
              "s3:Get*",
              "s3:List*"
            ],
            "Resource": [
              "arn:aws:s3:::example-bucket",
              "arn:aws:s3:::example-bucket/*"
            ]
          }
        ]
      },
      "IsDefaultVersion": true
    }
  ],
  "Path": "/",
  "Arn": "arn:aws:iam::123456789012:policy/S3-read-only-specific-bucket",
  "UpdateDate": "2015-01-21T23:39:41Z"
},
{
  "PolicyName": "AmazonEC2FullAccess",
  "CreateDate": "2015-02-06T18:40:15Z",
  "AttachmentCount": 1,
  "IsAttachable": true,
  "PolicyId": "ANPA1234567890EXAMPLE",
  "DefaultVersionId": "v1",
  "PolicyVersionList": [
    {
      "CreateDate": "2014-10-30T20:59:46Z",
      "VersionId": "v1",
      "Document": {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Action": "ec2:*",
            "Effect": "Allow",
            "Resource": "*"
          }
        ]
      }
    }
  ]
}
```

```

        },
        {
            "Effect": "Allow",
            "Action": "elasticloadbalancing:*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "cloudwatch:*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "autoscaling:*",
            "Resource": "*"
        }
    ]
},
    "IsDefaultVersion": true
}
],
    "Path": "/",
    "Arn": "arn:aws:iam::aws:policy/AmazonEC2FullAccess",
    "UpdateDate": "2015-02-06T18:40:15Z"
}
],
    "Marker": "EXAMPLEkakov9BCuUNFDtxWSyetzYwEx2ADc8dnzfvERF5S6YMvXKx41t6gCl/
    eeaCX3Jo94/bKqezEAg8TEVS99EKFLxm3jtbpl25FDWEXAMPLE",
    "IsTruncated": true
}

```

For more information, see [AWS security audit guidelines](#) in the *AWS IAM User Guide*.

- For API details, see [GetAccountAuthorizationDetails](#) in *AWS CLI Command Reference*.

get-account-password-policy

The following code example shows how to use `get-account-password-policy`.

AWS CLI

To see the current account password policy

The following `get-account-password-policy` command displays details about the password policy for the current account.

```
aws iam get-account-password-policy
```

Output:

```
{
  "PasswordPolicy": {
    "AllowUsersToChangePassword": false,
    "RequireLowercaseCharacters": false,
    "RequireUppercaseCharacters": false,
    "MinimumPasswordLength": 8,
    "RequireNumbers": true,
    "RequireSymbols": true
  }
}
```

If no password policy is defined for the account, the command returns a `NoSuchEntity` error.

For more information, see [Setting an account password policy for IAM users](#) in the *AWS IAM User Guide*.

- For API details, see [GetAccountPasswordPolicy](#) in *AWS CLI Command Reference*.

get-account-summary

The following code example shows how to use `get-account-summary`.

AWS CLI

To get information about IAM entity usage and IAM quotas in the current account

The following `get-account-summary` command returns information about the current IAM entity usage and current IAM entity quotas in the account.

```
aws iam get-account-summary
```

Output:

```
{
  "SummaryMap": {
```

```
"UsersQuota": 5000,  
"GroupsQuota": 100,  
"InstanceProfiles": 6,  
"SigningCertificatesPerUserQuota": 2,  
"AccountAccessKeysPresent": 0,  
"RolesQuota": 250,  
"RolePolicySizeQuota": 10240,  
"AccountSigningCertificatesPresent": 0,  
"Users": 27,  
"ServerCertificatesQuota": 20,  
"ServerCertificates": 0,  
"AssumeRolePolicySizeQuota": 2048,  
"Groups": 7,  
"MFADevicesInUse": 1,  
"Roles": 3,  
"AccountMFAEnabled": 1,  
"MFADevices": 3,  
"GroupsPerUserQuota": 10,  
"GroupPolicySizeQuota": 5120,  
"InstanceProfilesQuota": 100,  
"AccessKeysPerUserQuota": 2,  
"Providers": 0,  
"UserPolicySizeQuota": 2048  
}  
}
```

For more information about entity limitations, see [IAM and AWS STS quotas](#) in the *AWS IAM User Guide*.

- For API details, see [GetAccountSummary](#) in *AWS CLI Command Reference*.

get-context-keys-for-custom-policy

The following code example shows how to use `get-context-keys-for-custom-policy`.

AWS CLI

Example 1: To list the context keys referenced by one or more custom JSON policies provided as a parameter on the command line

The following `get-context-keys-for-custom-policy` command parses each supplied policy and lists the context keys used by those policies. Use this command to identify which context key values you must supply to successfully use the policy simulator commands

`simulate-custom-policy` and `simulate-custom-policy`. You can also retrieve the list of context keys used by all policies associated by an IAM user or role by using the `get-context-keys-for-custom-policy` command. Parameter values that begin with `file://` instruct the command to read the file and use the contents as the value for the parameter instead of the file name itself.

```
aws iam get-context-keys-for-custom-policy \
  --policy-input-list '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/${aws:username}","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
```

Output:

```
{
  "ContextKeyNames": [
    "aws:username",
    "aws:CurrentTime"
  ]
}
```

Example 2: To list the context keys referenced by one or more custom JSON policies provided as a file input

The following `get-context-keys-for-custom-policy` command is the same as the previous example, except that the policies are provided in a file instead of as a parameter. Because the command expects a JSON list of strings, and not a list of JSON structures, the file must be structured as follows, although you can collapse it into one one.

```
[
  "Policy1",
  "Policy2"
]
```

So for example, a file that contains the policy from the previous example must look like the following. You must escape each embedded double-quote inside the policy string by preceding it with a backslash `"`.

```
[ "{\"Version\": \"2012-10-17\", \"Statement\": {\"Effect\": \"Allow\", \"Action
\": \"dynamodb:*\", \"Resource\": \"arn:aws:dynamodb:us-west-2:128716708097:table/
```

```
{aws:username}\", \"Condition\": {\"DateGreaterThan\": {\"aws:CurrentTime\":  
  \"2015-08-16T12:00:00Z\"}}}}" ]
```

This file can then be submitted to the following command.

```
aws iam get-context-keys-for-custom-policy \  
  --policy-input-list file://policyfile.json
```

Output:

```
{  
  "ContextKeyNames": [  
    "aws:username",  
    "aws:CurrentTime"  
  ]  
}
```

For more information, see [Using the IAM Policy Simulator \(AWS CLI and AWS API\)](#) in the *AWS IAM User Guide*.

- For API details, see [GetContextKeysForCustomPolicy](#) in *AWS CLI Command Reference*.

get-context-keys-for-principal-policy

The following code example shows how to use `get-context-keys-for-principal-policy`.

AWS CLI

To list the context keys referenced by all policies associated with an IAM principal

The following `get-context-keys-for-principal-policy` command retrieves all policies that are attached to the user `saanvi` and any groups she is a member of. It then parses each and lists the context keys used by those policies. Use this command to identify which context key values you must supply to successfully use the `simulate-custom-policy` and `simulate-principal-policy` commands. You can also retrieve the list of context keys used by an arbitrary JSON policy by using the `get-context-keys-for-custom-policy` command.

```
aws iam get-context-keys-for-principal-policy \  
  --policy-source-arn arn:aws:iam::123456789012:user/saanvi
```


Output:

```
{
  "ContextKeyNames": [
    "aws:username",
    "aws:CurrentTime"
  ]
}
```

For more information, see [Using the IAM Policy Simulator \(AWS CLI and AWS API\)](#) in the *AWS IAM User Guide*.

- For API details, see [GetContextKeysForPrincipalPolicy](#) in *AWS CLI Command Reference*.

get-credential-report

The following code example shows how to use `get-credential-report`.

AWS CLI**To get a credential report**

This example opens the returned report and outputs it to the pipeline as an array of text lines.

```
aws iam get-credential-report
```

Output:

```
{
  "GeneratedTime": "2015-06-17T19:11:50Z",
  "ReportFormat": "text/csv"
}
```

For more information, see [Getting credential reports for your AWS account](#) in the *AWS IAM User Guide*.

- For API details, see [GetCredentialReport](#) in *AWS CLI Command Reference*.

get-group-policy

The following code example shows how to use `get-group-policy`.

AWS CLI

To get information about a policy attached to an IAM group

The following `get-group-policy` command gets information about the specified policy attached to the group named `Test-Group`.

```
aws iam get-group-policy \  
  --group-name Test-Group \  
  --policy-name S3-ReadOnly-Policy
```

Output:

```
{  
  "GroupName": "Test-Group",  
  "PolicyDocument": {  
    "Statement": [  
      {  
        "Action": [  
          "s3:Get*",  
          "s3:List*"  
        ],  
        "Resource": "*",  
        "Effect": "Allow"  
      }  
    ]  
  },  
  "PolicyName": "S3-ReadOnly-Policy"  
}
```

For more information, see [Managing IAM policies](#) in the *AWS IAM User Guide*.

- For API details, see [GetGroupPolicy](#) in *AWS CLI Command Reference*.

get-group

The following code example shows how to use `get-group`.

AWS CLI

To get an IAM group

This example returns details about the IAM group `Admins`.

```
aws iam get-group \  
  --group-name Admins
```

Output:

```
{  
  "Group": {  
    "Path": "/",  
    "CreateDate": "2015-06-16T19:41:48Z",  
    "GroupId": "AIDGPMS9R04H3FEXAMPLE",  
    "Arn": "arn:aws:iam::123456789012:group/Admins",  
    "GroupName": "Admins"  
  },  
  "Users": []  
}
```

For more information, see [IAM Identities \(users, user groups, and roles\)](#) in the *AWS IAM User Guide*.

- For API details, see [GetGroup](#) in *AWS CLI Command Reference*.

get-instance-profile

The following code example shows how to use `get-instance-profile`.

AWS CLI

To get information about an instance profile

The following `get-instance-profile` command gets information about the instance profile named `ExampleInstanceProfile`.

```
aws iam get-instance-profile \  
  --instance-profile-name ExampleInstanceProfile
```

Output:

```
{  
  "InstanceProfile": {  
    "InstanceProfileId": "AID2MAB8DPLSRHEXAMPLE",  
    "Roles": [  
      {  
        "RoleId": "AID2MAB8DPLSRHEXAMPLE",  
        "RoleName": "ExampleRole",  
        "AssumeRolePolicyDocument": {  
          "Statement": [  
            {  
              "Action": "sts:AssumeRole",  
              "Effect": "Allow",  
              "Principal": {  
                "AWS": "arn:aws:iam::123456789012:root"  
              }  
            }  
          ]  
        }  
      }  
    ]  
  }  
}
```

```
    {
      "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
      "RoleId": "AIDGPMS9R04H3FEXAMPLE",
      "CreateDate": "2013-01-09T06:33:26Z",
      "RoleName": "Test-Role",
      "Path": "/",
      "Arn": "arn:aws:iam::336924118301:role/Test-Role"
    }
  ],
  "CreateDate": "2013-06-12T23:52:02Z",
  "InstanceProfileName": "ExampleInstanceProfile",
  "Path": "/",
  "Arn": "arn:aws:iam::336924118301:instance-profile/ExampleInstanceProfile"
}
```

For more information, see [Using instance profiles](#) in the *AWS IAM User Guide*.

- For API details, see [GetInstanceProfile](#) in *AWS CLI Command Reference*.

get-login-profile

The following code example shows how to use `get-login-profile`.

AWS CLI

To get password information for an IAM user

The following `get-login-profile` command gets information about the password for the IAM user named Bob.

```
aws iam get-login-profile \
  --user-name Bob
```

Output:

```
{
  "LoginProfile": {
    "UserName": "Bob",
    "CreateDate": "2012-09-21T23:03:39Z"
  }
}
```

The `get-login-profile` command can be used to verify that an IAM user has a password. The command returns a `NoSuchEntity` error if no password is defined for the user.

You cannot view a password using this command. If the password is lost, you can reset the password (`update-login-profile`) for the user. Alternatively, you can delete the login profile (`delete-login-profile`) for the user and then create a new one (`create-login-profile`).

For more information, see [Managing passwords for IAM users](#) in the *AWS IAM User Guide*.

- For API details, see [GetLoginProfile](#) in *AWS CLI Command Reference*.

get-mfa-device

The following code example shows how to use `get-mfa-device`.

AWS CLI

To retrieve information about a FIDO security key

The following `get-mfa-device` command example retrieves information about the specified FIDO security key.

```
aws iam get-mfa-device \  
  --serial-number arn:aws:iam::123456789012:u2f/user/alice/fidokeyname-  
EXAMPLEBN5FHTECLFG7EXAMPLE
```

Output:

```
{  
  "UserName": "alice",  
  "SerialNumber": "arn:aws:iam::123456789012:u2f/user/alice/fidokeyname-  
EXAMPLEBN5FHTECLFG7EXAMPLE",  
  "EnableDate": "2023-09-19T01:49:18+00:00",  
  "Certifications": {  
    "FIDO": "L1"  
  }  
}
```

For more information, see [Using multi-factor authentication \(MFA\) in AWS](#) in the *AWS IAM User Guide*.

- For API details, see [GetMfaDevice](#) in *AWS CLI Command Reference*.

get-open-id-connect-provider

The following code example shows how to use `get-open-id-connect-provider`.

AWS CLI

To return information about the specified OpenID Connect provider

This example returns details about the OpenID Connect provider whose ARN is `arn:aws:iam::123456789012:oidc-provider/server.example.com`.

```
aws iam get-open-id-connect-provider \
  --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com
```

Output:

```
{
  "Url": "server.example.com"
  "CreateDate": "2015-06-16T19:41:48Z",
  "ThumbprintList": [
    "12345abcdefghijk67890lmnopqrst987example"
  ],
  "ClientIDList": [
    "example-application-ID"
  ]
}
```

For more information, see [Creating OpenID Connect \(OIDC\) identity providers](#) in the *AWS IAM User Guide*.

- For API details, see [GetOpenIdConnectProvider](#) in *AWS CLI Command Reference*.

get-organizations-access-report

The following code example shows how to use `get-organizations-access-report`.

AWS CLI

To retrieve an access report

The following `get-organizations-access-report` example displays a previously generated access report for an AWS Organizations entity. To generate a report, use the `generate-organizations-access-report` command.

```
aws iam get-organizations-access-report \  
  --job-id a8b6c06f-aaa4-8xmp-28bc-81da71836359
```

Output:

```
{  
  "JobStatus": "COMPLETED",  
  "JobCreationDate": "2019-09-30T06:53:36.187Z",  
  "JobCompletionDate": "2019-09-30T06:53:37.547Z",  
  "NumberOfServicesAccessible": 188,  
  "NumberOfServicesNotAccessed": 171,  
  "AccessDetails": [  
    {  
      "ServiceName": "Alexa for Business",  
      "ServiceNamespace": "a4b",  
      "TotalAuthenticatedEntities": 0  
    },  
    ...  
  ]  
}
```

For more information, see [Refining permissions in AWS using last accessed information](#) in the *AWS IAM User Guide*.

- For API details, see [GetOrganizationsAccessReport](#) in *AWS CLI Command Reference*.

get-policy-version

The following code example shows how to use `get-policy-version`.

AWS CLI

To retrieve information about the specified version of the specified managed policy

This example returns the policy document for the v2 version of the policy whose ARN is `arn:aws:iam::123456789012:policy/MyManagedPolicy`.

```
aws iam get-policy-version \  
  --policy-arn arn:aws:iam::123456789012:policy/MyManagedPolicy
```

```
--policy-arn arn:aws:iam::123456789012:policy/MyPolicy \  
--version-id v2
```

Output:

```
{  
  "PolicyVersion": {  
    "Document": {  
      "Version": "2012-10-17",  
      "Statement": [  
        {  
          "Effect": "Allow",  
          "Action": "iam:*",  
          "Resource": "*" }  
      ]  
    },  
    "VersionId": "v2",  
    "IsDefaultVersion": true,  
    "CreateDate": "2023-04-11T00:22:54+00:00"  
  }  
}
```

For more information, see [Policies and permissions in IAM](#) in the *AWS IAM User Guide*.

- For API details, see [GetPolicyVersion](#) in *AWS CLI Command Reference*.

get-policy

The following code example shows how to use `get-policy`.

AWS CLI**To retrieve information about the specified managed policy**

This example returns details about the managed policy whose ARN is `arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam get-policy \  
--policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Output:


```
{
  "Policy": {
    "PolicyName": "MySamplePolicy",
    "CreateDate": "2015-06-17T19:23:32Z",
    "AttachmentCount": 0,
    "IsAttachable": true,
    "PolicyId": "Z27SI6FQMGNQ2EXAMPLE1",
    "DefaultVersionId": "v1",
    "Path": "/",
    "Arn": "arn:aws:iam::123456789012:policy/MySamplePolicy",
    "UpdateDate": "2015-06-17T19:23:32Z"
  }
}
```

For more information, see [Policies and permissions in IAM](#) in the *AWS IAM User Guide*.

- For API details, see [GetPolicy](#) in *AWS CLI Command Reference*.

get-role-policy

The following code example shows how to use `get-role-policy`.

AWS CLI

To get information about a policy attached to an IAM role

The following `get-role-policy` command gets information about the specified policy attached to the role named `Test-Role`.

```
aws iam get-role-policy \
  --role-name Test-Role \
  --policy-name ExamplePolicy
```

Output:

```
{
  "RoleName": "Test-Role",
  "PolicyDocument": {
    "Statement": [
      {
        "Action": [
          "s3:ListBucket",
```

```
        "s3:Put*",
        "s3:Get*",
        "s3:*MultipartUpload*"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "1"
  }
]
}
"PolicyName": "ExamplePolicy"
}
```

For more information, see [Creating IAM roles](#) in the *AWS IAM User Guide*.

- For API details, see [GetRolePolicy](#) in *AWS CLI Command Reference*.

get-role

The following code example shows how to use `get-role`.

AWS CLI

To get information about an IAM role

The following `get-role` command gets information about the role named `Test-Role`.

```
aws iam get-role \
  --role-name Test-Role
```

Output:

```
{
  "Role": {
    "Description": "Test Role",
    "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
    "MaxSessionDuration": 3600,
    "RoleId": "AROA1234567890EXAMPLE",
    "CreateDate": "2019-11-13T16:45:56Z",
    "RoleName": "Test-Role",
    "Path": "/",
    "RoleLastUsed": {
```

```
        "Region": "us-east-1",
        "LastUsedDate": "2019-11-13T17:14:00Z"
    },
    "Arn": "arn:aws:iam::123456789012:role/Test-Role"
}
}
```

The command displays the trust policy attached to the role. To list the permissions policies attached to a role, use the `list-role-policies` command.

For more information, see [Creating IAM roles](#) in the *AWS IAM User Guide*.

- For API details, see [GetRole](#) in *AWS CLI Command Reference*.

get-saml-provider

The following code example shows how to use `get-saml-provider`.

AWS CLI

To retrieve the SAML provider metadocument

This example retrieves the details about the SAML 2.0 provider whose ARN is `arn:aws:iam::123456789012:saml-provider/SAMLADFS`. The response includes the metadata document that you got from the identity provider to create the AWS SAML provider entity as well as the creation and expiration dates.

```
aws iam get-saml-provider \
  --saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFS
```

Output:

```
{
  "SAMLMetadataDocument": "...SAMLMetadataDocument-XML...",
  "CreateDate": "2017-03-06T22:29:46+00:00",
  "ValidUntil": "2117-03-06T22:29:46.433000+00:00",
  "Tags": [
    {
      "Key": "DeptID",
      "Value": "123456"
    },
    {
```

```

        "Key": "Department",
        "Value": "Accounting"
    }
]
}

```

For more information, see [Creating IAM SAML identity providers](#) in the *AWS IAM User Guide*.

- For API details, see [GetSamlProvider](#) in *AWS CLI Command Reference*.

get-server-certificate

The following code example shows how to use `get-server-certificate`.

AWS CLI

To get details about a server certificate in your AWS account

The following `get-server-certificate` command retrieves all of the details about the specified server certificate in your AWS account.

```

aws iam get-server-certificate \
  --server-certificate-name myUpdatedServerCertificate

```

Output:

```

{
  "ServerCertificate": {
    "ServerCertificateMetadata": {
      "Path": "/",
      "ServerCertificateName": "myUpdatedServerCertificate",
      "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
      "Arn": "arn:aws:iam::123456789012:server-certificate/myUpdatedServerCertificate",
      "UploadDate": "2019-04-22T21:13:44+00:00",
      "Expiration": "2019-10-15T22:23:16+00:00"
    },
    "CertificateBody": "-----BEGIN CERTIFICATE-----
MIICiTCCAfICCQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAsTC0lBTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWVhZAd
BgkqhkiG9w0BCQEWEG5vb251QGFTYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

```

```

MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCMVVMxCzAJBgNVBAGTAldBMRAwDgYD
VQQHEwdTZWF0dGxLMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWxhZAdBgkqhkiG9w0BCQEWEG5vb251QGFt
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySwTc2XADZ4nB+BLyGVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb30hjZnzcVQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVvXyUntneD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjStb
NYiytVbZPQUQ5Yaxu2jXnimvrszlaEXAMPLE=-----END CERTIFICATE-----",
"CertificateChain": "-----BEGIN CERTIFICATE-----\nMIICiTCcAfICCD6md
7oRw0uX0jANBgkqhkiG9w0BAQQUFADCBiDELMAkGA1UEBhMCMVVMxCzAJBgNVBAGT
AldBMRAwDgYDVQQHEwdTZWF0dGxLMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAs
TC0lBTSBDb25zb2x1MRIwEAYDVQDEw1UZXR0Q21sYWxhZAdBgkqhkiG9w0BCQ
jb20wHhcNMTEwNDI1MjA0NTIxWhtcNMTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBh
MCMVVMxCzAJBgNVBAGTAldBMRAwDgsYDVQQHEwdTZWF0dGxLMQ8wDQYDVQQKEwZBb
WF6b24xFDASBgNVBAsTC0lBTSBDb2d5b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWxh
ZAdBgkqhkiG9w0BCQEWEG5vb251QGFtYXpvbi5jb20wgZ8wDQYJKoZIhvcNAQE
BBQADgY0AMIGJAoGBAMaK0dn+a4GmWIgWJ21uUSfwfEvySwTc2XADZ4nB+BLyGVI
k60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9TrDHudUZg3qX4waLG5M43q7Wgc/MbQ
ITx0USQv7c7ugFFDzQGBzZswY6786m86gjpEIbb30hjZnzcVQAaRHhd1QWIMm2nr
AgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCku4nUhVvXyUntneD9+h8Mg9q6q+auN
KyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0F1kbFFBjvSfpJI1J00zbhNYS5f6Guo
EDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjS;TbNYiytVbZPQUQ5Yaxu2jXnimvw
3rrszlaEWEG5vb251QGFtsYXpvbiEXAMPLE=\n-----END CERTIFICATE-----"
}
}

```

To list the server certificates available in your AWS account, use the `list-server-certificates` command.

For more information, see [Managing server certificates in IAM](#) in the *AWS IAM User Guide*.

- For API details, see [GetServerCertificate](#) in *AWS CLI Command Reference*.

get-service-last-accessed-details-with-entities

The following code example shows how to use `get-service-last-accessed-details-with-entities`.

AWS CLI

To retrieve a service access report with details for a service

The following `get-service-last-accessed-details-with-entities` example retrieves a report that contains details about IAM users and other entities that accessed the specified service. To generate a report, use the `generate-service-last-accessed-details` command. To get a list of services accessed with namespaces, use `get-service-last-accessed-details`.

```
aws iam get-service-last-accessed-details-with-entities \  
  --job-id 78b6c2ba-d09e-6xmp-7039-ecde30b26916 \  
  --service-namespace lambda
```

Output:

```
{  
  "JobStatus": "COMPLETED",  
  "JobCreationDate": "2019-10-01T03:55:41.756Z",  
  "JobCompletionDate": "2019-10-01T03:55:42.533Z",  
  "EntityDetailsList": [  
    {  
      "EntityInfo": {  
        "Arn": "arn:aws:iam::123456789012:user/admin",  
        "Name": "admin",  
        "Type": "USER",  
        "Id": "AIDAI02XMPLNQEEXAMPLE",  
        "Path": "/"  
      },  
      "LastAuthenticated": "2019-09-30T23:02:00Z"  
    },  
    {  
      "EntityInfo": {  
        "Arn": "arn:aws:iam::123456789012:user/developer",  
        "Name": "developer",  
        "Type": "USER",  
        "Id": "AIDAIBEYXMPL2YEXAMPLE",  
        "Path": "/"  
      },  
      "LastAuthenticated": "2019-09-16T19:34:00Z"  
    }  
  ]  
}
```

For more information, see [Refining permissions in AWS using last accessed information](#) in the *AWS IAM User Guide*.

- For API details, see [GetServiceLastAccessedDetailsWithEntities](#) in *AWS CLI Command Reference*.

get-service-last-accessed-details

The following code example shows how to use `get-service-last-accessed-details`.

AWS CLI

To retrieve a service access report

The following `get-service-last-accessed-details` example retrieves a previously generated report that lists the services accessed by IAM entities. To generate a report, use the `generate-service-last-accessed-details` command.

```
aws iam get-service-last-accessed-details \
  --job-id 2eb6c2b8-7b4c-3xmp-3c13-03b72c8cdfdc
```

Output:

```
{
  "JobStatus": "COMPLETED",
  "JobCreationDate": "2019-10-01T03:50:35.929Z",
  "ServicesLastAccessed": [
    ...
    {
      "ServiceName": "AWS Lambda",
      "LastAuthenticated": "2019-09-30T23:02:00Z",
      "ServiceNamespace": "lambda",
      "LastAuthenticatedEntity": "arn:aws:iam::123456789012:user/admin",
      "TotalAuthenticatedEntities": 6
    },
  ]
}
```

For more information, see [Refining permissions in AWS using last accessed information](#) in the *AWS IAM User Guide*.

- For API details, see [GetServiceLastAccessedDetails](#) in *AWS CLI Command Reference*.

get-service-linked-role-deletion-status

The following code example shows how to use `get-service-linked-role-deletion-status`.

AWS CLI

To check the status of a request to delete a service-linked role

The following `get-service-linked-role-deletion-status` example displays the status of a previously request to delete a service-linked role. The delete operation occurs asynchronously. When you make the request, you get a `DeletionTaskId` value that you provide as a parameter for this command.

```
aws iam get-service-linked-role-deletion-status \
  --deletion-task-id task/aws-service-role/lex.amazonaws.com/
  AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE
```

Output:

```
{
  "Status": "SUCCEEDED"
}
```

For more information, see [Using service-linked roles](#) in the *AWS IAM User Guide*.

- For API details, see [GetServiceLinkedRoleDeletionStatus](#) in *AWS CLI Command Reference*.

get-ssh-public-key

The following code example shows how to use `get-ssh-public-key`.

AWS CLI

Example 1: To retrieve an SSH public key attached to an IAM user in SSH encoded form

The following `get-ssh-public-key` command retrieves the specified SSH public key from the IAM user `sofia`. The output is in SSH encoding.

```
aws iam get-ssh-public-key \
  --user-name sofia \
  --ssh-public-key-id APKA123456789EXAMPLE \
```



```
--encoding SSH
```

Output:

```
{
  "SSHPublicKey": {
    "UserName": "sofia",
    "SSHPublicKeyId": "APKA123456789EXAMPLE",
    "Fingerprint": "12:34:56:78:90:ab:cd:ef:12:34:56:78:90:ab:cd:ef",
    "SSHPublicKeyBody": "ssh-rsa <<long encoded SSH string>>",
    "Status": "Inactive",
    "UploadDate": "2019-04-18T17:04:49+00:00"
  }
}
```

Example 2: To retrieve an SSH public key attached to an IAM user in PEM encoded form

The following `get-ssh-public-key` command retrieves the specified SSH public key from the IAM user `sofia`. The output is in PEM encoding.

```
aws iam get-ssh-public-key \
  --user-name sofia \
  --ssh-public-key-id APKA123456789EXAMPLE \
  --encoding PEM
```

Output:

```
{
  "SSHPublicKey": {
    "UserName": "sofia",
    "SSHPublicKeyId": "APKA123456789EXAMPLE",
    "Fingerprint": "12:34:56:78:90:ab:cd:ef:12:34:56:78:90:ab:cd:ef",
    "SSHPublicKeyBody": "'-----BEGIN PUBLIC KEY-----\n<<long encoded PEM string>>\n-----END PUBLIC KEY-----\n'",
    "Status": "Inactive",
    "UploadDate": "2019-04-18T17:04:49+00:00"
  }
}
```

For more information, see [Use SSH keys and SSH with CodeCommit](#) in the *AWS IAM User Guide*.

- For API details, see [GetSshPublicKey](#) in *AWS CLI Command Reference*.

get-user-policy

The following code example shows how to use `get-user-policy`.

AWS CLI

To list policy details for an IAM user

The following `get-user-policy` command lists the details of the specified policy that is attached to the IAM user named Bob.

```
aws iam get-user-policy \  
  --user-name Bob \  
  --policy-name ExamplePolicy
```

Output:

```
{  
  "UserName": "Bob",  
  "PolicyName": "ExamplePolicy",  
  "PolicyDocument": {  
    "Version": "2012-10-17",  
    "Statement": [  
      {  
        "Action": "*",  
        "Resource": "*",  
        "Effect": "Allow"  
      }  
    ]  
  }  
}
```

To get a list of policies for an IAM user, use the `list-user-policies` command.

For more information, see [Policies and permissions in IAM](#) in the *AWS IAM User Guide*.

- For API details, see [GetUserPolicy](#) in *AWS CLI Command Reference*.

get-user

The following code example shows how to use `get-user`.

AWS CLI

To get information about an IAM user

The following `get-user` command gets information about the IAM user named Paulo.

```
aws iam get-user \  
  --user-name Paulo
```

Output:

```
{  
  "User": {  
    "UserName": "Paulo",  
    "Path": "/",  
    "CreateDate": "2019-09-21T23:03:13Z",  
    "UserId": "AIDA123456789EXAMPLE",  
    "Arn": "arn:aws:iam::123456789012:user/Paulo"  
  }  
}
```

For more information, see [Managing IAM users](#) in the *AWS IAM User Guide*.

- For API details, see [GetUser](#) in *AWS CLI Command Reference*.

list-access-keys

The following code example shows how to use `list-access-keys`.

AWS CLI

To list the access key IDs for an IAM user

The following `list-access-keys` command lists the access keys IDs for the IAM user named Bob.

```
aws iam list-access-keys \  
  --user-name Bob
```

Output:

```
{
```

```
"AccessKeyMetadata": [  
  {  
    "UserName": "Bob",  
    "Status": "Active",  
    "CreateDate": "2013-06-04T18:17:34Z",  
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"  
  },  
  {  
    "UserName": "Bob",  
    "Status": "Inactive",  
    "CreateDate": "2013-06-06T20:42:26Z",  
    "AccessKeyId": "AKIAI44QH8DHBEXAMPLE"  
  }  
]  
}
```

You cannot list the secret access keys for IAM users. If the secret access keys are lost, you must create new access keys using the `create-access-keys` command.

For more information, see [Managing access keys for IAM users](#) in the *AWS IAM User Guide*.

- For API details, see [ListAccessKeys](#) in *AWS CLI Command Reference*.

list-account-aliases

The following code example shows how to use `list-account-aliases`.

AWS CLI

To list account aliases

The following `list-account-aliases` command lists the aliases for the current account.

```
aws iam list-account-aliases
```

Output:

```
{  
  "AccountAliases": [  
    "mycompany"  
  ]  
}
```

For more information, see [Your AWS account ID and its alias](#) in the *AWS IAM User Guide*.

- For API details, see [ListAccountAliases](#) in *AWS CLI Command Reference*.

list-attached-group-policies

The following code example shows how to use `list-attached-group-policies`.

AWS CLI

To list all managed policies that are attached to the specified group

This example returns the names and ARNs of the managed policies that are attached to the IAM group named `Admins` in the AWS account.

```
aws iam list-attached-group-policies \  
  --group-name Admins
```

Output:

```
{  
  "AttachedPolicies": [  
    {  
      "PolicyName": "AdministratorAccess",  
      "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"  
    },  
    {  
      "PolicyName": "SecurityAudit",  
      "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"  
    }  
  ],  
  "IsTruncated": false  
}
```

For more information, see [Policies and permissions in IAM](#) in the *AWS IAM User Guide*.

- For API details, see [ListAttachedGroupPolicies](#) in *AWS CLI Command Reference*.

list-attached-role-policies

The following code example shows how to use `list-attached-role-policies`.

AWS CLI

To list all managed policies that are attached to the specified role

This command returns the names and ARNs of the managed policies attached to the IAM role named `SecurityAuditRole` in the AWS account.

```
aws iam list-attached-role-policies \  
  --role-name SecurityAuditRole
```

Output:

```
{  
  "AttachedPolicies": [  
    {  
      "PolicyName": "SecurityAudit",  
      "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"  
    }  
  ],  
  "IsTruncated": false  
}
```

For more information, see [Policies and permissions in IAM](#) in the *AWS IAM User Guide*.

- For API details, see [ListAttachedRolePolicies](#) in *AWS CLI Command Reference*.

list-attached-user-policies

The following code example shows how to use `list-attached-user-policies`.

AWS CLI

To list all managed policies that are attached to the specified user

This command returns the names and ARNs of the managed policies for the IAM user named `Bob` in the AWS account.

```
aws iam list-attached-user-policies \  
  --user-name Bob
```

Output:

```
{
  "AttachedPolicies": [
    {
      "PolicyName": "AdministratorAccess",
      "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
    },
    {
      "PolicyName": "SecurityAudit",
      "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
    }
  ],
  "IsTruncated": false
}
```

For more information, see [Policies and permissions in IAM](#) in the *AWS IAM User Guide*.

- For API details, see [ListAttachedUserPolicies](#) in *AWS CLI Command Reference*.

list-entities-for-policy

The following code example shows how to use `list-entities-for-policy`.

AWS CLI

To list all users, groups, and roles that the specified managed policy is attached to

This example returns a list of IAM groups, roles, and users who have the policy `arn:aws:iam::123456789012:policy/TestPolicy` attached.

```
aws iam list-entities-for-policy \
  --policy-arn arn:aws:iam::123456789012:policy/TestPolicy
```

Output:

```
{
  "PolicyGroups": [
    {
      "GroupName": "Admins",
      "GroupId": "AGPACKCEVSQ6C2EXAMPLE"
    }
  ],
}
```

```
"PolicyUsers": [
  {
    "UserName": "Alice",
    "UserId": "AIDACKCEVSQ6C2EXAMPLE"
  }
],
"PolicyRoles": [
  {
    "RoleName": "DevRole",
    "RoleId": "AR0ADBQP57FF2AEXAMPLE"
  }
],
"IsTruncated": false
}
```

For more information, see [Policies and permissions in IAM](#) in the *AWS IAM User Guide*.

- For API details, see [ListEntitiesForPolicy](#) in *AWS CLI Command Reference*.

list-group-policies

The following code example shows how to use `list-group-policies`.

AWS CLI

To list all inline policies that are attached to the specified group

The following `list-group-policies` command lists the names of inline policies that are attached to the IAM group named `Admins` in the current account.

```
aws iam list-group-policies \
  --group-name Admins
```

Output:

```
{
  "PolicyNames": [
    "AdminRoot",
    "ExamplePolicy"
  ]
}
```


For more information, see [Managing IAM policies](#) in the *AWS IAM User Guide*.

- For API details, see [ListGroupPolicies](#) in *AWS CLI Command Reference*.

list-groups-for-user

The following code example shows how to use `list-groups-for-user`.

AWS CLI

To list the groups that an IAM user belongs to

The following `list-groups-for-user` command displays the groups that the IAM user named Bob belongs to.

```
aws iam list-groups-for-user \  
  --user-name Bob
```

Output:

```
{  
  "Groups": [  
    {  
      "Path": "/",  
      "CreateDate": "2013-05-06T01:18:08Z",  
      "GroupId": "AKIAIOSFODNN7EXAMPLE",  
      "Arn": "arn:aws:iam::123456789012:group/Admin",  
      "GroupName": "Admin"  
    },  
    {  
      "Path": "/",  
      "CreateDate": "2013-05-06T01:37:28Z",  
      "GroupId": "AKIAI44QH8DHBEXAMPLE",  
      "Arn": "arn:aws:iam::123456789012:group/s3-Users",  
      "GroupName": "s3-Users"  
    }  
  ]  
}
```

For more information, see [Managing IAM user groups](#) in the *AWS IAM User Guide*.

- For API details, see [ListGroupsForUser](#) in *AWS CLI Command Reference*.

list-groups

The following code example shows how to use `list-groups`.

AWS CLI

To list the IAM groups for the current account

The following `list-groups` command lists the IAM groups in the current account.

```
aws iam list-groups
```

Output:

```
{
  "Groups": [
    {
      "Path": "/",
      "CreateDate": "2013-06-04T20:27:27.972Z",
      "GroupId": "AIDACKCEVSQ6C2EXAMPLE",
      "Arn": "arn:aws:iam::123456789012:group/Admins",
      "GroupName": "Admins"
    },
    {
      "Path": "/",
      "CreateDate": "2013-04-16T20:30:42Z",
      "GroupId": "AIDGPM9R04H3FEXAMPLE",
      "Arn": "arn:aws:iam::123456789012:group/S3-Admins",
      "GroupName": "S3-Admins"
    }
  ]
}
```

For more information, see [Managing IAM user groups](#) in the *AWS IAM User Guide*.

- For API details, see [ListGroups](#) in *AWS CLI Command Reference*.

list-instance-profile-tags

The following code example shows how to use `list-instance-profile-tags`.

AWS CLI

To list the tags attached to an instance profile

The following `list-instance-profile-tags` command retrieves the list of tags associated with the specified instance profile.

```
aws iam list-instance-profile-tags \  
  --instance-profile-name deployment-role
```

Output:

```
{  
  "Tags": [  
    {  
      "Key": "DeptID",  
      "Value": "123456"  
    },  
    {  
      "Key": "Department",  
      "Value": "Accounting"  
    }  
  ]  
}
```

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [ListInstanceProfileTags](#) in *AWS CLI Command Reference*.

list-instance-profiles-for-role

The following code example shows how to use `list-instance-profiles-for-role`.

AWS CLI

To list the instance profiles for an IAM role

The following `list-instance-profiles-for-role` command lists the instance profiles that are associated with the role `Test-Role`.

```
aws iam list-instance-profiles-for-role \  
  --role-name Test-Role
```

Output:

```
{
  "InstanceProfiles": [
    {
      "InstanceId": "AIDGPM59R04H3FEXAMPLE",
      "Roles": [
        {
          "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
          "RoleId": "AIDACKCEVSQ6C2EXAMPLE",
          "CreateDate": "2013-06-07T20:42:15Z",
          "RoleName": "Test-Role",
          "Path": "/",
          "Arn": "arn:aws:iam::123456789012:role/Test-Role"
        }
      ],
      "CreateDate": "2013-06-07T21:05:24Z",
      "InstanceProfileName": "ExampleInstanceProfile",
      "Path": "/",
      "Arn": "arn:aws:iam::123456789012:instance-profile/
ExampleInstanceProfile"
    }
  ]
}
```

For more information, see [Using instance profiles](#) in the *AWS IAM User Guide*.

- For API details, see [ListInstanceProfilesForRole](#) in *AWS CLI Command Reference*.

list-instance-profiles

The following code example shows how to use `list-instance-profiles`.

AWS CLI**To lists the instance profiles for the account**

The following `list-instance-profiles` command lists the instance profiles that are associated with the current account.

```
aws iam list-instance-profiles
```

Output:

```
{
  "InstanceProfiles": [
    {
      "Path": "/",
      "InstanceProfileName": "example-dev-role",
      "InstanceProfileId": "AIPAIXEU4NUHUPEXAMPLE",
      "Arn": "arn:aws:iam::123456789012:instance-profile/example-dev-role",
      "CreateDate": "2023-09-21T18:17:41+00:00",
      "Roles": [
        {
          "Path": "/",
          "RoleName": "example-dev-role",
          "RoleId": "AR0AJ520TH4H7LEXAMPLE",
          "Arn": "arn:aws:iam::123456789012:role/example-dev-role",
          "CreateDate": "2023-09-21T18:17:40+00:00",
          "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
              {
                "Effect": "Allow",
                "Principal": {
                  "Service": "ec2.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
              }
            ]
          }
        }
      ]
    }
  ],
  {
    "Path": "/",
    "InstanceProfileName": "example-s3-role",
    "InstanceProfileId": "AIPAJVJVNRIQFREXAMPLE",
    "Arn": "arn:aws:iam::123456789012:instance-profile/example-s3-role",
    "CreateDate": "2023-09-21T18:18:50+00:00",
    "Roles": [
      {
        "Path": "/",
        "RoleName": "example-s3-role",
        "RoleId": "AR0AINUBC507XLEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:role/example-s3-role",
        "CreateDate": "2023-09-21T18:18:49+00:00",
```

```
    "AssumeRolePolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Principal": {
            "Service": "ec2.amazonaws.com"
          },
          "Action": "sts:AssumeRole"
        }
      ]
    }
  ]
}
```

For more information, see [Using instance profiles](#) in the *AWS IAM User Guide*.

- For API details, see [ListInstanceProfiles](#) in *AWS CLI Command Reference*.

list-mfa-device-tags

The following code example shows how to use `list-mfa-device-tags`.

AWS CLI

To list the tags attached to an MFA device

The following `list-mfa-device-tags` command retrieves the list of tags associated with the specified MFA device.

```
aws iam list-mfa-device-tags \
  --serial-number arn:aws:iam::123456789012:mfa/alice
```

Output:

```
{
  "Tags": [
    {
      "Key": "DeptID",
```

```
        "Value": "123456"
      },
      {
        "Key": "Department",
        "Value": "Accounting"
      }
    ]
  }
}
```

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [ListMfaDeviceTags](#) in *AWS CLI Command Reference*.

list-mfa-devices

The following code example shows how to use `list-mfa-devices`.

AWS CLI

To list all MFA devices for a specified user

This example returns details about the MFA device assigned to the IAM user Bob.

```
aws iam list-mfa-devices \
  --user-name Bob
```

Output:

```
{
  "MFADevices": [
    {
      "UserName": "Bob",
      "SerialNumber": "arn:aws:iam::123456789012:mfa/Bob",
      "EnableDate": "2019-10-28T20:37:09+00:00"
    },
    {
      "UserName": "Bob",
      "SerialNumber": "GAKT12345678",
      "EnableDate": "2023-02-18T21:44:42+00:00"
    },
    {
      "UserName": "Bob",
```

```

        "SerialNumber": "arn:aws:iam::123456789012:u2f/user/Bob/
fidosecuritykey1-7XNL7NFNLZ123456789EXAMPLE",
        "EnableDate": "2023-09-19T02:25:35+00:00"
    },
    {
        "UserName": "Bob",
        "SerialNumber": "arn:aws:iam::123456789012:u2f/user/Bob/
fidosecuritykey2-VDRQTDBBN5123456789EXAMPLE",
        "EnableDate": "2023-09-19T01:49:18+00:00"
    }
]
}

```

For more information, see [Using multi-factor authentication \(MFA\) in AWS](#) in the *AWS IAM User Guide*.

- For API details, see [ListMfaDevices](#) in *AWS CLI Command Reference*.

list-open-id-connect-provider-tags

The following code example shows how to use `list-open-id-connect-provider-tags`.

AWS CLI

To list the tags attached to an OpenID Connect (OIDC)-compatible identity provider

The following `list-open-id-connect-provider-tags` command retrieves the list of tags associated with the specified OIDC identity provider.

```

aws iam list-open-id-connect-provider-tags \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com

```

Output:

```

{
  "Tags": [
    {
      "Key": "DeptID",
      "Value": "123456"
    },
    {

```



```
        "Key": "Department",
        "Value": "Accounting"
    }
]
}
```

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [ListOpenIdConnectProviderTags](#) in *AWS CLI Command Reference*.

list-open-id-connect-providers

The following code example shows how to use `list-open-id-connect-providers`.

AWS CLI

To list information about the OpenID Connect providers in the AWS account

This example returns a list of ARNS of all the OpenID Connect providers that are defined in the current AWS account.

```
aws iam list-open-id-connect-providers
```

Output:

```
{
  "OpenIDConnectProviderList": [
    {
      "Arn": "arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com"
    }
  ]
}
```

For more information, see [Creating OpenID Connect \(OIDC\) identity providers](#) in the *AWS IAM User Guide*.

- For API details, see [ListOpenIdConnectProviders](#) in *AWS CLI Command Reference*.

list-policies-granting-service-access

The following code example shows how to use `list-policies-granting-service-access`.

AWS CLI

To list the policies that grant a principal access to the specified service

The following `list-policies-granting-service-access` example retrieves the list of policies that grant the IAM user `sofia` access to AWS CodeCommit service.

```
aws iam list-policies-granting-service-access \
  --arn arn:aws:iam::123456789012:user/sofia \
  --service-namespaces codecommit
```

Output:

```
{
  "PoliciesGrantingServiceAccess": [
    {
      "ServiceNamespace": "codecommit",
      "Policies": [
        {
          "PolicyName": "Grant-Sofia-Access-To-CodeCommit",
          "PolicyType": "INLINE",
          "EntityType": "USER",
          "EntityName": "sofia"
        }
      ]
    }
  ],
  "IsTruncated": false
}
```

For more information, see [Using IAM with CodeCommit: Git credentials, SSH keys, and AWS access keys](#) in the *AWS IAM User Guide*.

- For API details, see [ListPoliciesGrantingServiceAccess](#) in *AWS CLI Command Reference*.

list-policies

The following code example shows how to use `list-policies`.

AWS CLI

To list managed policies that are available to your AWS account

This example returns a collection of the first two managed policies available in the current AWS account.

```
aws iam list-policies \  
  --max-items 3
```

Output:

```
{  
  "Policies": [  
    {  
      "PolicyName": "AWSCloudTrailAccessPolicy",  
      "PolicyId": "ANPAXQE2B5PJ7YEXAMPLE",  
      "Arn": "arn:aws:iam::123456789012:policy/AWSCloudTrailAccessPolicy",  
      "Path": "/",  
      "DefaultVersionId": "v1",  
      "AttachmentCount": 0,  
      "PermissionsBoundaryUsageCount": 0,  
      "IsAttachable": true,  
      "CreateDate": "2019-09-04T17:43:42+00:00",  
      "UpdateDate": "2019-09-04T17:43:42+00:00"  
    },  
    {  
      "PolicyName": "AdministratorAccess",  
      "PolicyId": "ANPAIWMBCKSKIEE64ZLYK",  
      "Arn": "arn:aws:iam::aws:policy/AdministratorAccess",  
      "Path": "/",  
      "DefaultVersionId": "v1",  
      "AttachmentCount": 6,  
      "PermissionsBoundaryUsageCount": 0,  
      "IsAttachable": true,  
      "CreateDate": "2015-02-06T18:39:46+00:00",  
      "UpdateDate": "2015-02-06T18:39:46+00:00"  
    },  
    {  
      "PolicyName": "PowerUserAccess",  
      "PolicyId": "ANPAJYRXTHIB4F0VS3ZXS",  
      "Arn": "arn:aws:iam::aws:policy/PowerUserAccess",  
      "Path": "/",  
      "DefaultVersionId": "v5",  
      "AttachmentCount": 1,  
      "PermissionsBoundaryUsageCount": 0,  
      "IsAttachable": true,  
    }  
  ]  
}
```

```
        "CreateDate": "2015-02-06T18:39:47+00:00",
        "UpdateDate": "2023-07-06T22:04:00+00:00"
    }
],
"NextToken": "EXAMPLErZXIi0iBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQi0iA4fQ=="
}
```

For more information, see [Policies and permissions in IAM](#) in the *AWS IAM User Guide*.

- For API details, see [ListPolicies](#) in *AWS CLI Command Reference*.

list-policy-tags

The following code example shows how to use `list-policy-tags`.

AWS CLI

To list the tags attached to a managed policy

The following `list-policy-tags` command retrieves the list of tags associated with the specified managed policy.

```
aws iam list-policy-tags \
  --policy-arn arn:aws:iam::123456789012:policy/billing-access
```

Output:

```
{
  "Tags": [
    {
      "Key": "DeptID",
      "Value": "123456"
    },
    {
      "Key": "Department",
      "Value": "Accounting"
    }
  ]
}
```

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [ListPolicyTags](#) in *AWS CLI Command Reference*.

list-policy-versions

The following code example shows how to use `list-policy-versions`.

AWS CLI

To list information about the versions of the specified managed policy

This example returns the list of available versions of the policy whose ARN is `arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam list-policy-versions \  
  --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Output:

```
{  
  "IsTruncated": false,  
  "Versions": [  
    {  
      "VersionId": "v2",  
      "IsDefaultVersion": true,  
      "CreateDate": "2015-06-02T23:19:44Z"  
    },  
    {  
      "VersionId": "v1",  
      "IsDefaultVersion": false,  
      "CreateDate": "2015-06-02T22:30:47Z"  
    }  
  ]  
}
```

For more information, see [Policies and permissions in IAM](#) in the *AWS IAM User Guide*.

- For API details, see [ListPolicyVersions](#) in *AWS CLI Command Reference*.

list-role-policies

The following code example shows how to use `list-role-policies`.

AWS CLI

To list the policies attached to an IAM role

The following `list-role-policies` command lists the names of the permissions policies for the specified IAM role.

```
aws iam list-role-policies \  
  --role-name Test-Role
```

Output:

```
{  
  "PolicyNames": [  
    "ExamplePolicy"  
  ]  
}
```

To see the trust policy attached to a role, use the `get-role` command. To see the details of a permissions policy, use the `get-role-policy` command.

For more information, see [Creating IAM roles](#) in the *AWS IAM User Guide*.

- For API details, see [ListRolePolicies](#) in *AWS CLI Command Reference*.

list-role-tags

The following code example shows how to use `list-role-tags`.

AWS CLI

To list the tags attached to a role

The following `list-role-tags` command retrieves the list of tags associated with the specified role.

```
aws iam list-role-tags \  
  --role-name production-role
```

Output:

```
{  
  "Tags": [  
    {  
      "Key": "Department",  
      "Value": "Accounting"  
    }  
  ]  
}
```

```
    },
    {
      "Key": "DeptID",
      "Value": "12345"
    }
  ],
  "IsTruncated": false
}
```

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [ListRoleTags](#) in *AWS CLI Command Reference*.

list-roles

The following code example shows how to use `list-roles`.

AWS CLI

To list IAM roles for the current account

The following `list-roles` command lists IAM roles for the current account.

```
aws iam list-roles
```

Output:

```
{
  "Roles": [
    {
      "Path": "/",
      "RoleName": "ExampleRole",
      "RoleId": "AR0AJ520TH4H7LEXAMPLE",
      "Arn": "arn:aws:iam::123456789012:role/ExampleRole",
      "CreateDate": "2017-09-12T19:23:36+00:00",
      "AssumeRolePolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
              "Service": "ec2.amazonaws.com"
            }
          }
        ]
      }
    }
  ]
}
```

```

        },
        "Action": "sts:AssumeRole"
    }
]
},
"MaxSessionDuration": 3600
},
{
    "Path": "/example_path/",
    "RoleName": "ExampleRoleWithPath",
    "RoleId": "AROAI4QRP7UFT7EXAMPLE",
    "Arn": "arn:aws:iam::123456789012:role/example_path/
ExampleRoleWithPath",
    "CreateDate": "2023-09-21T20:29:38+00:00",
    "AssumeRolePolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Sid": "",
                "Effect": "Allow",
                "Principal": {
                    "Service": "ec2.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
            }
        ]
    },
    "MaxSessionDuration": 3600
}
]
}

```

For more information, see [Creating IAM roles](#) in the *AWS IAM User Guide*.

- For API details, see [ListRoles](#) in *AWS CLI Command Reference*.

list-saml-provider-tags

The following code example shows how to use `list-saml-provider-tags`.

AWS CLI

To list the tags attached to a SAML provider

The following `list-saml-provider-tags` command retrieves the list of tags associated with the specified SAML provider.

```
aws iam list-saml-provider-tags \  
  --saml-provider-arn arn:aws:iam::123456789012:saml-provider/ADFS
```

Output:

```
{  
  "Tags": [  
    {  
      "Key": "DeptID",  
      "Value": "123456"  
    },  
    {  
      "Key": "Department",  
      "Value": "Accounting"  
    }  
  ]  
}
```

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [ListSamlProviderTags](#) in *AWS CLI Command Reference*.

list-saml-providers

The following code example shows how to use `list-saml-providers`.

AWS CLI

To list the SAML providers in the AWS account

This example retrieves the list of SAML 2.0 providers created in the current AWS account.

```
aws iam list-saml-providers
```

Output:

```
{  
  "SAMLProviderList": [  
    {  
      "Name": "ADFS",  
      "CreateDate": "2017-01-01T00:00:00Z",  
      "LastModifiedDate": "2017-01-01T00:00:00Z",  
      "Status": "Active",  
      "Type": "SAML20"  
    }  
  ]  
}
```

```
{
  "Arn": "arn:aws:iam::123456789012:saml-provider/SAML-ADFS",
  "ValidUntil": "2015-06-05T22:45:14Z",
  "CreateDate": "2015-06-05T22:45:14Z"
}
]
```

For more information, see [Creating IAM SAML identity providers](#) in the *AWS IAM User Guide*.

- For API details, see [ListSAMLProviders](#) in *AWS CLI Command Reference*.

list-server-certificate-tags

The following code example shows how to use `list-server-certificate-tags`.

AWS CLI

To list the tags attached to a server certificate

The following `list-server-certificate-tags` command retrieves the list of tags associated with the specified server certificate.

```
aws iam list-server-certificate-tags \
  --server-certificate-name ExampleCertificate
```

Output:

```
{
  "Tags": [
    {
      "Key": "DeptID",
      "Value": "123456"
    },
    {
      "Key": "Department",
      "Value": "Accounting"
    }
  ]
}
```

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [ListServerCertificateTags](#) in *AWS CLI Command Reference*.

list-server-certificates

The following code example shows how to use `list-server-certificates`.

AWS CLI

To list the server certificates in your AWS account

The following `list-server-certificates` command lists all of the server certificates stored and available for use in your AWS account.

```
aws iam list-server-certificates
```

Output:

```
{
  "ServerCertificateMetadataList": [
    {
      "Path": "/",
      "ServerCertificateName": "myUpdatedServerCertificate",
      "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
      "Arn": "arn:aws:iam::123456789012:server-certificate/myUpdatedServerCertificate",
      "UploadDate": "2019-04-22T21:13:44+00:00",
      "Expiration": "2019-10-15T22:23:16+00:00"
    },
    {
      "Path": "/cloudfront/",
      "ServerCertificateName": "MyTestCert",
      "ServerCertificateId": "ASCAEXAMPLE456EXAMPLE",
      "Arn": "arn:aws:iam::123456789012:server-certificate/Org1/Org2/MyTestCert",
      "UploadDate": "2015-04-21T18:14:16+00:00",
      "Expiration": "2018-01-14T17:52:36+00:00"
    }
  ]
}
```

For more information, see [Managing server certificates in IAM](#) in the *AWS IAM User Guide*.

- For API details, see [ListServerCertificates](#) in *AWS CLI Command Reference*.

list-service-specific-credential

The following code example shows how to use `list-service-specific-credential`.

AWS CLI

Example 1: List the service-specific credentials for a user

The following `list-service-specific-credentials` example displays all service-specific credentials assigned to the specified user. Passwords are not included in the response.

```
aws iam list-service-specific-credentials \  
  --user-name sofia
```

Output:

```
{  
  "ServiceSpecificCredential": {  
    "CreateDate": "2019-04-18T20:45:36+00:00",  
    "ServiceName": "codecommit.amazonaws.com",  
    "ServiceUserName": "sofia-at-123456789012",  
    "ServiceSpecificCredentialId": "ACCAEXAMPLE123EXAMPLE",  
    "UserName": "sofia",  
    "Status": "Active"  
  }  
}
```

Example 2: List the service-specific credentials for a user filtered to a specified service

The following `list-service-specific-credentials` example displays the service-specific credentials assigned to the user making the request. The list is filtered to include only those credentials for the specified service. Passwords are not included in the response.

```
aws iam list-service-specific-credentials \  
  --service-name codecommit.amazonaws.com
```

Output:

```
{  
  "ServiceSpecificCredential": {  
    "CreateDate": "2019-04-18T20:45:36+00:00",
```

```
    "ServiceName": "codecommit.amazonaws.com",
    "ServiceUserName": "sofia-at-123456789012",
    "ServiceSpecificCredentialId": "ACCAEXAMPLE123EXAMPLE",
    "UserName": "sofia",
    "Status": "Active"
  }
}
```

For more information, see [Create Git credentials for HTTPS connections to CodeCommit](#) in the *AWS CodeCommit User Guide*.

- For API details, see [ListServiceSpecificCredential](#) in *AWS CLI Command Reference*.

list-service-specific-credentials

The following code example shows how to use `list-service-specific-credentials`.

AWS CLI

To retrieve a list of credentials

The following `list-service-specific-credentials` example lists the credentials generated for HTTPS access to AWS CodeCommit repositories for a user named `developer`.

```
aws iam list-service-specific-credentials \
  --user-name developer \
  --service-name codecommit.amazonaws.com
```

Output:

```
{
  "ServiceSpecificCredentials": [
    {
      "UserName": "developer",
      "Status": "Inactive",
      "ServiceUserName": "developer-at-123456789012",
      "CreateDate": "2019-10-01T04:31:41Z",
      "ServiceSpecificCredentialId": "ACCAQFODXMPL4YFHP7DZE",
      "ServiceName": "codecommit.amazonaws.com"
    },
    {
      "UserName": "developer",
```

```

        "Status": "Active",
        "ServiceUserName": "developer+1-at-123456789012",
        "CreateDate": "2019-10-01T04:31:45Z",
        "ServiceSpecificCredentialId": "ACCAQFOX MPL6VW57M7AJP",
        "ServiceName": "codecommit.amazonaws.com"
    }
]
}

```

For more information, see [Create Git credentials for HTTPS connections to CodeCommit](#) in the *AWS CodeCommit User Guide*.

- For API details, see [ListServiceSpecificCredentials](#) in *AWS CLI Command Reference*.

list-signing-certificates

The following code example shows how to use `list-signing-certificates`.

AWS CLI

To list the signing certificates for an IAM user

The following `list-signing-certificates` command lists the signing certificates for the IAM user named Bob.

```
aws iam list-signing-certificates \
    --user-name Bob
```

Output:

```

{
  "Certificates": [
    {
      "UserName": "Bob",
      "Status": "Inactive",
      "CertificateBody": "-----BEGIN CERTIFICATE-----<certificate-body>-----
END CERTIFICATE-----",
      "CertificateId": "TA7SMP42TDN5Z260BPJE7EXAMPLE",
      "UploadDate": "2013-06-06T21:40:08Z"
    }
  ]
}

```

For more information, see [Manage signing certificates](#) in the *Amazon EC2 User Guide*.

- For API details, see [ListSigningCertificates](#) in *AWS CLI Command Reference*.

list-ssh-public-keys

The following code example shows how to use `list-ssh-public-keys`.

AWS CLI

To list the SSH public keys attached to an IAM user

The following `list-ssh-public-keys` example lists the SSH public keys attached to the IAM user `sofia`.

```
aws iam list-ssh-public-keys \  
  --user-name sofia
```

Output:

```
{  
  "SSHPublicKeys": [  
    {  
      "UserName": "sofia",  
      "SSHPublicKeyId": "APKA1234567890EXAMPLE",  
      "Status": "Inactive",  
      "UploadDate": "2019-04-18T17:04:49+00:00"  
    }  
  ]  
}
```

For more information, see [Use SSH keys and SSH with CodeCommit](#) in the *AWS IAM User Guide*

- For API details, see [ListSshPublicKeys](#) in *AWS CLI Command Reference*.

list-user-policies

The following code example shows how to use `list-user-policies`.

AWS CLI

To list policies for an IAM user

The following `list-user-policies` command lists the policies that are attached to the IAM user named Bob.

```
aws iam list-user-policies \  
  --user-name Bob
```

Output:

```
{  
  "PolicyNames": [  
    "ExamplePolicy",  
    "TestPolicy"  
  ]  
}
```

For more information, see [Creating an IAM user in your AWS account](#) in the *AWS IAM User Guide*.

- For API details, see [ListUserPolicies](#) in *AWS CLI Command Reference*.

list-user-tags

The following code example shows how to use `list-user-tags`.

AWS CLI

To list the tags attached to a user

The following `list-user-tags` command retrieves the list of tags associated with the specified IAM user.

```
aws iam list-user-tags \  
  --user-name alice
```

Output:

```
{  
  "Tags": [  
    {  
      "Key": "Department",  
      "Value": "Accounting"  
    },  
    {
```



```
        "Key": "DeptID",
        "Value": "12345"
    }
],
"IsTruncated": false
}
```

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [ListUserTags](#) in *AWS CLI Command Reference*.

list-users

The following code example shows how to use `list-users`.

AWS CLI

To list IAM users

The following `list-users` command lists the IAM users in the current account.

```
aws iam list-users
```

Output:

```
{
  "Users": [
    {
      "UserName": "Adele",
      "Path": "/",
      "CreateDate": "2013-03-07T05:14:48Z",
      "UserId": "AKIAI44QH8DHBEXAMPLE",
      "Arn": "arn:aws:iam::123456789012:user/Adele"
    },
    {
      "UserName": "Bob",
      "Path": "/",
      "CreateDate": "2012-09-21T23:03:13Z",
      "UserId": "AKIAIOSFODNN7EXAMPLE",
      "Arn": "arn:aws:iam::123456789012:user/Bob"
    }
  ]
}
```

For more information, see [Listing IAM users](#) in the *AWS IAM User Guide*.

- For API details, see [ListUsers](#) in *AWS CLI Command Reference*.

list-virtual-mfa-devices

The following code example shows how to use `list-virtual-mfa-devices`.

AWS CLI

To list virtual MFA devices

The following `list-virtual-mfa-devices` command lists the virtual MFA devices that have been configured for the current account.

```
aws iam list-virtual-mfa-devices
```

Output:

```
{
  "VirtualMFADevices": [
    {
      "SerialNumber": "arn:aws:iam::123456789012:mfa/ExampleMFADevice"
    },
    {
      "SerialNumber": "arn:aws:iam::123456789012:mfa/Fred"
    }
  ]
}
```

For more information, see [Enabling a virtual multi-factor authentication \(MFA\) device](#) in the *AWS IAM User Guide*.

- For API details, see [ListVirtualMfaDevices](#) in *AWS CLI Command Reference*.

put-group-policy

The following code example shows how to use `put-group-policy`.

AWS CLI

To add a policy to a group

The following `put-group-policy` command adds a policy to the IAM group named Admins.

```
aws iam put-group-policy \  
  --group-name Admins \  
  --policy-document file://AdminPolicy.json \  
  --policy-name AdminRoot
```

This command produces no output.

The policy is defined as a JSON document in the `AdminPolicy.json` file. (The file name and extension do not have significance.)

For more information, see [Managing IAM policies](#) in the *AWS IAM User Guide*.

- For API details, see [PutGroupPolicy](#) in *AWS CLI Command Reference*.

put-role-permissions-boundary

The following code example shows how to use `put-role-permissions-boundary`.

AWS CLI

Example 1: To apply a permissions boundary based on a custom policy to an IAM role

The following `put-role-permissions-boundary` example applies the custom policy named `intern-boundary` as the permissions boundary for the specified IAM role.

```
aws iam put-role-permissions-boundary \  
  --permissions-boundary arn:aws:iam::123456789012:policy/intern-boundary \  
  --role-name lambda-application-role
```

This command produces no output.

Example 2: To apply a permissions boundary based on an AWS managed policy to an IAM role

The following `put-role-permissions-boundary` example applies the AWS managed `PowerUserAccess` policy as the permissions boundary for the specified IAM role.

```
aws iam put-role-permissions-boundary \  
  --permissions-boundary arn:aws:iam::aws:policy/PowerUserAccess \  
  --role-name lambda-application-role
```

```
--role-name x-account-admin
```

This command produces no output.

For more information, see [Modifying a role](#) in the *AWS IAM User Guide*.

- For API details, see [PutRolePermissionsBoundary](#) in *AWS CLI Command Reference*.

put-role-policy

The following code example shows how to use `put-role-policy`.

AWS CLI

To attach a permissions policy to an IAM role

The following `put-role-policy` command adds a permissions policy to the role named `Test-Role`.

```
aws iam put-role-policy \  
  --role-name Test-Role \  
  --policy-name ExamplePolicy \  
  --policy-document file://AdminPolicy.json
```

This command produces no output.

The policy is defined as a JSON document in the *AdminPolicy.json* file. (The file name and extension do not have significance.)

To attach a trust policy to a role, use the `update-assume-role-policy` command.

For more information, see [Modifying a role](#) in the *AWS IAM User Guide*.

- For API details, see [PutRolePolicy](#) in *AWS CLI Command Reference*.

put-user-permissions-boundary

The following code example shows how to use `put-user-permissions-boundary`.

AWS CLI

Example 1: To apply a permissions boundary based on a custom policy to an IAM user

The following `put-user-permissions-boundary` example applies a custom policy named `intern-boundary` as the permissions boundary for the specified IAM user.

```
aws iam put-user-permissions-boundary \  
  --permissions-boundary arn:aws:iam::123456789012:policy/intern-boundary \  
  --user-name intern
```

This command produces no output.

Example 2: To apply a permissions boundary based on an AWS managed policy to an IAM user

The following `put-user-permissions-boundary` example applies the AWS managed policy named `PowerUserAccess` as the permissions boundary for the specified IAM user.

```
aws iam put-user-permissions-boundary \  
  --permissions-boundary arn:aws:iam::aws:policy/PowerUserAccess \  
  --user-name developer
```

This command produces no output.

For more information, see [Adding and removing IAM identity permissions](#) in the *AWS IAM User Guide*.

- For API details, see [PutUserPermissionsBoundary](#) in *AWS CLI Command Reference*.

put-user-policy

The following code example shows how to use `put-user-policy`.

AWS CLI

To attach a policy to an IAM user

The following `put-user-policy` command attaches a policy to the IAM user named Bob.

```
aws iam put-user-policy \  
  --user-name Bob \  
  --policy-name ExamplePolicy \  
  --policy-document file://AdminPolicy.json
```

This command produces no output.

The policy is defined as a JSON document in the *AdminPolicy.json* file. (The file name and extension do not have significance.)

For more information, see [Adding and removing IAM identity permissions](#) in the *AWS IAM User Guide*.

- For API details, see [PutUserPolicy](#) in *AWS CLI Command Reference*.

remove-client-id-from-open-id-connect-provider

The following code example shows how to use `remove-client-id-from-open-id-connect-provider`.

AWS CLI

To remove the specified client ID from the list of client IDs registered for the specified IAM OpenID Connect provider

This example removes the client ID `My-TestApp-3` from the list of client IDs associated with the IAM OIDC provider whose ARN is `arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`.

```
aws iam remove-client-id-from-open-id-connect-provider
  --client-id My-TestApp-3 \
  --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

This command produces no output.

For more information, see [Creating OpenID Connect \(OIDC\) identity providers](#) in the *AWS IAM User Guide*.

- For API details, see [RemoveClientIdFromOpenIdConnectProvider](#) in *AWS CLI Command Reference*.

remove-role-from-instance-profile

The following code example shows how to use `remove-role-from-instance-profile`.

AWS CLI

To remove a role from an instance profile

The following `remove-role-from-instance-profile` command removes the role named `Test-Role` from the instance profile named `ExampleInstanceProfile`.

```
aws iam remove-role-from-instance-profile \  
  --instance-profile-name ExampleInstanceProfile \  
  --role-name Test-Role
```

For more information, see [Using instance profiles](#) in the *AWS IAM User Guide*.

- For API details, see [RemoveRoleFromInstanceProfile](#) in *AWS CLI Command Reference*.

remove-user-from-group

The following code example shows how to use `remove-user-from-group`.

AWS CLI

To remove a user from an IAM group

The following `remove-user-from-group` command removes the user named `Bob` from the IAM group named `Admins`.

```
aws iam remove-user-from-group \  
  --user-name Bob \  
  --group-name Admins
```

This command produces no output.

For more information, see [Adding and removing users in an IAM user group](#) in the *AWS IAM User Guide*.

- For API details, see [RemoveUserFromGroup](#) in *AWS CLI Command Reference*.

reset-service-specific-credential

The following code example shows how to use `reset-service-specific-credential`.

AWS CLI

Example 1: Reset the password for a service-specific credential attached to the user making the request

The following `reset-service-specific-credential` example generates a new cryptographically strong password for the specified service-specific credential attached to the user making the request.

```
aws iam reset-service-specific-credential \  
  --service-specific-credential-id ACCAEXAMPLE123EXAMPLE
```

Output:

```
{  
  "ServiceSpecificCredential": {  
    "CreateDate": "2019-04-18T20:45:36+00:00",  
    "ServiceName": "codecommit.amazonaws.com",  
    "ServiceUserName": "sofia-at-123456789012",  
    "ServicePassword": "+oaFsNk7tLco+C/obP9Ghhc0zGcK0ayTmE3LnAmAmH4=",  
    "ServiceSpecificCredentialId": "ACCAEXAMPLE123EXAMPLE",  
    "UserName": "sofia",  
    "Status": "Active"  
  }  
}
```

Example 2: Reset the password for a service-specific credential attached to a specified user

The following `reset-service-specific-credential` example generates a new cryptographically strong password for a service-specific credential attached to the specified user.

```
aws iam reset-service-specific-credential \  
  --user-name sofia \  
  --service-specific-credential-id ACCAEXAMPLE123EXAMPLE
```

Output:

```
{  
  "ServiceSpecificCredential": {  
    "CreateDate": "2019-04-18T20:45:36+00:00",  
    "ServiceName": "codecommit.amazonaws.com",  
    "ServiceUserName": "sofia-at-123456789012",  
    "ServicePassword": "+oaFsNk7tLco+C/obP9Ghhc0zGcK0ayTmE3LnAmAmH4=",  
    "ServiceSpecificCredentialId": "ACCAEXAMPLE123EXAMPLE",  
    "UserName": "sofia",
```



```
    "Status": "Active"
  }
}
```

For more information, see [Create Git credentials for HTTPS connections to CodeCommit](#) in the *AWS CodeCommit User Guide*.

- For API details, see [ResetServiceSpecificCredential](#) in *AWS CLI Command Reference*.

resync-mfa-device

The following code example shows how to use `resync-mfa-device`.

AWS CLI

To synchronize an MFA device

The following `resync-mfa-device` example synchronizes the MFA device that is associated with the IAM user Bob and whose ARN is `arn:aws:iam::123456789012:mfa/BobsMFADevice` with an authenticator program that provided the two authentication codes.

```
aws iam resync-mfa-device \
  --user-name Bob \
  --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice \
  --authentication-code1 123456 \
  --authentication-code2 987654
```

This command produces no output.

For more information, see [Using multi-factor authentication \(MFA\) in AWS](#) in the *AWS IAM User Guide*.

- For API details, see [ResyncMfaDevice](#) in *AWS CLI Command Reference*.

set-default-policy-version

The following code example shows how to use `set-default-policy-version`.

AWS CLI

To set the specified version of the specified policy as the policy's default version.

This example sets the v2 version of the policy whose ARN is `arn:aws:iam::123456789012:policy/MyPolicy` as the default active version.

```
aws iam set-default-policy-version \  
  --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \  
  --version-id v2
```

For more information, see [Policies and permissions in IAM](#) in the *AWS IAM User Guide*.

- For API details, see [SetDefaultPolicyVersion](#) in *AWS CLI Command Reference*.

set-security-token-service-preferences

The following code example shows how to use `set-security-token-service-preferences`.

AWS CLI

To set the global endpoint token version

The following `set-security-token-service-preferences` example configures Amazon STS to use version 2 tokens when you authenticate against the global endpoint.

```
aws iam set-security-token-service-preferences \  
  --global-endpoint-token-version v2Token
```

This command produces no output.

For more information, see [Managing AWS STS in an AWS Region](#) in the *AWS IAM User Guide*.

- For API details, see [SetSecurityTokenServicePreferences](#) in *AWS CLI Command Reference*.

simulate-custom-policy

The following code example shows how to use `simulate-custom-policy`.

AWS CLI

Example 1: To simulate the effects of all IAM policies associated with an IAM user or role

The following `simulate-custom-policy` shows how to provide both the policy and define variable values and simulate an API call to see if it is allowed or denied. The following example shows a policy that enables database access only after a specified date and time. The simulation

succeeds because the simulated actions and the specified `aws:CurrentTime` variable all match the requirements of the policy.

```
aws iam simulate-custom-policy \
  --policy-input-list '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"*","Condition":
{"DateGreaterThan":{"aws:CurrentTime":"2018-08-16T12:00:00Z"}}}' \
  --action-names dynamodb:CreateBackup \
  --context-entries
"ContextKeyName='aws:CurrentTime',ContextKeyValues='2019-04-25T11:00:00Z',ContextKeyType=da
```

Output:

```
{
  "EvaluationResults": [
    {
      "EvalActionName": "dynamodb:CreateBackup",
      "EvalResourceName": "*",
      "EvalDecision": "allowed",
      "MatchedStatements": [
        {
          "SourcePolicyId": "PolicyInputList.1",
          "StartPosition": {
            "Line": 1,
            "Column": 38
          },
          "EndPosition": {
            "Line": 1,
            "Column": 167
          }
        }
      ],
      "MissingContextValues": []
    }
  ]
}
```

Example 2: To simulate a command that is prohibited by the policy

The following `simulate-custom-policy` example shows the results of simulating a command that is prohibited by the policy. In this example, the provided date is before that required by the policy's condition.

```
aws iam simulate-custom-policy \
  --policy-input-list '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"*","Condition":
{"DateGreaterThan":{"aws:CurrentTime":"2018-08-16T12:00:00Z"}}}' \
  --action-names dynamodb>CreateBackup \
  --context-entries
"ContextKeyName='aws:CurrentTime',ContextKeyValues='2014-04-25T11:00:00Z',ContextKeyType=da
```

Output:

```
{
  "EvaluationResults": [
    {
      "EvalActionName": "dynamodb>CreateBackup",
      "EvalResourceName": "*",
      "EvalDecision": "implicitDeny",
      "MatchedStatements": [],
      "MissingContextValues": []
    }
  ]
}
```

For more information, see [Testing IAM policies with the IAM policy simulator](#) in the *AWS IAM User Guide*.

- For API details, see [SimulateCustomPolicy](#) in *AWS CLI Command Reference*.

simulate-principal-policy

The following code example shows how to use `simulate-principal-policy`.

AWS CLI

Example 1: To simulate the effects of an arbitrary IAM policy

The following `simulate-principal-policy` shows how to simulate a user calling an API action and determining whether the policies associated with that user allow or deny the action. In the following example, the user has a policy that allows only the `codecommit:ListRepositories` action.

```
aws iam simulate-principal-policy \
```

```
--policy-source-arn arn:aws:iam::123456789012:user/alejandro \  
--action-names codecommit:ListRepositories
```

Output:

```
{  
  "EvaluationResults": [  
    {  
      "EvalActionName": "codecommit:ListRepositories",  
      "EvalResourceName": "*",  
      "EvalDecision": "allowed",  
      "MatchedStatements": [  
        {  
          "SourcePolicyId": "Grant-Access-To-CodeCommit-ListRepo",  
          "StartPosition": {  
            "Line": 3,  
            "Column": 19  
          },  
          "EndPosition": {  
            "Line": 9,  
            "Column": 10  
          }  
        }  
      ],  
      "MissingContextValues": []  
    }  
  ]  
}
```

Example 2: To simulate the effects of a prohibited command

The following `simulate-custom-policy` example shows the results of simulating a command that is prohibited by one of the user's policies. In the following example, the user has a policy that permits access to a DynamoDB database only after a certain date and time. The simulation has the user attempting to access the database with an `aws:CurrentTime` value that is earlier than the policy's condition permits.

```
aws iam simulate-principal-policy \  
  --policy-source-arn arn:aws:iam::123456789012:user/alejandro \  
  --action-names dynamodb:CreateBackup \  
  --context-entries  
  "ContextKeyName='aws:CurrentTime',ContextKeyValues='2018-04-25T11:00:00Z',ContextKeyType=da
```

Output:

```
{
  "EvaluationResults": [
    {
      "EvalActionName": "dynamodb:CreateBackup",
      "EvalResourceName": "*",
      "EvalDecision": "implicitDeny",
      "MatchedStatements": [],
      "MissingContextValues": []
    }
  ]
}
```

For more information, see [Testing IAM policies with the IAM policy simulator](#) in the *AWS IAM User Guide*.

- For API details, see [SimulatePrincipalPolicy](#) in *AWS CLI Command Reference*.

tag-instance-profile

The following code example shows how to use tag-instance-profile.

AWS CLI**To add a tag to an instance profile**

The following tag-instance-profile command adds a tag with a Department name to the specified instance profile.

```
aws iam tag-instance-profile \
  --instance-profile-name deployment-role \
  --tags '[{"Key": "Department", "Value": "Accounting"}]'
```

This command produces no output.

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [TagInstanceProfile](#) in *AWS CLI Command Reference*.

tag-mfa-device

The following code example shows how to use tag-mfa-device.

AWS CLI

To add a tag to an MFA device

The following `tag-mfa-device` command adds a tag with a Department name to the specified MFA device.

```
aws iam tag-mfa-device \  
  --serial-number arn:aws:iam::123456789012:mfa/alice \  
  --tags '[{"Key": "Department", "Value": "Accounting"}]'
```

This command produces no output.

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [TagMfaDevice](#) in *AWS CLI Command Reference*.

tag-open-id-connect-provider

The following code example shows how to use `tag-open-id-connect-provider`.

AWS CLI

To add a tag to an OpenID Connect (OIDC)-compatible identity provider

The following `tag-open-id-connect-provider` command adds a tag with a Department name to the specified OIDC identity provider.

```
aws iam tag-open-id-connect-provider \  
  --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/  
server.example.com \  
  --tags '[{"Key": "Department", "Value": "Accounting"}]'
```

This command produces no output.

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [TagOpenIdConnectProvider](#) in *AWS CLI Command Reference*.

tag-policy

The following code example shows how to use `tag-policy`.

AWS CLI

To add a tag to a customer managed policy

The following `tag-policy` command adds a tag with a Department name to the specified customer managed policy.

```
aws iam tag-policy \  
  --policy-arn arn:aws:iam::123456789012:policy/billing-access \  
  --tags '[{"Key": "Department", "Value": "Accounting"}]'
```

This command produces no output.

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [TagPolicy](#) in *AWS CLI Command Reference*.

tag-role

The following code example shows how to use `tag-role`.

AWS CLI

To add a tag to a role

The following `tag-role` command adds a tag with a Department name to the specified role.

```
aws iam tag-role --role-name my-role \  
  --tags '{"Key": "Department", "Value": "Accounting"}'
```

This command produces no output.

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [TagRole](#) in *AWS CLI Command Reference*.

tag-saml-provider

The following code example shows how to use `tag-saml-provider`.

AWS CLI

To add a tag to a SAML provider

The following `tag-saml-provider` command adds a tag with a Department name to the specified SAML provider.

```
aws iam tag-saml-provider \  
  --saml-provider-arn arn:aws:iam::123456789012:saml-provider/ADFS \  
  --tags '[{"Key": "Department", "Value": "Accounting"}]'
```

This command produces no output.

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [TagSamlProvider](#) in *AWS CLI Command Reference*.

tag-server-certificate

The following code example shows how to use `tag-server-certificate`.

AWS CLI

To add a tag to a server certificate

The following `tag-saml-provider` command adds a tag with a Department name to the specified sever certificate.

```
aws iam tag-server-certificate \  
  --server-certificate-name ExampleCertificate \  
  --tags '[{"Key": "Department", "Value": "Accounting"}]'
```

This command produces no output.

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [TagServerCertificate](#) in *AWS CLI Command Reference*.

tag-user

The following code example shows how to use `tag-user`.

AWS CLI

To add a tag to a user

The following `tag-user` command adds a tag with the associated Department to the specified user.

```
aws iam tag-user \  
  --user-name alice \  
  --tags '{"Key": "Department", "Value": "Accounting"}'
```

This command produces no output.

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [TagUser](#) in *AWS CLI Command Reference*.

untag-instance-profile

The following code example shows how to use `untag-instance-profile`.

AWS CLI

To remove a tag from an instance profile

The following `untag-instance-profile` command removes any tag with the key name 'Department' from the specified instance profile.

```
aws iam untag-instance-profile \  
  --instance-profile-name deployment-role \  
  --tag-keys Department
```

This command produces no output.

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [UntagInstanceProfile](#) in *AWS CLI Command Reference*.

untag-mfa-device

The following code example shows how to use `untag-mfa-device`.

AWS CLI

To remove a tag from an MFA device

The following `untag-mfa-device` command removes any tag with the key name 'Department' from the specified MFA device.

```
aws iam untag-mfa-device \  
  --serial-number arn:aws:iam::123456789012:mfa/alice \  
  --tag-keys Department
```

This command produces no output.

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [UntagMfaDevice](#) in *AWS CLI Command Reference*.

untag-open-id-connect-provider

The following code example shows how to use `untag-open-id-connect-provider`.

AWS CLI

To remove a tag from an OIDC identity provider

The following `untag-open-id-connect-provider` command removes any tag with the key name 'Department' from the specified OIDC identity provider.

```
aws iam untag-open-id-connect-provider \  
  --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/  
server.example.com \  
  --tag-keys Department
```

This command produces no output.

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [UntagOpenIdConnectProvider](#) in *AWS CLI Command Reference*.

untag-policy

The following code example shows how to use `untag-policy`.

AWS CLI

To remove a tag from a customer managed policy

The following `untag-policy` command removes any tag with the key name 'Department' from the specified customer managed policy.

```
aws iam untag-policy \  
  --policy-arn arn:aws:iam::452925170507:policy/billing-access \  
  --tag-keys Department
```

This command produces no output.

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [UntagPolicy](#) in *AWS CLI Command Reference*.

untag-role

The following code example shows how to use `untag-role`.

AWS CLI

To remove a tag from a role

The following `untag-role` command removes any tag with the key name 'Department' from the specified role.

```
aws iam untag-role \  
  --role-name my-role \  
  --tag-keys Department
```

This command produces no output.

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [UntagRole](#) in *AWS CLI Command Reference*.

untag-saml-provider

The following code example shows how to use `untag-saml-provider`.

AWS CLI

To remove a tag from a SAML provider

The following `untag-saml-provider` command removes any tag with the key name 'Department' from the specified instance profile.

```
aws iam untag-saml-provider \  
  --saml-provider-arn arn:aws:iam::123456789012:saml-provider/ADFS \  
  --tag-keys Department
```

This command produces no output.

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [UntagSamlProvider](#) in *AWS CLI Command Reference*.

untag-server-certificate

The following code example shows how to use `untag-server-certificate`.

AWS CLI

To remove a tag from a server certificate

The following `untag-server-certificate` command removes any tag with the key name 'Department' from the specified server certificate.

```
aws iam untag-server-certificate \  
  --server-certificate-name ExampleCertificate \  
  --tag-keys Department
```

This command produces no output.

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [UntagServerCertificate](#) in *AWS CLI Command Reference*.

untag-user

The following code example shows how to use `untag-user`.

AWS CLI

To remove a tag from a user

The following `untag-user` command removes any tag with the key name 'Department' from the specified user.

```
aws iam untag-user \  
  --user-name alice \  
  --tag-keys Department
```

This command produces no output.

For more information, see [Tagging IAM resources](#) in the *AWS IAM User Guide*.

- For API details, see [UntagUser](#) in *AWS CLI Command Reference*.

update-access-key

The following code example shows how to use `update-access-key`.

AWS CLI

To activate or deactivate an access key for an IAM user

The following `update-access-key` command deactivates the specified access key (access key ID and secret access key) for the IAM user named Bob.

```
aws iam update-access-key \  
  --access-key-id AKIAIOSFODNN7EXAMPLE \  
  --status Inactive \  
  --user-name Bob
```

This command produces no output.

Deactivating the key means that it cannot be used for programmatic access to AWS. However, the key is still available and can be reactivated.

For more information, see [Managing access keys for IAM users](#) in the *AWS IAM User Guide*.

- For API details, see [UpdateAccessKey](#) in *AWS CLI Command Reference*.

update-account-password-policy

The following code example shows how to use `update-account-password-policy`.

AWS CLI

To set or change the current account password policy

The following `update-account-password-policy` command sets the password policy to require a minimum length of eight characters and to require one or more numbers in the password.

```
aws iam update-account-password-policy \  
  --minimum-password-length 8 \  
  --require-numbers
```

This command produces no output.

Changes to an account's password policy affect any new passwords that are created for IAM users in the account. Password policy changes do not affect existing passwords.

For more information, see [Setting an account password policy for IAM users](#) in the *AWS IAM User Guide*.

- For API details, see [UpdateAccountPasswordPolicy](#) in *AWS CLI Command Reference*.

update-assume-role-policy

The following code example shows how to use `update-assume-role-policy`.

AWS CLI

To update the trust policy for an IAM role

The following `update-assume-role-policy` command updates the trust policy for the role named `Test-Role`.

```
aws iam update-assume-role-policy \  
  --role-name Test-Role \  
  --policy-document file://Test-Role-Trust-Policy.json
```

This command produces no output.

The trust policy is defined as a JSON document in the `Test-Role-Trust-Policy.json` file. (The file name and extension do not have significance.) The trust policy must specify a principal.

To update the permissions policy for a role, use the `put-role-policy` command.

For more information, see [Creating IAM roles](#) in the *AWS IAM User Guide*.

- For API details, see [UpdateAssumeRolePolicy](#) in *AWS CLI Command Reference*.

update-group

The following code example shows how to use `update-group`.

AWS CLI

To rename an IAM group

The following `update-group` command changes the name of the IAM group `Test` to `Test-1`.

```
aws iam update-group \  
  --group-name Test \  
  --new-group-name Test-1
```

This command produces no output.

For more information, see [Renaming an IAM user group](#) in the *AWS IAM User Guide*.

- For API details, see [UpdateGroup](#) in *AWS CLI Command Reference*.

update-login-profile

The following code example shows how to use `update-login-profile`.

AWS CLI

To update the password for an IAM user

The following `update-login-profile` command creates a new password for the IAM user named `Bob`.

```
aws iam update-login-profile \  
  --user-name Bob \  
  --password <password>
```


This command produces no output.

To set a password policy for the account, use the `update-account-password-policy` command. If the new password violates the account password policy, the command returns a `PasswordPolicyViolation` error.

If the account password policy allows them to, IAM users can change their own passwords using the `change-password` command.

Store the password in a secure place. If the password is lost, it cannot be recovered, and you must create a new one using the `create-login-profile` command.

For more information, see [Managing passwords for IAM users](#) in the *AWS IAM User Guide*.

- For API details, see [UpdateLoginProfile](#) in *AWS CLI Command Reference*.

update-open-id-connect-provider-thumbprint

The following code example shows how to use `update-open-id-connect-provider-thumbprint`.

AWS CLI

To replace the existing list of server certificate thumbprints with a new list

This example updates the certificate thumbprint list for the OIDC provider whose ARN is `arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com` to use a new thumbprint.

```
aws iam update-open-id-connect-provider-thumbprint \  
  --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/  
example.oidcprovider.com \  
  --thumbprint-list 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

This command produces no output.

For more information, see [Creating OpenID Connect \(OIDC\) identity providers](#) in the *AWS IAM User Guide*.

- For API details, see [UpdateOpenIdConnectProviderThumbprint](#) in *AWS CLI Command Reference*.

update-role-description

The following code example shows how to use `update-role-description`.

AWS CLI

To change an IAM role's description

The following `update-role` command changes the description of the IAM role `production-role` to `Main production role`.

```
aws iam update-role-description \  
  --role-name production-role \  
  --description 'Main production role'
```

Output:

```
{  
  "Role": {  
    "Path": "/",  
    "RoleName": "production-role",  
    "RoleId": "AROA1234567890EXAMPLE",  
    "Arn": "arn:aws:iam::123456789012:role/production-role",  
    "CreateDate": "2017-12-06T17:16:37+00:00",  
    "AssumeRolePolicyDocument": {  
      "Version": "2012-10-17",  
      "Statement": [  
        {  
          "Effect": "Allow",  
          "Principal": {  
            "AWS": "arn:aws:iam::123456789012:root"  
          },  
          "Action": "sts:AssumeRole",  
          "Condition": {}  
        }  
      ]  
    },  
    "Description": "Main production role"  
  }  
}
```

For more information, see [Modifying a role](#) in the *AWS IAM User Guide*.

- For API details, see [UpdateRoleDescription](#) in *AWS CLI Command Reference*.

update-role

The following code example shows how to use `update-role`.

AWS CLI

To change an IAM role's description or session duration

The following `update-role` command changes the description of the IAM role `production-role` to `Main production role` and sets the maximum session duration to 12 hours.

```
aws iam update-role \  
  --role-name production-role \  
  --description 'Main production role' \  
  --max-session-duration 43200
```

This command produces no output.

For more information, see [Modifying a role](#) in the *AWS IAM User Guide*.

- For API details, see [UpdateRole](#) in *AWS CLI Command Reference*.

update-saml-provider

The following code example shows how to use `update-saml-provider`.

AWS CLI

To update the metadata document for an existing SAML provider

This example updates the SAML provider in IAM whose ARN is `arn:aws:iam::123456789012:saml-provider/SAMLADFS` with a new SAML metadata document from the file `SAMLMetaData.xml`.

```
aws iam update-saml-provider \  
  --saml-metadata-document file://SAMLMetaData.xml \  
  --saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFS
```

Output:

```
{
  "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/SAMLADFS"
}
```

For more information, see [Creating IAM SAML identity providers](#) in the *AWS IAM User Guide*.

- For API details, see [UpdateSamlProvider](#) in *AWS CLI Command Reference*.

update-server-certificate

The following code example shows how to use `update-server-certificate`.

AWS CLI

To change the path or name of a server certificate in your AWS account

The following `update-server-certificate` command changes the name of the certificate from `myServerCertificate` to `myUpdatedServerCertificate`. It also changes the path to `/cloudfront/` so that it can be accessed by the Amazon CloudFront service. This command produces no output. You can see the results of the update by running the `list-server-certificates` command.

```
aws-iam update-server-certificate \
  --server-certificate-name myServerCertificate \
  --new-server-certificate-name myUpdatedServerCertificate \
  --new-path /cloudfront/
```

This command produces no output.

For more information, see [Managing server certificates in IAM](#) in the *AWS IAM User Guide*.

- For API details, see [UpdateServerCertificate](#) in *AWS CLI Command Reference*.

update-service-specific-credential

The following code example shows how to use `update-service-specific-credential`.

AWS CLI

Example 1: To update the status of the requesting user's service-specific credential

The following `update-service-specific-credential` example changes the status for the specified credential for the user making the request to Inactive.

```
aws iam update-service-specific-credential \  
  --service-specific-credential-id ACCAEXAMPLE123EXAMPLE \  
  --status Inactive
```

This command produces no output.

Example 2: To update the status of a specified user's service-specific credential

The following `update-service-specific-credential` example changes the status for the credential of the specified user to Inactive.

```
aws iam update-service-specific-credential \  
  --user-name sofia \  
  --service-specific-credential-id ACCAEXAMPLE123EXAMPLE \  
  --status Inactive
```

This command produces no output.

For more information, see [Create Git Credentials for HTTPS Connections to CodeCommit](#) in the *AWS CodeCommit User Guide*

- For API details, see [UpdateServiceSpecificCredential](#) in *AWS CLI Command Reference*.

update-signing-certificate

The following code example shows how to use `update-signing-certificate`.

AWS CLI

To activate or deactivate a signing certificate for an IAM user

The following `update-signing-certificate` command deactivates the specified signing certificate for the IAM user named Bob.

```
aws iam update-signing-certificate \  
  --certificate-id TA7SMP42TDN5Z260BPJE7EXAMPLE \  
  --status Inactive \  
  --user-name Bob
```

To get the ID for a signing certificate, use the `list-signing-certificates` command.

For more information, see [Manage signing certificates](#) in the *Amazon EC2 User Guide*.

- For API details, see [UpdateSigningCertificate](#) in *AWS CLI Command Reference*.

update-ssh-public-key

The following code example shows how to use `update-ssh-public-key`.

AWS CLI

To change the status of an SSH public key

The following `update-ssh-public-key` command changes the status of the specified public key to `Inactive`.

```
aws iam update-ssh-public-key \  
  --user-name sofia \  
  --ssh-public-key-id APKA1234567890EXAMPLE \  
  --status Inactive
```

This command produces no output.

For more information, see [Use SSH keys and SSH with CodeCommit](#) in the *AWS IAM User Guide*.

- For API details, see [UpdateSshPublicKey](#) in *AWS CLI Command Reference*.

update-user

The following code example shows how to use `update-user`.

AWS CLI

To change an IAM user's name

The following `update-user` command changes the name of the IAM user Bob to Robert.

```
aws iam update-user \  
  --user-name Bob \  
  --new-user-name Robert
```

This command produces no output.

For more information, see [Renaming an IAM user group](#) in the *AWS IAM User Guide*.

- For API details, see [UpdateUser](#) in *AWS CLI Command Reference*.

upload-server-certificate

The following code example shows how to use `upload-server-certificate`.

AWS CLI

To upload a server certificate to your AWS account

The following `upload-server-certificate` command uploads a server certificate to your AWS account. In this example, the certificate is in the file `public_key_cert_file.pem`, the associated private key is in the file `my_private_key.pem`, and the the certificate chain provided by the certificate authority (CA) is in the `my_certificate_chain_file.pem` file. When the file has finished uploading, it is available under the name `myServerCertificate`. Parameters that begin with `file://` tells the command to read the contents of the file and use that as the parameter value instead of the file name itself.

```
aws iam upload-server-certificate \  
  --server-certificate-name myServerCertificate \  
  --certificate-body file://public_key_cert_file.pem \  
  --private-key file://my_private_key.pem \  
  --certificate-chain file://my_certificate_chain_file.pem
```

Output:

```
{  
  "ServerCertificateMetadata": {  
    "Path": "/",  
    "ServerCertificateName": "myServerCertificate",  
    "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",  
    "Arn": "arn:aws:iam::1234567989012:server-certificate/myServerCertificate",  
    "UploadDate": "2019-04-22T21:13:44+00:00",  
    "Expiration": "2019-10-15T22:23:16+00:00"  
  }  
}
```

For more information, see [Creating, Uploading, and Deleting Server Certificates](#) in the *Using IAM* guide.

- For API details, see [UploadServerCertificate](#) in *AWS CLI Command Reference*.

upload-signing-certificate

The following code example shows how to use `upload-signing-certificate`.

AWS CLI

To upload a signing certificate for an IAM user

The following `upload-signing-certificate` command uploads a signing certificate for the IAM user named Bob.

```
aws iam upload-signing-certificate \  
  --user-name Bob \  
  --certificate-body file://certificate.pem
```

Output:

```
{  
  "Certificate": {  
    "UserName": "Bob",  
    "Status": "Active",  
    "CertificateBody": "-----BEGIN CERTIFICATE-----<certificate-body>-----END  
CERTIFICATE-----",  
    "CertificateId": "TA7SMP42TDN5Z260BPJE7EXAMPLE",  
    "UploadDate": "2013-06-06T21:40:08.121Z"  
  }  
}
```

The certificate is in a file named *certificate.pem* in PEM format.

For more information, see [Creating and Uploading a User Signing Certificate](#) in the *Using IAM* guide.

- For API details, see [UploadSigningCertificate](#) in *AWS CLI Command Reference*.

upload-ssh-public-key

The following code example shows how to use `upload-ssh-public-key`.

AWS CLI

To upload an SSH public key and associate it with a user

The following `upload-ssh-public-key` command uploads the public key found in the file `sshkey.pub` and attaches it to the user `sofia`.

```
aws iam upload-ssh-public-key \  
  --user-name sofia \  
  --ssh-public-key-body file://sshkey.pub
```

Output:

```
{  
  "SSHPublicKey": {  
    "UserName": "sofia",  
    "SSHPublicKeyId": "APKA1234567890EXAMPLE",  
    "Fingerprint": "12:34:56:78:90:ab:cd:ef:12:34:56:78:90:ab:cd:ef",  
    "SSHPublicKeyBody": "ssh-rsa <<long string generated by ssh-keygen  
command>>",  
    "Status": "Active",  
    "UploadDate": "2019-04-18T17:04:49+00:00"  
  }  
}
```

For more information about how to generate keys in a format suitable for this command, see [SSH and Linux, macOS, or Unix: Set up the public and private keys for Git and CodeCommit](#) or [SSH and Windows: Set up the public and private keys for Git and CodeCommit](#) in the *AWS CodeCommit User Guide*.

- For API details, see [UploadSshPublicKey](#) in *AWS CLI Command Reference*.

IAM Access Analyzer examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with IAM Access Analyzer.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

apply-archive-rule

The following code example shows how to use `apply-archive-rule`.

AWS CLI

To apply an archive rule to existing findings that meet the archive rule criteria

The following `apply-archive-rule` example applies an archive rule to existing findings that meet the archive rule criteria.

```
aws accessanalyzer apply-archive-rule \  
  --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/  
UnusedAccess-ConsoleAnalyzer-organization \  
  --rule-name MyArchiveRule
```

This command produces no output.

For more information, see [Archive rules](#) in the *AWS IAM User Guide*.

- For API details, see [ApplyArchiveRule](#) in *AWS CLI Command Reference*.

cancel-policy-generation

The following code example shows how to use `cancel-policy-generation`.

AWS CLI

To cancel the requested policy generation

The following `cancel-policy-generation` example cancels the requested policy generation job id.

```
aws accessanalyzer cancel-policy-generation \  
  --job-id 923a56b0-ebb8-4e80-8a3c-a11ccfbcd6f2
```

This command produces no output.

For more information, see [IAM Access Analyzer policy generation](#) in the *AWS IAM User Guide*.

- For API details, see [CancelPolicyGeneration](#) in *AWS CLI Command Reference*.

check-access-not-granted

The following code example shows how to use `check-access-not-granted`.

AWS CLI

To check whether the specified access isn't allowed by a policy

The following `check-access-not-granted` example checks whether the specified access isn't allowed by a policy.

```
aws accessanalyzer check-access-not-granted \  
  --policy-document file://myfile.json \  
  --access actions="s3:DeleteBucket","s3:GetBucketLocation" \  
  --policy-type IDENTITY_POLICY
```

Contents of `myfile.json`:

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "s3:GetObject",  
        "s3:ListBucket"  
      ],  
      "Resource": [  
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
```

```

        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
    ]
}

```

Output:

```

{
  "result": "PASS",
  "message": "The policy document does not grant access to perform the listed
actions."
}

```

For more information, see [Previewing access with IAM Access Analyzer APIs](#) in the *AWS IAM User Guide*.

- For API details, see [CheckAccessNotGranted](#) in *AWS CLI Command Reference*.

check-no-new-access

The following code example shows how to use `check-no-new-access`.

AWS CLI**To check whether new access is allowed for an updated policy when compared to the existing policy**

The following `check-no-new-access` example checks whether new access is allowed for an updated policy when compared to the existing policy.

```

aws accessanalyzer check-no-new-access \
  --existing-policy-document file://existing-policy.json \
  --new-policy-document file://new-policy.json \
  --policy-type IDENTITY_POLICY

```

Contents of `existing-policy.json`:

```

{
  "Version": "2012-10-17",
  "Statement": [

```

```
{
  "Effect": "Allow",
  "Action": [
    "s3:GetObject",
    "s3:ListBucket"
  ],
  "Resource": [
    "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
    "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
  ]
}
```

Contents of new-policy.json:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectAcl",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      ]
    }
  ]
}
```

Output:

```
{
  "result": "FAIL",
  "message": "The modified permissions grant new access compared to your existing policy.",
  "reasons": [
    {
      "description": "New access in the statement with index: 0.",
```

```

        "statementIndex": 0
      }
    ]
  }

```

For more information, see [Previewing access with IAM Access Analyzer APIs](#) in the *AWS IAM User Guide*.

- For API details, see [CheckNoNewAccess](#) in *AWS CLI Command Reference*.

create-access-preview

The following code example shows how to use create-access-preview.

AWS CLI

To create an access preview that allows you to preview IAM Access Analyzer findings for your resource before deploying resource permissions

The following create-access-preview example creates an access preview that allows you to preview IAM Access Analyzer findings for your resource before deploying resource permissions in your AWS account.

```

aws accessanalyzer create-access-preview \
  --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
  ConsoleAnalyzer-account \
  --configurations file://myfile.json

```

Contents of myfile.json:

```

{
  "arn:aws:s3:::DOC-EXAMPLE-BUCKET": {
    "s3Bucket": {
      "bucketPolicy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",\"Principal\":{\"AWS\":[\"arn:aws:iam::111122223333:root\"]},\"Action\":[\"s3:PutObject\",\"s3:PutObjectAcl\"],\"Resource\":[\"arn:aws:s3:::DOC-EXAMPLE-BUCKET/*\"]}]}",
      "bucketPublicAccessBlock": {
        "ignorePublicAcls": true,
        "restrictPublicBuckets": true
      },
      "bucketAclGrants": [

```

```
{
  "grantee": {
    "id":
"79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be"
  },
  "permission": "READ"
}
]
```

Output:

```
{
  "id": "3c65eb13-6ef9-4629-8919-a32043619e6b"
}
```

For more information, see [Previewing access with IAM Access Analyzer APIs](#) in the *AWS IAM User Guide*.

- For API details, see [CreateAccessPreview](#) in *AWS CLI Command Reference*.

create-analyzer

The following code example shows how to use `create-analyzer`.

AWS CLI**To create an analyzer**

The following `create-analyzer` example creates an analyzer in your AWS account.

```
aws accessanalyzer create-analyzer \
  --analyzer-name example \
  --type ACCOUNT
```

Output:

```
{
  "arn": "arn:aws:access-analyzer:us-east-2:111122223333:analyzer/example"
```

```
}
```

For more information, see [Getting started with AWS Identity and Access Management Access Analyzer findings](#) in the *AWS IAM User Guide*.

- For API details, see [CreateAnalyzer](#) in *AWS CLI Command Reference*.

create-archive-rule

The following code example shows how to use `create-archive-rule`.

AWS CLI

To create an archive rule for the specified analyzer

The following `create-archive-rule` example creates an archive rule for the specified analyzer in your AWS account.

```
aws accessanalyzer create-archive-rule \  
  --analyzer-name UnusedAccess-ConsoleAnalyzer-organization \  
  --rule-name MyRule \  
  --filter '{"resource": {"contains": ["Cognito"]}, "resourceType": {"eq":  
  ["AWS::IAM::Role"]}]'
```

This command produces no output.

For more information, see [Archive rules](#) in the *AWS IAM User Guide*.

- For API details, see [CreateArchiveRule](#) in *AWS CLI Command Reference*.

delete-analyzer

The following code example shows how to use `delete-analyzer`.

AWS CLI

To delete the specified analyzer

The following `delete-analyzer` example deletes the specified analyzer in your AWS account.

```
aws accessanalyzer delete-analyzer \  
  --analyzer-name example
```


This command produces no output.

For more information, see [Archive rules](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteAnalyzer](#) in *AWS CLI Command Reference*.

delete-archive-rule

The following code example shows how to use `delete-archive-rule`.

AWS CLI

To delete the specified archive rule

The following `delete-archive-rule` example deletes the specified archive rule in your AWS account.

```
aws accessanalyzer delete-archive-rule \  
  --analyzer-name UnusedAccess-ConsoleAnalyzer-organization \  
  --rule-name MyRule
```

This command produces no output.

For more information, see [Archive rules](#) in the *AWS IAM User Guide*.

- For API details, see [DeleteArchiveRule](#) in *AWS CLI Command Reference*.

get-access-preview

The following code example shows how to use `get-access-preview`.

AWS CLI

To retrieves information about an access preview for the specified analyzer

The following `get-access-preview` example retrieves information about an access preview for the specified analyzer in your AWS account.

```
aws accessanalyzer get-access-preview \  
  --access-preview-id 3c65eb13-6ef9-4629-8919-a32043619e6b \  
  --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/  
ConsoleAnalyzer-account
```

Output:

```
{
  "accessPreview": {
    "id": "3c65eb13-6ef9-4629-8919-a32043619e6b",
    "analyzerArn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account",
    "configurations": {
      "arn:aws:s3:::DOC-EXAMPLE-BUCKET": {
        "s3Bucket": {
          "bucketPolicy": "{\"Version\":\"2012-10-17\",\"Statement\":
[{\n\"Effect\":\n\"Allow\", \"Principal\":{\n\"AWS\":[\n\"arn:aws:iam::111122223333:root\"
]}\n\"Action\":[\n\"s3:PutObject\", \"s3:PutObjectAcl\"], \"Resource\":\n\"arn:aws:s3:::DOC-
EXAMPLE-BUCKET/*\"}]}",
          "bucketAclGrants": [
            {
              "permission": "READ",
              "grantee": {
                "id":
"79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be"
              }
            }
          ],
          "bucketPublicAccessBlock": {
            "ignorePublicAcls": true,
            "restrictPublicBuckets": true
          }
        }
      }
    },
    "createdAt": "2024-02-17T00:18:44+00:00",
    "status": "COMPLETED"
  }
}
```

For more information, see [Previewing access with IAM Access Analyzer APIs](#) in the *AWS IAM User Guide*.

- For API details, see [GetAccessPreview](#) in *AWS CLI Command Reference*.

get-analyzed-resource

The following code example shows how to use `get-analyzed-resource`.

AWS CLI

To retrieve information about a resource that was analyzed

The following `get-analyzed-resource` example retrieves information about a resource that was analyzed in your AWS account.

```
aws accessanalyzer get-analyzed-resource \
  --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
  ConsoleAnalyzer-account \
  --resource-arn arn:aws:s3:::DOC-EXAMPLE-BUCKET
```

Output:

```
{
  "resource": {
    "analyzedAt": "2024-02-15T18:01:53.002000+00:00",
    "isPublic": false,
    "resourceArn": "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
    "resourceOwnerAccount": "111122223333",
    "resourceType": "AWS::S3::Bucket"
  }
}
```

For more information, see [Using AWS Identity and Access Management Access Analyzer](#) in the *AWS IAM User Guide*.

- For API details, see [GetAnalyzedResource](#) in *AWS CLI Command Reference*.

get-analyzer

The following code example shows how to use `get-analyzer`.

AWS CLI

To retrieve information about the specified analyzer

The following `get-analyzer` example retrieves information about the specified analyzer in your AWS account.

```
aws accessanalyzer get-analyzer \
  --analyzer-name ConsoleAnalyzer-account
```

Output:

```
{
  "analyzer": {
    "arn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account",
    "createdAt": "2019-12-03T07:28:17+00:00",
    "lastResourceAnalyzed": "arn:aws:sns:us-west-2:111122223333:config-topic",
    "lastResourceAnalyzedAt": "2024-02-15T18:01:53.003000+00:00",
    "name": "ConsoleAnalyzer-account",
    "status": "ACTIVE",
    "tags": {
      "auto-delete": "no"
    },
    "type": "ACCOUNT"
  }
}
```

For more information, see [Using AWS Identity and Access Management Access Analyzer](#) in the *AWS IAM User Guide*.

- For API details, see [GetAnalyzer](#) in *AWS CLI Command Reference*.

get-archive-rule

The following code example shows how to use `get-archive-rule`.

AWS CLI**To retrieve information about an archive rule**

The following `get-archive-rule` example retrieves information about an archive rule in your AWS account.

```
aws accessanalyzer get-archive-rule \
  --analyzer-name UnusedAccess-ConsoleAnalyzer-organization \
  --rule-name MyArchiveRule
```

Output:

```
{
```

```
"archiveRule": {
  "createdAt": "2024-02-15T00:49:27+00:00",
  "filter": {
    "resource": {
      "contains": [
        "Cognito"
      ]
    },
    "resourceType": {
      "eq": [
        "AWS::IAM::Role"
      ]
    }
  },
  "ruleName": "MyArchiveRule",
  "updatedAt": "2024-02-15T00:49:27+00:00"
}
}
```

For more information, see [Archive rules](#) in the *AWS IAM User Guide*.

- For API details, see [GetArchiveRule](#) in *AWS CLI Command Reference*.

get-finding-v2

The following code example shows how to use `get-finding-v2`.

AWS CLI

To retrieve information about the specified finding

The following `get-finding-v2` example retrieves information about the specified finding in your AWS account.

```
aws accessanalyzer get-finding-v2 \
  --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
  ConsoleAnalyzer-organization \
  --id 0910eedb-381e-4e95-adda-0d25c19e6e90
```

Output:

```
{
```

```

    "findingDetails": [
      {
        "externalAccessDetails": {
          "action": [
            "sts:AssumeRoleWithWebIdentity"
          ],
          "condition": {
            "cognito-identity.amazonaws.com:aud": "us-
west-2:EXAMPLE0-0000-0000-0000-000000000000"
          },
          "isPublic": false,
          "principal": {
            "Federated": "cognito-identity.amazonaws.com"
          }
        }
      }
    ],
    "resource": "arn:aws:iam::111122223333:role/Cognito_testpoolAuth_Role",
    "status": "ACTIVE",
    "error": null,
    "createdAt": "2021-02-26T21:17:50.905000+00:00",
    "resourceType": "AWS::IAM::Role",
    "findingType": "ExternalAccess",
    "resourceOwnerAccount": "111122223333",
    "analyzedAt": "2024-02-16T18:17:47.888000+00:00",
    "id": "0910eedb-381e-4e95-adda-0d25c19e6e90",
    "updatedAt": "2021-02-26T21:17:50.905000+00:00"
  }
}

```

For more information, see [Reviewing findings](#) in the *AWS IAM User Guide*.

- For API details, see [GetFindingV2](#) in *AWS CLI Command Reference*.

get-finding

The following code example shows how to use `get-finding`.

AWS CLI

To retrieve information about the specified finding

The following `get-finding` example retrieves information about the specified finding in your AWS account.

```
aws accessanalyzer get-finding \  
  --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/  
ConsoleAnalyzer-organization \  
  --id 0910eedb-381e-4e95-adda-0d25c19e6e90
```

Output:

```
{  
  "finding": {  
    "id": "0910eedb-381e-4e95-adda-0d25c19e6e90",  
    "principal": {  
      "Federated": "cognito-identity.amazonaws.com"  
    },  
    "action": [  
      "sts:AssumeRoleWithWebIdentity"  
    ],  
    "resource": "arn:aws:iam::111122223333:role/Cognito_testpoolAuth_Role",  
    "isPublic": false,  
    "resourceType": "AWS::IAM::Role",  
    "condition": {  
      "cognito-identity.amazonaws.com:aud": "us-  
west-2:EXAMPLE0-0000-0000-0000-000000000000"  
    },  
    "createdAt": "2021-02-26T21:17:50.905000+00:00",  
    "analyzedAt": "2024-02-16T18:17:47.888000+00:00",  
    "updatedAt": "2021-02-26T21:17:50.905000+00:00",  
    "status": "ACTIVE",  
    "resourceOwnerAccount": "111122223333"  
  }  
}
```

For more information, see [Reviewing findings](#) in the *AWS IAM User Guide*.

- For API details, see [GetFinding](#) in *AWS CLI Command Reference*.

get-generated-policy

The following code example shows how to use `get-generated-policy`.

AWS CLI

To retrieve the policy that was generated using the `StartPolicyGeneration` API`

The following `get-generated-policy` example retrieves the policy that was generated using the `StartPolicyGeneration` API in your AWS account.

```
aws accessanalyzer get-generated-policy \
  --job-id c557dc4a-0338-4489-95dd-739014860ff9
```

Output:

```
{
  "generatedPolicyResult": {
    "generatedPolicies": [
      {
        "policy": "{\"Version\":\"2012-10-17\",\"Statement\":
[{\n\"Sid\":\n\"SupportedServiceSid0\", \"Effect\": \"Allow\", \"Action\":
[\"access-analyzer:GetAnalyzer\", \"access-analyzer:ListAnalyzers\",
\n\"access-analyzer:ListArchiveRules\", \"access-analyzer:ListFindings
\n\", \"cloudtrail:DescribeTrails\", \"cloudtrail:GetEventDataStore\",
\n\"cloudtrail:GetEventSelectors\", \"cloudtrail:GetInsightSelectors
\n\", \"cloudtrail:GetTrailStatus\", \"cloudtrail:ListChannels\",
\n\"cloudtrail:ListEventDataStores\", \"cloudtrail:ListQueries\", \"cloudtrail:ListTags
\n\", \"cloudtrail:LookupEvents\", \"ec2:DescribeRegions\", \"iam:GetAccountSummary
\n\", \"iam:GetOpenIDConnectProvider\", \"iam:GetRole\", \"iam:ListAccessKeys\",
\n\"iam:ListAccountAliases\", \"iam:ListOpenIDConnectProviders\", \"iam:ListRoles
\n\", \"iam:ListSAMLProviders\", \"kms:ListAliases\", \"s3:GetBucketLocation\",
\n\"s3:ListAllMyBuckets\"]\", \"Resource\": \"*\"]}]"
      }
    ],
    "properties": {
      "cloudTrailProperties": {
        "endTime": "2024-02-14T22:44:40+00:00",
        "startTime": "2024-02-13T00:30:00+00:00",
        "trailProperties": [
          {
            "allRegions": true,
            "cloudTrailArn": "arn:aws:cloudtrail:us-
west-2:111122223333:trail/my-trail",
            "regions": []
          }
        ]
      },
      "isComplete": false,
      "principalArn": "arn:aws:iam::111122223333:role/Admin"
    }
  }
}
```



```
  },
  "jobDetails": {
    "completedOn": "2024-02-14T22:47:01+00:00",
    "jobId": "c557dc4a-0338-4489-95dd-739014860ff9",
    "startedOn": "2024-02-14T22:44:41+00:00",
    "status": "SUCCEEDED"
  }
}
```

For more information, see [IAM Access Analyzer policy generation](#) in the *AWS IAM User Guide*.

- For API details, see [GetGeneratedPolicy](#) in *AWS CLI Command Reference*.

list-access-preview-findings

The following code example shows how to use `list-access-preview-findings`.

AWS CLI

To retrieve a list of access preview findings generated by the specified access preview

The following `list-access-preview-findings` example retrieves a list of access preview findings generated by the specified access preview in your AWS account.

```
aws accessanalyzer list-access-preview-findings \
  --access-preview-id 3c65eb13-6ef9-4629-8919-a32043619e6b \
  --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account
```

Output:

```
{
  "findings": [
    {
      "id": "e22fc158-1c87-4c32-9464-e7f405ce8d74",
      "principal": {
        "AWS": "111122223333"
      },
      "action": [
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
    }
  ],
}
```

```

    "condition": {},
    "resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
    "isPublic": false,
    "resourceType": "AWS::S3::Bucket",
    "createdAt": "2024-02-17T00:18:46+00:00",
    "changeType": "NEW",
    "status": "ACTIVE",
    "resourceOwnerAccount": "111122223333",
    "sources": [
      {
        "type": "POLICY"
      }
    ]
  }
]
}

```

For more information, see [Previewing access with IAM Access Analyzer APIs](#) in the *AWS IAM User Guide*.

- For API details, see [ListAccessPreviewFindings](#) in *AWS CLI Command Reference*.

list-access-previews

The following code example shows how to use `list-access-previews`.

AWS CLI

To retrieve a list of access previews for the specified analyzer

The following `list-access-previews` example retrieves a list of access previews for the specified analyzer in your AWS account.

```

aws accessanalyzer list-access-previews \
  --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
  ConsoleAnalyzer-account

```

Output:

```

{
  "accessPreviews": [
    {

```

```

        "id": "3c65eb13-6ef9-4629-8919-a32043619e6b",
        "analyzerArn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account",
        "createdAt": "2024-02-17T00:18:44+00:00",
        "status": "COMPLETED"
    }
]
}

```

For more information, see [Previewing access with IAM Access Analyzer APIs](#) in the *AWS IAM User Guide*.

- For API details, see [ListAccessPreviews](#) in *AWS CLI Command Reference*.

list-analyzed-resources

The following code example shows how to use `list-analyzed-resources`.

AWS CLI

To list the available widgets

The following `list-analyzed-resources` example lists the available widgets in your AWS account.

```

aws accessanalyzer list-analyzed-resources \
  --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account \
  --resource-type AWS::IAM::Role

```

Output:

```

{
  "analyzedResources": [
    {
      "resourceArn": "arn:aws:sns:us-west-2:111122223333:Validation-Email",
      "resourceOwnerAccount": "111122223333",
      "resourceType": "AWS::SNS::Topic"
    },
    {
      "resourceArn": "arn:aws:sns:us-west-2:111122223333:admin-alerts",
      "resourceOwnerAccount": "111122223333",

```

```
    "resourceType": "AWS::SNS::Topic"
  },
  {
    "resourceArn": "arn:aws:sns:us-west-2:111122223333:config-topic",
    "resourceOwnerAccount": "111122223333",
    "resourceType": "AWS::SNS::Topic"
  },
  {
    "resourceArn": "arn:aws:sns:us-west-2:111122223333:inspector-topic",
    "resourceOwnerAccount": "111122223333",
    "resourceType": "AWS::SNS::Topic"
  }
]
}
```

For more information, see [Using AWS Identity and Access Management Access Analyzer](#) in the *AWS IAM User Guide*.

- For API details, see [ListAnalyzedResources](#) in *AWS CLI Command Reference*.

list-analyzers

The following code example shows how to use `list-analyzers`.

AWS CLI

To retrieve a list of analyzers

The following `list-analyzers` example retrieves a list of analyzers in your AWS account.

```
aws accessanalyzer list-analyzers
```

Output:

```
{
  "analyzers": [
    {
      "arn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/UnusedAccess-ConsoleAnalyzer-organization",
      "createdAt": "2024-02-15T00:46:40+00:00",
      "name": "UnusedAccess-ConsoleAnalyzer-organization",
      "status": "ACTIVE",
```

```

    "tags": {
      "auto-delete": "no"
    },
    "type": "ORGANIZATION_UNUSED_ACCESS"
  },
  {
    "arn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-organization",
    "createdAt": "2020-04-25T07:43:28+00:00",
    "lastResourceAnalyzed": "arn:aws:s3::DOC-EXAMPLE-BUCKET",
    "lastResourceAnalyzedAt": "2024-02-15T21:51:56.517000+00:00",
    "name": "ConsoleAnalyzer-organization",
    "status": "ACTIVE",
    "tags": {
      "auto-delete": "no"
    },
    "type": "ORGANIZATION"
  },
  {
    "arn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
ConsoleAnalyzer-account",
    "createdAt": "2019-12-03T07:28:17+00:00",
    "lastResourceAnalyzed": "arn:aws:sns:us-west-2:111122223333:config-
topic",
    "lastResourceAnalyzedAt": "2024-02-15T18:01:53.003000+00:00",
    "name": "ConsoleAnalyzer-account",
    "status": "ACTIVE",
    "tags": {
      "auto-delete": "no"
    },
    "type": "ACCOUNT"
  }
]
}

```

For more information, see [Using AWS Identity and Access Management Access Analyzer](#) in the *AWS IAM User Guide*.

- For API details, see [ListAnalyzers](#) in *AWS CLI Command Reference*.

list-archive-rules

The following code example shows how to use `list-archive-rules`.

AWS CLI

To retrieve a list of archive rules created for the specified analyzer

The following `list-archive-rules` example retrieves a list of archive rules created for the specified analyzer in your AWS account.

```
aws accessanalyzer list-archive-rules \  
  --analyzer-name UnusedAccess-ConsoleAnalyzer-organization
```

Output:

```
{  
  "archiveRules": [  
    {  
      "createdAt": "2024-02-15T00:49:27+00:00",  
      "filter": {  
        "resource": {  
          "contains": [  
            "Cognito"  
          ]  
        },  
        "resourceType": {  
          "eq": [  
            "AWS::IAM::Role"  
          ]  
        }  
      },  
      "ruleName": "MyArchiveRule",  
      "updatedAt": "2024-02-15T00:49:27+00:00"  
    },  
    {  
      "createdAt": "2024-02-15T23:27:45+00:00",  
      "filter": {  
        "findingType": {  
          "eq": [  
            "UnusedIAMUserAccessKey"  
          ]  
        }  
      },  
      "ruleName": "ArchiveRule-56125a39-e517-4ff8-afb1-ef06f58db612",  
      "updatedAt": "2024-02-15T23:27:45+00:00"  
    }  
  ]  
}
```

```
]
}
```

For more information, see [Using AWS Identity and Access Management Access Analyzer](#) in the *AWS IAM User Guide*.

- For API details, see [ListArchiveRules](#) in *AWS CLI Command Reference*.

list-findings-v2

The following code example shows how to use `list-findings-v2`.

AWS CLI

To retrieve a list of findings generated by the specified analyzer

The following `list-findings-v2` example retrieves a list of findings generated by the specified analyzer in your AWS account. This example filters the results to include only IAM roles whose name contains Cognito.

```
aws accessanalyzer list-findings-v2 \
  --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
  ConsoleAnalyzer-account \
  --filter '{"resource": {"contains": ["Cognito"]}, "resourceType": {"eq":
  ["AWS::IAM::Role"]}]'
```

Output:

```
{
  "findings": [
    {
      "analyzedAt": "2024-02-16T18:17:47.888000+00:00",
      "createdAt": "2021-02-26T21:17:24.710000+00:00",
      "id": "597f3bc2-3adc-4c18-9879-5c4b23485e46",
      "resource": "arn:aws:iam::111122223333:role/
  Cognito_testpoolUnauth_Role",
      "resourceType": "AWS::IAM::Role",
      "resourceOwnerAccount": "111122223333",
      "status": "ACTIVE",
      "updatedAt": "2021-02-26T21:17:24.710000+00:00",
      "findingType": "ExternalAccess"
    },
  ],
}
```

```

    {
      "analyzedAt": "2024-02-16T18:17:47.888000+00:00",
      "createdAt": "2021-02-26T21:17:50.905000+00:00",
      "id": "ce0e221a-85b9-4d52-91ff-d7678075442f",
      "resource": "arn:aws:iam::111122223333:role/Cognito_testpoolAuth_Role",
      "resourceType": "AWS::IAM::Role",
      "resourceOwnerAccount": "111122223333",
      "status": "ACTIVE",
      "updatedAt": "2021-02-26T21:17:50.905000+00:00",
      "findingType": "ExternalAccess"
    }
  ]
}

```

For more information, see [Using AWS Identity and Access Management Access Analyzer](#) in the *AWS IAM User Guide*.

- For API details, see [ListFindingsV2](#) in *AWS CLI Command Reference*.

list-findings

The following code example shows how to use `list-findings`.

AWS CLI

To retrieve a list of findings generated by the specified analyzer

The following `list-findings` example retrieves a list of findings generated by the specified analyzer in your AWS account. This example filters the results to include only IAM roles whose name contains Cognito.

```

aws accessanalyzer list-findings \
  --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/
  ConsoleAnalyzer-account \
  --filter '{"resource": {"contains": ["Cognito"]}, "resourceType": {"eq":
  ["AWS::IAM::Role"]}}'

```

Output:

```

{
  "findings": [
    {

```



```

    "id": "597f3bc2-3adc-4c18-9879-5c4b23485e46",
    "principal": {
      "Federated": "cognito-identity.amazonaws.com"
    },
    "action": [
      "sts:AssumeRoleWithWebIdentity"
    ],
    "resource": "arn:aws:iam::111122223333:role/
Cognito_testpoolUnauth_Role",
    "isPublic": false,
    "resourceType": "AWS::IAM::Role",
    "condition": {
      "cognito-identity.amazonaws.com:aud": "us-
west-2:EXAMPLE0-0000-0000-0000-000000000000"
    },
    "createdAt": "2021-02-26T21:17:24.710000+00:00",
    "analyzedAt": "2024-02-16T18:17:47.888000+00:00",
    "updatedAt": "2021-02-26T21:17:24.710000+00:00",
    "status": "ACTIVE",
    "resourceOwnerAccount": "111122223333"
  },
  {
    "id": "ce0e221a-85b9-4d52-91ff-d7678075442f",
    "principal": {
      "Federated": "cognito-identity.amazonaws.com"
    },
    "action": [
      "sts:AssumeRoleWithWebIdentity"
    ],
    "resource": "arn:aws:iam::111122223333:role/Cognito_testpoolAuth_Role",
    "isPublic": false,
    "resourceType": "AWS::IAM::Role",
    "condition": {
      "cognito-identity.amazonaws.com:aud": "us-
west-2:EXAMPLE0-0000-0000-0000-000000000000"
    },
    "createdAt": "2021-02-26T21:17:50.905000+00:00",
    "analyzedAt": "2024-02-16T18:17:47.888000+00:00",
    "updatedAt": "2021-02-26T21:17:50.905000+00:00",
    "status": "ACTIVE",
    "resourceOwnerAccount": "111122223333"
  }
]

```

```
}
```

For more information, see [Using AWS Identity and Access Management Access Analyzer](#) in the *AWS IAM User Guide*.

- For API details, see [ListFindings](#) in *AWS CLI Command Reference*.

list-policy-generations

The following code example shows how to use `list-policy-generations`.

AWS CLI

To list all of the policy generations requested in the last seven days

The following `list-policy-generations` example lists all of the policy generations requested in the last seven days in your AWS account.

```
aws accessanalyzer list-policy-generations
```

Output:

```
{
  "policyGenerations": [
    {
      "completedOn": "2024-02-14T23:43:38+00:00",
      "jobId": "923a56b0-ebb8-4e80-8a3c-a11ccfbcd6f2",
      "principalArn": "arn:aws:iam::111122223333:role/Admin",
      "startedOn": "2024-02-14T23:43:02+00:00",
      "status": "CANCELED"
    },
    {
      "completedOn": "2024-02-14T22:47:01+00:00",
      "jobId": "c557dc4a-0338-4489-95dd-739014860ff9",
      "principalArn": "arn:aws:iam::111122223333:role/Admin",
      "startedOn": "2024-02-14T22:44:41+00:00",
      "status": "SUCCEEDED"
    }
  ]
}
```

For more information, see [IAM Access Analyzer policy generation](#) in the *AWS IAM User Guide*.

- For API details, see [ListPolicyGenerations](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To retrieve a list of tags applied to the specified resource

The following `list-tags-for-resource` example retrieves a list of tags applied to the specified resource in your AWS account.

```
aws accessanalyzer list-tags-for-resource \  
  --resource-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/  
  ConsoleAnalyzer-account
```

Output:

```
{  
  "tags": {  
    "Zone-of-trust": "Account",  
    "Name": "ConsoleAnalyzer"  
  }  
}
```

For more information, see [IAM Access Analyzer policy generation](#) in the *AWS IAM User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

start-policy-generation

The following code example shows how to use `start-policy-generation`.

AWS CLI

To start a policy generation request

The following `start-policy-generation` example starts a policy generation request in your AWS account.

```
aws accessanalyzer start-policy-generation \
  --policy-generation-details '{"principalArn":"arn:aws:iam::111122223333:role/
Admin"}' \
  --cloud-trail-details file://myfile.json
```

Contents of `myfile.json`:

```
{
  "accessRole": "arn:aws:iam::111122223333:role/service-role/
AccessAnalyzerMonitorServiceRole",
  "startTime": "2024-02-13T00:30:00Z",
  "trails": [
    {
      "allRegions": true,
      "cloudTrailArn": "arn:aws:cloudtrail:us-west-2:111122223333:trail/my-
trail"
    }
  ]
}
```

Output:

```
{
  "jobId": "c557dc4a-0338-4489-95dd-739014860ff9"
}
```

For more information, see [IAM Access Analyzer policy generation](#) in the *AWS IAM User Guide*.

- For API details, see [StartPolicyGeneration](#) in *AWS CLI Command Reference*.

start-resource-scan

The following code example shows how to use `start-resource-scan`.

AWS CLI

To immediately start a scan of the policies applied to the specified resource

The following `start-resource-scan` example immediately starts a scan of the policies applied to the specified resource in your AWS account.

```
aws accessanalyzer start-resource-scan \  
  --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/  
ConsoleAnalyzer-account \  
  --resource-arn arn:aws:iam::111122223333:role/Cognito_testpoolAuth_Role
```

This command produces no output.

For more information, see [IAM Access Analyzer policy generation](#) in the *AWS IAM User Guide*.

- For API details, see [StartResourceScan](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To add a tag to the specified resource

The following tag-resource example adds a tag to the specified resource in your AWS account.

```
aws accessanalyzer tag-resource \  
  --resource-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/  
ConsoleAnalyzer-account \  
  --tags Environment=dev,Purpose=testing
```

This command produces no output.

For more information, see [Using AWS Identity and Access Management Access Analyzer](#) in the *AWS IAM User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use untag-resource.

AWS CLI

To remove tags from the specified resources

The following `untag-resource` example removes tags from the specified resource in your AWS account.

```
aws accessanalyzer untag-resource \  
  --resource-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/  
ConsoleAnalyzer-account \  
  --tag-keys Environment Purpose
```

This command produces no output.

For more information, see [Using AWS Identity and Access Management Access Analyzer](#) in the *AWS IAM User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-archive-rule

The following code example shows how to use `update-archive-rule`.

AWS CLI

To update the criteria and values for the specified archive rule

The following `update-archive-rule` example updates the criteria and values for the specified archive rule in your AWS account.

```
aws accessanalyzer update-archive-rule \  
  --analyzer-name UnusedAccess-ConsoleAnalyzer-organization \  
  --rule-name MyArchiveRule \  
  --filter '{"resource": {"contains": ["Cognito"]}, "resourceType": {"eq":  
["AWS::IAM::Role"]}]'
```

This command produces no output.

For more information, see [Archive rules](#) in the *AWS IAM User Guide*.

- For API details, see [UpdateArchiveRule](#) in *AWS CLI Command Reference*.

update-findings

The following code example shows how to use `update-findings`.

AWS CLI

To update the status for the specified findings

The following `update-findings` example updates the status for the specified findings in your AWS account.

```
aws accessanalyzer update-findings \  
  --analyzer-arn arn:aws:access-analyzer:us-west-2:111122223333:analyzer/  
UnusedAccess-ConsoleAnalyzer-organization \  
  --ids 4f319ac3-2e0c-4dc4-bf51-7013a086b6ae 780d586a-2cce-4f72-aff6-359d450e7500  
 \  
  --status ARCHIVED
```

This command produces no output.

For more information, see [Using AWS Identity and Access Management Access Analyzer](#) in the *AWS IAM User Guide*.

- For API details, see [UpdateFindings](#) in *AWS CLI Command Reference*.

validate-policy

The following code example shows how to use `validate-policy`.

AWS CLI

To request the validation of a policy and returns a list of findings

The following `validate-policy` example requests the validation of a policy and returns a list of findings. The policy in the example is a role trust policy for an Amazon Cognito role used for web identity federation. The findings generated from the trust policy relate to an empty `Sid` element value and a mismatched policy principal due to the incorrect assume role action being used, `sts:AssumeRole`. The correct assume role action for use with Cognito is `sts:AssumeRoleWithWebIdentity`.

```
aws accessanalyzer validate-policy \  
  --policy-document file://myfile.json \  
  --policy-type RESOURCE_POLICY
```

Contents of `myfile.json`:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Federated": "cognito-identity.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ],
      "Condition": {
        "StringEquals": {
          "cognito-identity.amazonaws.com:aud": "us-west-2_EXAMPLE"
        }
      }
    }
  ]
}
```

Output:

```
{
  "findings": [
    {
      "findingDetails": "Add a value to the empty string in the Sid element.",
      "findingType": "SUGGESTION",
      "issueCode": "EMPTY_SID_VALUE",
      "learnMoreLink": "https://docs.aws.amazon.com/IAM/latest/UserGuide/access-analyzer-reference-policy-checks.html#access-analyzer-reference-policy-checks-suggestion-empty-sid-value",
      "locations": [
        {
          "path": [
            {
              "value": "Statement"
            },
            {
              "index": 0
            }
          ]
        }
      ]
    }
  ]
}
```



```

        "value": "Sid"
      }
    ],
    "span": {
      "end": {
        "column": 21,
        "line": 5,
        "offset": 81
      },
      "start": {
        "column": 19,
        "line": 5,
        "offset": 79
      }
    }
  }
]
},
{
  "findingDetails": "The sts:AssumeRole action is invalid with the
following principal(s): cognito-identity.amazonaws.com. Use a SAML provider
principal with the sts:AssumeRoleWithSAML action or use an OIDC provider principal
with the sts:AssumeRoleWithWebIdentity action. Ensure the provider is Federated if
you use either of the two options.",
  "findingType": "ERROR",
  "issueCode": "MISMATCHED_ACTION_FOR_PRINCIPAL",
  "learnMoreLink": "https://docs.aws.amazon.com/IAM/latest/UserGuide/
access-analyzer-reference-policy-checks.html#access-analyzer-reference-policy-
checks-error-mismatched-action-for-principal",
  "locations": [
    {
      "path": [
        {
          "value": "Statement"
        },
        {
          "index": 0
        },
        {
          "value": "Action"
        },
        {
          "index": 0
        }
      ]
    }
  ]
}

```

```
    ],
    "span": {
      "end": {
        "column": 32,
        "line": 11,
        "offset": 274
      },
      "start": {
        "column": 16,
        "line": 11,
        "offset": 258
      }
    }
  },
  {
    "path": [
      {
        "value": "Statement"
      },
      {
        "index": 0
      },
      {
        "value": "Principal"
      },
      {
        "value": "Federated"
      }
    ],
    "span": {
      "end": {
        "column": 61,
        "line": 8,
        "offset": 202
      },
      "start": {
        "column": 29,
        "line": 8,
        "offset": 170
      }
    }
  }
]
},
```

```

    {
      "findingDetails": "The following actions: sts:TagSession are not
supported by the condition key cognito-identity.amazonaws.com:aud. The condition
will not be evaluated for these actions. We recommend that you move these actions
to a different statement without this condition key.",
      "findingType": "ERROR",
      "issueCode": "UNSUPPORTED_ACTION_FOR_CONDITION_KEY",
      "learnMoreLink": "https://docs.aws.amazon.com/IAM/latest/UserGuide/
access-analyzer-reference-policy-checks.html#access-analyzer-reference-policy-
checks-error-unsupported-action-for-condition-key",
      "locations": [
        {
          "path": [
            {
              "value": "Statement"
            },
            {
              "index": 0
            },
            {
              "value": "Action"
            },
            {
              "index": 1
            }
          ],
          "span": {
            "end": {
              "column": 32,
              "line": 12,
              "offset": 308
            },
            "start": {
              "column": 16,
              "line": 12,
              "offset": 292
            }
          }
        },
        {
          "path": [
            {
              "value": "Statement"
            },

```

```
    {
      "index": 0
    },
    {
      "value": "Condition"
    },
    {
      "value": "StringEquals"
    },
    {
      "value": "cognito-identity.amazonaws.com:aud"
    }
  ],
  "span": {
    "end": {
      "column": 79,
      "line": 16,
      "offset": 464
    },
    "start": {
      "column": 58,
      "line": 16,
      "offset": 443
    }
  }
}
]
```

For more information, see [Checks for validating policies](#) in the *AWS IAM User Guide*.

- For API details, see [ValidatePolicy](#) in *AWS CLI Command Reference*.

Image Builder examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Image Builder.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-component

The following code example shows how to use `create-component`.

AWS CLI

To create a component

The following `create-component` example creates a component that uses a JSON document file and references a component document in YAML format that is uploaded to an Amazon S3 bucket.

```
aws imagebuilder create-component \  
  --cli-input-json file://create-component.json
```

Contents of `create-component.json`:

```
{  
  "name": "MyExampleComponent",  
  "semanticVersion": "2019.12.02",  
  "description": "An example component that builds, validates and tests an image",  
  "changeDescription": "Initial version.",  
  "platform": "Windows",  
  "uri": "s3://s3-bucket-name/s3-bucket-path/component.yaml"  
}
```

Output:

```
{
```

```
"requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
"clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
"componentBuildVersionArn": "arn:aws:imagebuilder:us-
west-2:123456789012:component/examplecomponent/2019.12.02/1"
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [CreateComponent](#) in *AWS CLI Command Reference*.

create-distribution-configuration

The following code example shows how to use create-distribution-configuration.

AWS CLI

To create a distribution configuration

The following create-distribution-configuration example creates a distribution configuration using a JSON file.

```
aws imagebuilder create-distribution-configuration \
  --cli-input-json file:/create-distribution-configuration.json
```

Contents of create-distribution-configuration.json:

```
{
  "name": "MyExampleDistribution",
  "description": "Copies AMI to eu-west-1",
  "distributions": [
    {
      "region": "us-west-2",
      "amiDistributionConfiguration": {
        "name": "Name {{imagebuilder:buildDate}}",
        "description": "An example image name with parameter references",
        "amiTags": {
          "KeyName": "{{ssm:parameter_name}}"
        },
        "launchPermission": {
          "userIds": [
            "123456789012"
          ]
        }
      }
    }
  ]
}
```

```

    ]
  }
},
{
  "region": "eu-west-1",
  "amiDistributionConfiguration": {
    "name": "My {{imagebuilder:buildVersion}} image
{{imagebuilder:buildDate}}",
    "amiTags": {
      "KeyName": "Value"
    },
    "launchPermission": {
      "userIds": [
        "123456789012"
      ]
    }
  }
}
]
}

```

Output:

```

{
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/myexampledistribution"
}

```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [CreateDistributionConfiguration](#) in *AWS CLI Command Reference*.

create-image-pipeline

The following code example shows how to use `create-image-pipeline`.

AWS CLI**To create an image pipeline**

The following `create-image-pipeline` example creates an image pipeline using a JSON file.

```
aws imagebuilder create-image-pipeline \  
  --cli-input-json file://create-image-pipeline.json
```

Contents of `create-image-pipeline.json`:

```
{  
  "name": "MyWindows2016Pipeline",  
  "description": "Builds Windows 2016 Images",  
  "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/  
mybasicrecipe/2019.12.03",  
  "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-  
west-2:123456789012:infrastructure-configuration/myexampleinfrastructure",  
  "distributionConfigurationArn": "arn:aws:imagebuilder:us-  
west-2:123456789012:distribution-configuration/myexampledistribution",  
  "imageTestsConfiguration": {  
    "imageTestsEnabled": true,  
    "timeoutMinutes": 60  
  },  
  "schedule": {  
    "scheduleExpression": "cron(0 0 * * SUN)",  
    "pipelineExecutionStartCondition":  
"EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"  
  },  
  "status": "ENABLED"  
}
```

Output:

```
{  
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
  "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",  
  "imagePipelineArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/  
mywindows2016pipeline"  
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [CreateImagePipeline](#) in *AWS CLI Command Reference*.

create-image-recipe

The following code example shows how to use `create-image-recipe`.

AWS CLI

To create a recipe

The following `create-image-recipe` example creates an image recipe using a JSON file. Components are installed in the order in which they are specified.

```
aws imagebuilder create-image-recipe \  
  --cli-input-json file://create-image-recipe.json
```

Contents of `create-image-recipe.json`:

```
{  
  "name": "MyBasicRecipe",  
  "description": "This example image recipe creates a Windows 2016 image.",  
  "semanticVersion": "2019.12.03",  
  "components":  
  [  
    {  
      "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/  
myexamplecomponent/2019.12.02/1"  
    },  
    {  
      "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/  
myimportedcomponent/1.0.0/1"  
    }  
  ],  
  "parentImage": "arn:aws:imagebuilder:us-west-2:aws:image/windows-server-2016-  
english-full-base-x86/xxxx.x.x"  
}
```

Output:

```
{  
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
  "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",  
  "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/  
mybasicrecipe/2019.12.03"
```

```
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [CreateImageRecipe](#) in *AWS CLI Command Reference*.

create-image

The following code example shows how to use `create-image`.

AWS CLI

To create an image

The following `create-image` example creates an image.

```
aws imagebuilder create-image \  
  --image-recipe-arn arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/  
mybasicrecipe/2019.12.03 \  
  --infrastructure-configuration-arn arn:aws:imagebuilder:us-  
west-2:123456789012:infrastructure-configuration/myexampleinfrastructure
```

Output:

```
{  
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
  "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",  
  "imageBuildVersionArn": "arn:aws:imagebuilder:us-west-2:123456789012:image/  
mybasicrecipe/2019.12.03/1"  
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [CreateImage](#) in *AWS CLI Command Reference*.

create-infrastructure-configuration

The following code example shows how to use `create-infrastructure-configuration`.

AWS CLI

To create an infrastructure configuration

The following `create-infrastructure-configuration` example creates an infrastructure configuration using a JSON file.

```
aws imagebuilder create-infrastructure-configuration \  
  --cli-input-json file://create-infrastructure-configuration.json
```

Contents of `create-infrastructure-configuration.json`:

```
{  
  "name": "MyExampleInfrastructure",  
  "description": "An example that will retain instances of failed builds",  
  "instanceTypes": [  
    "m5.large", "m5.xlarge"  
  ],  
  "instanceProfileName": "EC2InstanceProfileForImageBuilder",  
  "securityGroupIds": [  
    "sg-a1b2c3d4"  
  ],  
  "subnetId": "subnet-a1b2c3d4",  
  "logging": {  
    "s3Logs": {  
      "s3BucketName": "bucket-name",  
      "s3KeyPrefix": "bucket-path"  
    }  
  },  
  "keyPair": "key-pair-name",  
  "terminateInstanceOnFailure": false,  
  "snsTopicArn": "arn:aws:sns:us-west-2:123456789012:sns-topic-name"  
}
```

Output:

```
{  
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
  "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",  
  "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-  
west-2:123456789012:infrastructure-configuration/myexampleinfrastructure"  
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [CreateInfrastructureConfiguration](#) in *AWS CLI Command Reference*.

delete-component

The following code example shows how to use delete-component.

AWS CLI

To delete a component

The following delete-component example deletes a component build version by specifying its ARN.

```
aws imagebuilder delete-component \  
  --component-build-version-arn arn:aws:imagebuilder:us-  
west-2:123456789012:component/myexamplecomponent/2019.12.02/1
```

Output:

```
{  
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
  "componentBuildVersionArn": "arn:aws:imagebuilder:us-  
west-2:123456789012:component/myexamplecomponent/2019.12.02/1"  
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [DeleteComponent](#) in *AWS CLI Command Reference*.

delete-image-pipeline

The following code example shows how to use delete-image-pipeline.

AWS CLI

To delete an image pipeline

The following `delete-image-pipeline` example deletes an image pipeline by specifying its ARN.

```
aws imagebuilder delete-image-pipeline \  
  --image-pipeline-arn arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/  
  my-example-pipeline
```

Output:

```
{  
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
  "imagePipelineArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/  
  mywindows2016pipeline"  
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [DeleteImagePipeline](#) in *AWS CLI Command Reference*.

delete-image-recipe

The following code example shows how to use `delete-image-recipe`.

AWS CLI

To delete an image recipe

The following `delete-image-recipe` example deletes an image recipe by specifying its ARN.

```
aws imagebuilder delete-image-recipe \  
  --image-recipe-arn arn:aws:imagebuilder:us-east-1:123456789012:image-recipe/  
  mybasicrecipe/2019.12.03
```

Output:

```
{  
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
  "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/  
  mybasicrecipe/2019.12.03"  
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [DeleteImageRecipe](#) in *AWS CLI Command Reference*.

delete-image

The following code example shows how to use `delete-image`.

AWS CLI

To delete an image

The following `delete-image` example deletes an image build version by specifying its ARN.

```
aws imagebuilder delete-image \  
  --image-build-version-arn arn:aws:imagebuilder:us-west-2:123456789012:image/my-  
example-image/2019.12.02/1
```

Output:

```
{  
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
  "imageBuildVersionArn": "arn:aws:imagebuilder:us-west-2:123456789012:image/  
mybasicrecipe/2019.12.03/1"  
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [DeleteImage](#) in *AWS CLI Command Reference*.

delete-infrastructure-configuration

The following code example shows how to use `delete-infrastructure-configuration`.

AWS CLI

To delete an infrastructure configuration

The following `delete-infrastructure-configuration` example deletes an image pipeline by specifying its ARN.

```
aws imagebuilder delete-infrastructure-configuration \  
  --infrastructure-configuration-arn arn:aws:imagebuilder:us-  
east-1:123456789012:infrastructure-configuration/myexampleinfrastructure
```

Output:

```
{  
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
  "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-  
west-2:123456789012:infrastructure-configuration/myexampleinfrastructure"  
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [DeleteInfrastructureConfiguration](#) in *AWS CLI Command Reference*.

get-component-policy

The following code example shows how to use `get-component-policy`.

AWS CLI

To get component policy details

The following `get-component-policy` example lists the details of a component policy by specifying its ARN.

```
aws imagebuilder get-component-policy \  
  --component-arn arn:aws:imagebuilder:us-west-2:123456789012:component/my-  
example-component/2019.12.03/1
```

Output:

```
{  
  "Policy": "{ \"Version\": \"2012-10-17\", \"Statement\": [ { \"Effect\":  
\"Allow\", \"Principal\": { \"AWS\": [ \"123456789012\" ] }, \"Action\":  
[ \"imagebuilder:GetComponent\", \"imagebuilder:ListComponents\" ], \"Resource\":  
[ \"arn:aws:imagebuilder:us-west-2:123456789012:component/my-example-  
component/2019.12.03/1\" ] } ] }"
```

```
}

```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](https://docs.aws.amazon.com/imagebuilder/latest/userguide/managing-image-builder-cli.html) <<https://docs.aws.amazon.com/imagebuilder/latest/userguide/managing-image-builder-cli.html>> `__` in the *EC2 Image Builder Users Guide*.

- For API details, see [GetComponentPolicy](#) in *AWS CLI Command Reference*.

get-component

The following code example shows how to use `get-component`.

AWS CLI

To get component details

The following `get-component` example lists the details of a component by specifying its ARN.

```
aws imagebuilder get-component \
  --component-build-version-arn arn:aws:imagebuilder:us-
west-2:123456789012:component/component-name/1.0.0/1
```

Output:

```
{
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "component": {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/component-
name/1.0.0/1",
    "name": "component-name",
    "version": "1.0.0",
    "type": "TEST",
    "platform": "Linux",
    "owner": "123456789012",
    "data": "name: HelloWorldTestingDocument\ndescription: This is hello world
testing document.\nschemaVersion: 1.0\n\nphases:\n - name: test\n   steps:\n
- name: HelloWorldStep\n       action: ExecuteBash\n       inputs:\n
commands:\n         - echo \"Hello World! Test.\"\n",
    "encrypted": true,
    "dateCreated": "2020-01-27T20:43:30.306Z",
    "tags": {}
  }
}
```



```
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [GetComponent](#) in *AWS CLI Command Reference*.

get-distribution-configuration

The following code example shows how to use `get-distribution-configuration`.

AWS CLI

To get the details of a distribution configuration

The following `get-distribution-configuration` example displays the details of a distribution configuration by specifying its ARN.

```
aws imagebuilder get-distribution-configuration \
  --distribution-configuration-arn arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/myexempldistribution
```

Output:

```
{
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "distributionConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:distribution-
configuration/myexempldistribution",
    "name": "MyExampleDistribution",
    "description": "Copies AMI to eu-west-1 and exports to S3",
    "distributions": [
      {
        "region": "us-west-2",
        "amiDistributionConfiguration": {
          "name": "Name {{imagebuilder:buildDate}}",
          "description": "An example image name with parameter
references",
          "amiTags": {
            "KeyName": "{{ssm:parameter_name}}"
          },
          "launchPermission": {
            "userIds": [
```

```

        "123456789012"
    ]
  }
},
{
  "region": "eu-west-1",
  "amiDistributionConfiguration": {
    "name": "My {{imagebuilder:buildVersion}} image
{{imagebuilder:buildDate}}",
    "amiTags": {
      "KeyName": "Value"
    },
    "launchPermission": {
      "userIds": [
        "123456789012"
      ]
    }
  }
},
],
"dateCreated": "2020-02-19T18:40:10.529Z",
"tags": {}
}
}

```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [GetDistributionConfiguration](#) in *AWS CLI Command Reference*.

get-image-pipeline

The following code example shows how to use `get-image-pipeline`.

AWS CLI

To get image pipeline details

The following `get-image-pipeline` example lists the details of an image pipeline by specifying its ARN.

```
aws imagebuilder get-image-pipeline \
```

```
--image-pipeline-arn arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/
mywindows2016pipeline
```

Output:

```
{
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "imagePipeline": {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/
mywindows2016pipeline",
    "name": "MyWindows2016Pipeline",
    "description": "Builds Windows 2016 Images",
    "platform": "Windows",
    "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/
mybasicrecipe/2019.12.03",
    "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/myexampleinfrastructure",
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/myexampledistribution",
    "imageTestsConfiguration": {
      "imageTestsEnabled": true,
      "timeoutMinutes": 60
    },
    "schedule": {
      "scheduleExpression": "cron(0 0 * * SUN)",
      "pipelineExecutionStartCondition":
"EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
    },
    "status": "ENABLED",
    "dateCreated": "2020-02-19T19:04:01.253Z",
    "dateUpdated": "2020-02-19T19:04:01.253Z",
    "tags": {}
  }
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [GetImagePipeline](#) in *AWS CLI Command Reference*.

get-image-policy

The following code example shows how to use `get-image-policy`.

AWS CLI

To get image policy details

The following `get-image-policy` example lists the details of an image policy by specifying its ARN.

```
aws imagebuilder get-image-policy \  
  --image-arn arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-  
image/2019.12.03/1
```

Output:

```
{  
  "Policy": "{ \"Version\": \"2012-10-17\", \"Statement\": [ { \"Effect\": \"Allow\",  
\"Principal\": { \"AWS\": [ \"123456789012\" ] }, \"Action\": [ \"imagebuilder:GetImage\",  
\"imagebuilder:ListImages\" ], \"Resource\": [ \"arn:aws:imagebuilder:us-  
west-2:123456789012:image/my-example-image/2019.12.03/1\" ] } ] }"
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [GetImagePolicy](#) in *AWS CLI Command Reference*.

get-image-recipe-policy

The following code example shows how to use `get-image-recipe-policy`.

AWS CLI

To get image recipe policy details

The following `get-image-recipe-policy` example lists the details of an image recipe policy by specifying its ARN.

```
aws imagebuilder get-image-recipe-policy \  
  --image-recipe-arn arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-  
example-image-recipe/2019.12.03/1
```

Output:

```
{
  "Policy": "{ \"Version\": \"2012-10-17\", \"Statement\": [ { \"Effect\":
\"Allow\", \"Principal\": { \"AWS\": [ \"123456789012\" ] }, \"Action\":
[ \"imagebuilder:GetImageRecipe\", \"imagebuilder:ListImageRecipes\" ], \"Resource\":
[ \"arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-image-
recipe/2019.12.03/1\" ] } ] }"
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [GetImageRecipePolicy](#) in *AWS CLI Command Reference*.

get-image

The following code example shows how to use `get-image`.

AWS CLI

To get image details

The following `get-image` example lists the details of an image by specifying its ARN.

```
aws imagebuilder get-image \
  --image-build-version-arn arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03/1
```

Output:

```
{
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "image": {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03/1",
    "name": "MyBasicRecipe",
    "version": "2019.12.03/1",
    "platform": "Windows",
    "state": {
      "status": "BUILDING"
    },
    "imageRecipe": {
```

```
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/
mybasicrecipe/2019.12.03",
    "name": "MyBasicRecipe",
    "description": "This example image recipe creates a Windows 2016
image.",
    "platform": "Windows",
    "version": "2019.12.03",
    "components": [
      {
        "componentArn": "arn:aws:imagebuilder:us-
west-2:123456789012:component/myexamplecomponent/2019.12.02/1"
      },
      {
        "componentArn": "arn:aws:imagebuilder:us-
west-2:123456789012:component/myimportedcomponent/1.0.0/1"
      }
    ],
    "parentImage": "arn:aws:imagebuilder:us-west-2:aws:image/windows-
server-2016-english-full-base-x86/2019.12.17/1",
    "dateCreated": "2020-02-14T19:46:16.904Z",
    "tags": {}
  },
  "infrastructureConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-
configuration/myexampleinfrastructure",
    "name": "MyExampleInfrastructure",
    "description": "An example that will retain instances of failed builds",
    "instanceTypes": [
      "m5.large",
      "m5.xlarge"
    ],
    "instanceProfileName": "EC2InstanceProfileForImageFactory",
    "securityGroupIds": [
      "sg-a1b2c3d4"
    ],
    "subnetId": "subnet-a1b2c3d4",
    "logging": {
      "s3Logs": {
        "s3BucketName": "bucket-name",
        "s3KeyPrefix": "bucket-path"
      }
    }
  },
  "keyPair": "Sam",
  "terminateInstanceOnFailure": false,
```

```

        "snsTopicArn": "arn:aws:sns:us-west-2:123456789012:sns-name",
        "dateCreated": "2020-02-14T21:21:05.098Z",
        "tags": {}
    },
    "imageTestsConfiguration": {
        "imageTestsEnabled": true,
        "timeoutMinutes": 720
    },
    "dateCreated": "2020-02-14T23:14:13.597Z",
    "outputResources": {
        "amis": []
    },
    "tags": {}
}
}

```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [GetImage](#) in *AWS CLI Command Reference*.

get-infrastructure-configuration

The following code example shows how to use `get-infrastructure-configuration`.

AWS CLI

To get infrastructure configuration details

The following `get-infrastructure-configuration` example lists the details of an infrastructure configuration by specifying its ARN.

```

aws imagebuilder get-infrastructure-configuration \
  --infrastructure-configuration-arn arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/myexampleinfrastructure

```

Output:

```

{
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "infrastructureConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-
configuration/myexampleinfrastructure",

```

```
    "name": "MyExampleInfrastructure",
    "description": "An example that will retain instances of failed builds",
    "instanceTypes": [
      "m5.large",
      "m5.xlarge"
    ],
    "instanceProfileName": "EC2InstanceProfileForImageBuilder",
    "securityGroupIds": [
      "sg-a48c95ef"
    ],
    "subnetId": "subnet-a48c95ef",
    "logging": {
      "s3Logs": {
        "s3BucketName": "bucket-name",
        "s3KeyPrefix": "bucket-path"
      }
    },
    "keyPair": "Name",
    "terminateInstanceOnFailure": false,
    "snsTopicArn": "arn:aws:sns:us-west-2:123456789012:sns-name",
    "dateCreated": "2020-02-19T19:11:51.858Z",
    "tags": {}
  }
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [GetInfrastructureConfiguration](#) in *AWS CLI Command Reference*.

import-component

The following code example shows how to use `import-component`.

AWS CLI

To import a component

The following `import-component` example imports a preexisting script using a JSON file.

```
aws imagebuilder import-component \  
  --cli-input-json file://import-component.json
```


Contents of `import-component.json`:

```
{
  "name": "MyImportedComponent",
  "semanticVersion": "1.0.0",
  "description": "An example of how to import a component",
  "changeDescription": "First commit message.",
  "format": "SHELL",
  "platform": "Windows",
  "type": "BUILD",
  "uri": "s3://s3-bucket-name/s3-bucket-path/component.yaml"
}
```

Output:

```
{
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "componentBuildVersionArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/myimportedcomponent/1.0.0/1"
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [ImportComponent](#) in *AWS CLI Command Reference*.

`list-component-build-versions`

The following code example shows how to use `list-component-build-versions`.

AWS CLI

To list component build versions

The following `list-component-build-versions` example lists the component build versions with a specific semantic version.

```
aws imagebuilder list-component-build-versions --component-version-arn
arn:aws:imagebuilder:us-west-2:123456789012:component/myexamplecomponent/2019.12.02
```

Output:

```
{
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "componentSummaryList": [
    {
      "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/myexamplecomponent/2019.12.02/1",
      "name": "MyExampleComponent",
      "version": "2019.12.02",
      "platform": "Windows",
      "type": "BUILD",
      "owner": "123456789012",
      "description": "An example component that builds, validates and tests an image",
      "changeDescription": "Initial version.",
      "dateCreated": "2020-02-19T18:53:45.940Z",
      "tags": {
        "KeyName": "KeyValue"
      }
    }
  ]
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [ListComponentBuildVersions](#) in *AWS CLI Command Reference*.

list-components

The following code example shows how to use `list-components`.

AWS CLI

To list all of the component semantic versions

The following `list-components` example lists all of the component semantic versions to which you have access. You can optionally filter on whether to list components owned by you, by Amazon, or that have been shared with you by other accounts.

```
aws imagebuilder list-components
```

Output:

```
{
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "componentVersionList": [
    {
      "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/component-
name/1.0.0",
      "name": "component-name",
      "version": "1.0.0",
      "platform": "Linux",
      "type": "TEST",
      "owner": "123456789012",
      "dateCreated": "2020-01-27T20:43:30.306Z"
    }
  ]
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [ListComponents](#) in *AWS CLI Command Reference*.

list-distribution-configurations

The following code example shows how to use `list-distribution-configurations`.

AWS CLI

To list distributions

The following `list-distribution-configurations` example lists all of your distributions.

```
aws imagebuilder list-distribution-configurations
```

Output:

```
{
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "distributionConfigurationSummaryList": [
    {
      "arn": "arn:aws:imagebuilder:us-west-2:123456789012:distribution-
configuration/myexampledistribution",
      "name": "MyExampleDistribution",
```

```

        "description": "Copies AMI to eu-west-1 and exports to S3",
        "dateCreated": "2020-02-19T18:40:10.529Z",
        "tags": {
            "KeyName": "KeyValue"
        }
    ]
}

```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [ListDistributionConfigurations](#) in *AWS CLI Command Reference*.

list-image-build-versions

The following code example shows how to use `list-image-build-versions`.

AWS CLI

To list image build versions

The following `list-image-build-versions` example lists all of the image build versions with a semantic version.

```

aws imagebuilder list-image-build-versions \
  --image-version-arn arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03

```

Output:

```

{
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "imageSummaryList": [
    {
      "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03/7",
      "name": "MyBasicRecipe",
      "version": "2019.12.03/7",
      "platform": "Windows",
      "state": {
        "status": "FAILED",

```

```

        "reason": "Can't start SSM Automation for arn
arn:aws:imagebuilder:us-west-2:123456789012:image/mybasicrecipe/2019.12.03/7 during
building. Parameter \"iamInstanceProfileName\" has a null value."
    },
    "owner": "123456789012",
    "dateCreated": "2020-02-19T18:56:11.511Z",
    "outputResources": {
        "amis": []
    },
    },
    {
        "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03/6",
        "name": "MyBasicRecipe",
        "version": "2019.12.03/6",
        "platform": "Windows",
        "state": {
            "status": "FAILED",
            "reason": "An internal error has occurred."
        },
        "owner": "123456789012",
        "dateCreated": "2020-02-18T22:49:08.142Z",
        "outputResources": {
            "amis": [
                {
                    "region": "us-west-2",
                    "image": "ami-a1b2c3d4567890ab",
                    "name": "MyBasicRecipe 2020-02-18T22-49-38.704Z",
                    "description": "This example image recipe creates a Windows
2016 image."
                },
                {
                    "region": "us-west-2",
                    "image": "ami-a1b2c3d4567890ab",
                    "name": "Name 2020-02-18T22-49-08.131Z",
                    "description": "Copies AMI to eu-west-2 and exports to S3"
                },
                {
                    "region": "eu-west-2",
                    "image": "ami-a1b2c3d4567890ab",
                    "name": "My 6 image 2020-02-18T22-49-08.131Z",
                    "description": "Copies AMI to eu-west-2 and exports to S3"
                }
            ]
        }
    }
}

```

```
    ]
  },
  "tags": {}
},
{
  "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03/5",
  "name": "MyBasicRecipe",
  "version": "2019.12.03/5",
  "platform": "Windows",
  "state": {
    "status": "AVAILABLE"
  },
  "owner": "123456789012",
  "dateCreated": "2020-02-18T16:51:48.403Z",
  "outputResources": {
    "amis": [
      {
        "region": "us-west-2",
        "image": "ami-a1b2c3d4567890ab",
        "name": "MyBasicRecipe 2020-02-18T16-52-18.965Z",
        "description": "This example image recipe creates a Windows
2016 image."
      }
    ]
  },
  "tags": {}
},
{
  "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03/4",
  "name": "MyBasicRecipe",
  "version": "2019.12.03/4",
  "platform": "Windows",
  "state": {
    "status": "AVAILABLE"
  },
  "owner": "123456789012",
  "dateCreated": "2020-02-18T16:50:01.827Z",
  "outputResources": {
    "amis": [
      {
        "region": "us-west-2",
        "image": "ami-a1b2c3d4567890ab",
```

```

                "name": "MyBasicRecipe 2020-02-18T16-50-32.280Z",
                "description": "This example image recipe creates a Windows
2016 image."
            }
        ]
    },
    "tags": {}
},
{
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03/3",
    "name": "MyBasicRecipe",
    "version": "2019.12.03/3",
    "platform": "Windows",
    "state": {
        "status": "AVAILABLE"
    },
    "owner": "123456789012",
    "dateCreated": "2020-02-14T23:14:13.597Z",
    "outputResources": {
        "amis": [
            {
                "region": "us-west-2",
                "image": "ami-a1b2c3d4567890ab",
                "name": "MyBasicRecipe 2020-02-14T23-14-44.243Z",
                "description": "This example image recipe creates a Windows
2016 image."
            }
        ]
    },
    "tags": {}
},
{
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03/2",
    "name": "MyBasicRecipe",
    "version": "2019.12.03/2",
    "platform": "Windows",
    "state": {
        "status": "FAILED",
        "reason": "SSM execution 'a1b2c3d4-5678-90ab-cdef-EXAMPLE11111'
failed with status = 'Failed' and failure message = 'Step fails when it is
verifying the command has completed. Command a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
returns unexpected invocation result: \n{Status=[Failed], ResponseCode=[1],

```

```

Output=[\n-----ERROR-----\nfailed to run commands: exit status 1],
OutputPayload=[{"Status\":"Failed","\nResponseCode\":1,\nOutput\":"\n-----ERROR-----\nfailed to run commands: exit status 1","\nCommandId\":"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"}], CommandId=[a1b2c3d4-5678-90ab-cdef-EXAMPLE11111]}. Please refer to Automation Service Troubleshooting Guide for more diagnosis details.'"
    },
    "owner": "123456789012",
    "dateCreated": "2020-02-14T22:57:42.593Z",
    "outputResources": {
      "amis": []
    },
    "tags": {}
  }
]
}

```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [ListImageBuildVersions](#) in *AWS CLI Command Reference*.

list-image-pipeline-images

The following code example shows how to use `list-image-pipeline-images`.

AWS CLI

To list image pipeline pipeline images

The following `list-image-pipeline-images` example lists all images that were created by a specific image pipeline.

```

aws imagebuilder list-image-pipeline-images \
  --image-pipeline-arn arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/
mywindows2016pipeline

```

Output:

```

{
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "imagePipelineList": [
    {

```



```

    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/
mywindows2016pipeline",
    "name": "MyWindows2016Pipeline",
    "description": "Builds Windows 2016 Images",
    "platform": "Windows",
    "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-
recipe/mybasicrecipe/2019.12.03",
    "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/myexampleinfrastructure",
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/myexampledistribution",
    "imageTestsConfiguration": {
        "imageTestsEnabled": true,
        "timeoutMinutes": 60
    },
    "schedule": {
        "scheduleExpression": "cron(0 0 * * SUN)",
        "pipelineExecutionStartCondition":
"EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
    },
    "status": "ENABLED",
    "dateCreated": "2020-02-19T19:04:01.253Z",
    "dateUpdated": "2020-02-19T19:04:01.253Z",
    "tags": {
        "KeyName": "KeyValue"
    }
},
{
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/sam",
    "name": "PipelineName",
    "platform": "Linux",
    "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-
recipe/recipe-name-a1b2c3d45678/1.0.0",
    "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/infrastructureconfiguration-name-
a1b2c3d45678",
    "imageTestsConfiguration": {
        "imageTestsEnabled": true,
        "timeoutMinutes": 720
    },
    "status": "ENABLED",
    "dateCreated": "2019-12-16T18:19:02.068Z",
    "dateUpdated": "2019-12-16T18:19:02.068Z",
    "tags": {

```

```
        "KeyName": "KeyValue"
      }
    ]
  }
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [ListImagePipelineImages](#) in *AWS CLI Command Reference*.

list-image-recipes

The following code example shows how to use `list-image-recipes`.

AWS CLI

To list image recipes

The following `list-image-recipes` example lists all of your image recipes.

```
aws imagebuilder list-image-recipes
```

Output:

```
{
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "imageRecipeSummaryList": [
    {
      "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/mybasicrecipe/2019.12.03",
      "name": "MyBasicRecipe",
      "platform": "Windows",
      "owner": "123456789012",
      "parentImage": "arn:aws:imagebuilder:us-west-2:aws:image/windows-server-2016-english-full-base-x86/2019.x.x",
      "dateCreated": "2020-02-19T18:54:25.975Z",
      "tags": {
        "KeyName": "KeyValue"
      }
    },
    {
```

```

    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/recipe-
name-a1b2c3d45678/1.0.0",
    "name": "recipe-name-a1b2c3d45678",
    "platform": "Linux",
    "owner": "123456789012",
    "parentImage": "arn:aws:imagebuilder:us-west-2:aws:image/amazon-linux-2-
x86/2019.11.21",
    "dateCreated": "2019-12-16T18:19:00.120Z",
    "tags": {
      "KeyName": "KeyValue"
    }
  }
]
}

```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [ListImageRecipes](#) in *AWS CLI Command Reference*.

list-images

The following code example shows how to use `list-images`.

AWS CLI

To list images

The following `list-images` example lists all of the semantic versions you have access to.

```
aws imagebuilder list-images
```

Output:

```

{
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "imageVersionList": [
    {
      "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03",
      "name": "MyBasicRecipe",
      "version": "2019.12.03",
      "platform": "Windows",

```

```

        "owner": "123456789012",
        "dateCreated": "2020-02-14T21:29:18.810Z"
    }
]
}

```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [ListImages](#) in *AWS CLI Command Reference*.

list-infrastructure-configurations

The following code example shows how to use `list-infrastructure-configurations`.

AWS CLI

To list infrastructure configurations

The following `list-infrastructure-configurations` example lists all of your infrastructure configurations.

```
aws imagebuilder list-infrastructure-configurations
```

Output:

```

{
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "infrastructureConfigurationSummaryList": [
    {
      "arn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/myexampleinfrastructure",
      "name": "MyExampleInfrastructure",
      "description": "An example that will retain instances of failed builds",
      "dateCreated": "2020-02-19T19:11:51.858Z",
      "tags": {}
    },
    {
      "arn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/infrastructureconfiguration-name-a1b2c3d45678",
      "name": "infrastructureConfiguration-name-a1b2c3d45678",
      "dateCreated": "2019-12-16T18:19:01.038Z",

```

```
        "tags": {
            "KeyName": "KeyValue"
        }
    ]
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [ListInfrastructureConfigurations](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags for a specific resource

The following `list-tags-for-resource` example lists all of the tags for a specific resource.

```
aws imagebuilder list-tags-for-resource \
  --resource-arn arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/
mywindows2016pipeline
```

Output:

```
{
  "tags": {
    "KeyName": "KeyValue"
  }
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

put-component-policy

The following code example shows how to use `put-component-policy`.

AWS CLI

To apply a resource policy to a component

The following `put-component-policy` command applies a resource policy to a build component to enable cross-account sharing of build components. We recommend you use the RAM CLI command `create-resource-share`. If you use the EC2 Image Builder CLI command `put-component-policy`, you must also use the RAM CLI command `promote-resource-share-create-from-policy` in order for the resource to be visible to all principals with whom the resource is shared.

```
aws imagebuilder put-component-policy \  
  --component-arn arn:aws:imagebuilder:us-west-2:123456789012:component/  
examplecomponent/2019.12.02/1 \  
  --policy '{ "Version": "2012-10-17", "Statement": [ { "Effect":  
"Allow", "Principal": { "AWS": [ "123456789012" ] }, "Action":  
[ "imagebuilder:GetComponent", "imagebuilder:ListComponents" ],  
"Resource": [ "arn:aws:imagebuilder:us-west-2:123456789012:component/  
examplecomponent/2019.12.02/1" ] } ] }'
```

Output:

```
{  
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
  "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/  
examplecomponent/2019.12.02/1"  
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [PutComponentPolicy](#) in *AWS CLI Command Reference*.

put-image-policy

The following code example shows how to use `put-image-policy`.

AWS CLI

To apply a resource policy to an image

The following `put-image-policy` command applies a resource policy to an image to enable cross-account sharing of images. We recommend you use the RAM CLI command `create-resource-share`. If you use the EC2 Image Builder CLI command `put-image-policy`, you must also use the RAM CLI command `promote-resource-share-create-from-policy` in order for the resource to be visible to all principals with whom the resource is shared.

```
aws imagebuilder put-image-policy \  
  --image-arn arn:aws:imagebuilder:us-west-2:123456789012:image/example-  
image/2019.12.02/1 \  
  --policy '{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow",  
"Principal": { "AWS": [ "123456789012" ] }, "Action": [ "imagebuilder:GetImage",  
"imagebuilder:ListImages" ], "Resource": [ "arn:aws:imagebuilder:us-  
west-2:123456789012:image/example-image/2019.12.02/1" ] } ] }'
```

Output:

```
{  
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
  "imageArn": "arn:aws:imagebuilder:us-west-2:123456789012:image/example-  
image/2019.12.02/1"  
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [PutImagePolicy](#) in *AWS CLI Command Reference*.

put-image-recipe-policy

The following code example shows how to use `put-image-recipe-policy`.

AWS CLI

To apply a resource policy to an image recipe

The following `put-image-recipe-policy` command applies a resource policy to an image recipe to enable cross-account sharing of image recipes. We recommend that you use the RAM CLI command `create-resource-share`. If you use the EC2 Image Builder CLI command `put-image-recipe-policy`, you must also use the RAM CLI command `promote-resource-share-create-from-policy` in order for the resource to be visible to all principals with whom the resource is shared.

```
aws imagebuilder put-image-recipe-policy \  
  --image-recipe-arn arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/  
example-image-recipe/2019.12.02 \  
  --policy '{ "Version": "2012-10-17", "Statement": [ { "Effect":  
"Allow", "Principal": { "AWS": [ "123456789012" ] }, "Action":  
[ "imagebuilder:GetImageRecipe", "imagebuilder:ListImageRecipes" ], "Resource":  
[ "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/example-image-  
recipe/2019.12.02" ] } ] }'
```

Output:

```
{  
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
  "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/  
example-image-recipe/2019.12.02/1"  
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [PutImageRecipePolicy](#) in *AWS CLI Command Reference*.

start-image-pipeline-execution

The following code example shows how to use `start-image-pipeline-execution`.

AWS CLI

To start an image pipeline manually

The following `start-image-pipeline-execution` example manually starts an image pipeline.

```
aws imagebuilder start-image-pipeline-execution \  
  --image-pipeline-arn arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/  
mywindows2016pipeline
```

Output:

```
{
```



```
"requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
"clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
"imageBuildVersionArn": "arn:aws:imagebuilder:us-west-2:123456789012:image/
mybasicrecipe/2019.12.03/1"
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [StartImagePipelineExecution](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To tag a resource

The following `tag-resource` example adds and tags a resource to EC2 Image Builder using a JSON file.

```
aws imagebuilder tag-resource \
  --cli-input-json file://tag-resource.json
```

Contents of `tag-resource.json`:

```
{
  "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/
mywindows2016pipeline",
  "tags": {
    "KeyName": "KeyValue"
  }
}
```

This command produces no output.

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag from a resource

The following `untag-resource` example removes a tag from a resource using a JSON file.

```
aws imagebuilder untag-resource \  
  --cli-input-json file://tag-resource.json
```

Contents of `untag-resource.json`:

```
{  
  "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/  
mywindows2016pipeline",  
  "tagKeys": [  
    "KeyName"  
  ]  
}
```

This command produces no output.

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-distribution-configuration

The following code example shows how to use `update-distribution-configuration`.

AWS CLI

To update a distribution configuration

The following `update-distribution-configuration` example updates a distribution configuration using a JSON file.

```
aws imagebuilder update-distribution-configuration \  
  --cli-input-json file://update-distribution-configuration.json
```

```
--cli-input-json file://update-distribution-configuration.json
```

Contents of `update-distribution-configuration.json`:

```
{
  "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/myexampledistribution",
  "description": "Copies AMI to eu-west-2 and exports to S3",
  "distributions": [
    {
      "region": "us-west-2",
      "amiDistributionConfiguration": {
        "name": "Name {{imagebuilder:buildDate}}",
        "description": "An example image name with parameter references"
      }
    },
    {
      "region": "eu-west-2",
      "amiDistributionConfiguration": {
        "name": "My {{imagebuilder:buildVersion}} image
{{imagebuilder:buildDate}}"
      }
    }
  ]
}
```

Output:

```
{
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [UpdateDistributionConfiguration](#) in *AWS CLI Command Reference*.

update-image-pipeline

The following code example shows how to use `update-image-pipeline`.

AWS CLI

To update an image pipeline

The following `update-image-pipeline` example updates an image pipeline using a JSON file.

```
aws imagebuilder update-image-pipeline \  
  --cli-input-json file://update-image-pipeline.json
```

Contents of `update-image-pipeline.json`:

```
{  
  "imagePipelineArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/  
mywindows2016pipeline",  
  "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/  
mybasicrecipe/2019.12.03",  
  "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-  
west-2:123456789012:infrastructure-configuration/myexampleinfrastructure",  
  "distributionConfigurationArn": "arn:aws:imagebuilder:us-  
west-2:123456789012:distribution-configuration/myexampledistribution",  
  "imageTestsConfiguration": {  
    "imageTestsEnabled": true,  
    "timeoutMinutes": 120  
  },  
  "schedule": {  
    "scheduleExpression": "cron(0 0 * * MON)",  
    "pipelineExecutionStartCondition":  
"EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"  
  },  
  "status": "DISABLED"  
}
```

Output:

```
{  
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"  
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [UpdateImagePipeline](#) in *AWS CLI Command Reference*.

update-infrastructure-configuration

The following code example shows how to use update-infrastructure-configuration.

AWS CLI

To update an infrastructure configuration

The following update-infrastructure-configuration example updates an infrastructure configuration using a JSON file.

```
aws imagebuilder update-infrastructure-configuration \  
  --cli-input-json file:/update-infrastructure-configuration.json
```

Contents of update-infrastructure-configuration.json:

```
{  
  "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-  
west-2:123456789012:infrastructure-configuration/myexampleinfrastructure",  
  "description": "An example that will terminate instances of failed builds",  
  "instanceTypes": [  
    "m5.large", "m5.2xlarge"  
  ],  
  "instanceProfileName": "EC2InstanceProfileForImageFactory",  
  "securityGroupIds": [  
    "sg-a48c95ef"  
  ],  
  "subnetId": "subnet-a48c95ef",  
  "logging": {  
    "s3Logs": {  
      "s3BucketName": "bucket-name",  
      "s3KeyPrefix": "bucket-path"  
    }  
  },  
  "terminateInstanceOnFailure": true,  
  "snsTopicArn": "arn:aws:sns:us-west-2:123456789012:sns-name"  
}
```

Output:

```
{  
  "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE111111"
```

```
}
```

For more information, see [Setting Up and Managing an EC2 Image Builder Image Pipeline Using the AWS CLI](#) in the *EC2 Image Builder Users Guide*.

- For API details, see [UpdateInfrastructureConfiguration](#) in *AWS CLI Command Reference*.

Incident Manager examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Incident Manager.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-replication-set

The following code example shows how to use `create-replication-set`.

AWS CLI

To create the replication set

The following `create-replication-set` example creates the replication set Incident Manager uses to replicate and encrypt data in your Amazon Web Services account. This example uses the `us-east-1` and `us-east-2` Regions while creating the replication set.

```
aws ssm-incidents create-replication-set \
```

```
--regions '{"us-east-1": {"sseKmsKeyId": "arn:aws:kms:us-east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"}, "us-east-2": {"sseKmsKeyId": "arn:aws:kms:us-east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"}}'
```

Output:

```
{
  "replicationSetArns": [
    "arn:aws:ssm-incidents::111122223333:replication-set/c4bcb603-4bf9-bb3f-413c-08df53673b57"
  ]
}
```

For more information, see [Using the Incident Manager replication set](#) in the *Incident Manager User Guide*.

- For API details, see [CreateReplicationSet](#) in *AWS CLI Command Reference*.

create-response-plan

The following code example shows how to use create-response-plan.

AWS CLI

To create a response plan

The following create-response-plan example creates a response plan with the specified details.

```
aws ssm-incidents create-response-plan \
  --chat-channel '{"chatbotSns": ["arn:aws:sns:us-east-1:111122223333:Standard_User"]}' \
  --display-name "Example response plan" \
  --incident-template '{"impact": 5, "title": "example-incident"}' \
  --name "example-response" \
  --actions '[{"ssmAutomation": {"documentName": "AWSIncidents-CriticalIncidentRunbookTemplate", "documentVersion": "$DEFAULT", "roleArn": "arn:aws:iam::111122223333:role/aws-service-role/ssm-incidents.amazonaws.com/AWSServiceRoleForIncidentManager", "targetAccount": "RESPONSE_PLAN_OWNER_ACCOUNT"}}]' \
  --engagements '["arn:aws:ssm-contacts:us-east-1:111122223333:contact/example"]'
```

Output:

```
{
  "arn": "arn:aws:ssm-incidents::111122223333:response-plan/example-response"
}
```

For more information, see [Incident preparation](#) in the *Incident Manager User Guide*.

- For API details, see [CreateResponsePlan](#) in *AWS CLI Command Reference*.

create-timeline-event

The following code example shows how to use `create-timeline-event`.

AWS CLI**Example 1: To create a custom timeline event**

The following `create-timeline-event` example creates a custom timeline event at the specified time on the specified incident.

```
aws ssm-incidents create-timeline-event \
  --event-data "\"example timeline event\"" \
  --event-time 2022-10-01T20:30:00.000 \
  --event-type "Custom Event" \
  --incident-record-arn "arn:aws:ssm-incidents::111122223333:incident-record/
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4EXAMPLE"
```

Output:

```
{
  "eventId": "c0bcc885-a41d-eb01-b4ab-9d2deEXAMPLE",
  "incidentRecordArn": "arn:aws:ssm-incidents::111122223333:incident-record/
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4EXAMPLE"
}
```

Example 2: To create a timeline event with an incident note

The following `create-timeline-event` example creates a timeline event that is listed in the 'Incident notes' panel.

```
aws ssm-incidents create-timeline-event \
```



```
--event-data "\"New Note\"" \
--event-type "Note" \
--incident-record-arn "arn:aws:ssm-incidents::111122223333:incident-record/
Test/6cc46130-ca6c-3b38-68f1-f6abeEXAMPLE" \
--event-time 2023-06-20T12:06:00.000 \
--event-references ' [{"resource": "arn:aws:ssm-incidents::111122223333:incident-
record/Test/6cc46130-ca6c-3b38-68f1-f6abeEXAMPLE"} ]'
```

Output:

```
{
  "eventId": "a41dc885-c0bc-b4ab-eb01-de9d2EXAMPLE",
  "incidentRecordArn": "arn:aws:ssm-incidents::111122223333:incident-record/
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4EXAMPLE"
}
```

For more information, see [Incident details](#) in the *Incident Manager User Guide*.

- For API details, see [CreateTimelineEvent](#) in *AWS CLI Command Reference*.

delete-incident-record

The following code example shows how to use `delete-incident-record`.

AWS CLI

To delete an incident record

The following `delete-incident-record` example deletes the specified incident record.

```
aws ssm-incidents delete-incident-record \
  --arn "arn:aws:ssm-incidents::111122223333:incident-record/Example-Response-
Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
```

This command produces no output.

For more information, see [Incident tracking](#) in the *Incident Manager User Guide*.

- For API details, see [DeleteIncidentRecord](#) in *AWS CLI Command Reference*.

delete-replication-set

The following code example shows how to use `delete-replication-set`.

AWS CLI

To delete the replication set

The following `delete-replication-set` example deletes the replication set from your Amazon Web Services account. Deleting the replication set also deletes all Incident Manager data. This can't be undone.

```
aws ssm-incidents delete-replication-set \  
  --arn "arn:aws:ssm-incidents::111122223333:replication-set/c4bcb603-4bf9-  
bb3f-413c-08df53673b57"
```

This command produces no output.

For more information, see [Using the Incident Manager replication set](#) in the *Incident Manager User Guide*.

- For API details, see [DeleteReplicationSet](#) in *AWS CLI Command Reference*.

`delete-resource-policy`

The following code example shows how to use `delete-resource-policy`.

AWS CLI

To delete a resource policy

The following `delete-resource-policy` example deletes a resource policy from a response plan. This will revoke access from the principal or organization that the response plan was shared with.

```
aws ssm-incidents delete-resource-policy \  
  --policy-id "be8b57191f0371f1c6827341aa3f0a03" \  
  --resource-arn "arn:aws:ssm-incidents::111122223333:response-plan/Example-  
Response-Plan"
```

This command produces no output.

For more information, see [Working with shared contacts and response plans](#) in the *Incident Manager User Guide*.

- For API details, see [DeleteResourcePolicy](#) in *AWS CLI Command Reference*.

delete-response-plan

The following code example shows how to use `delete-response-plan`.

AWS CLI

To delete a response plan

The following `delete-response-plan` example deletes the specified response plan.

```
aws ssm-incidents delete-response-plan \  
  --arn "arn:aws:ssm-incidents::111122223333:response-plan/example-response"
```

This command produces no output.

For more information, see [Incident preparation](#) in the *Incident Manager User Guide*.

- For API details, see [DeleteResponsePlan](#) in *AWS CLI Command Reference*.

delete-timeline-event

The following code example shows how to use `delete-timeline-event`.

AWS CLI

To delete a timeline event

The following `delete-timeline-event` example deletes a custom timeline event from the specified incident record.

```
aws ssm-incidents delete-timeline-event \  
  --event-id "c0bcc885-a41d-eb01-b4ab-9d2de193643c" \  
  --incident-record-arn "arn:aws:ssm-incidents::111122223333:incident-record/  
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
```

This command produces no output.

For more information, see [Incident details](#) in the *Incident Manager User Guide*.

- For API details, see [DeleteTimelineEvent](#) in *AWS CLI Command Reference*.

get-incident-record

The following code example shows how to use `get-incident-record`.

AWS CLI

To get an incident record

The following `get-incident-record` example gets details about the specified incident record.

```
aws ssm-incidents get-incident-record \  
  --arn "arn:aws:ssm-incidents::111122223333:incident-record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
```

Output:

```
{  
  "incidentRecord": {  
    "arn": "arn:aws:ssm-incidents::111122223333:incident-record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308",  
    "automationExecutions": [],  
    "creationTime": "2021-05-21T18:16:57.579000+00:00",  
    "dedupeString": "c4bcc812-85e7-938d-2b78-17181176ee1a",  
    "impact": 5,  
    "incidentRecordSource": {  
      "createdBy": "arn:aws:iam::111122223333:user/draliatp",  
      "invokedBy": "arn:aws:iam::111122223333:user/draliatp",  
      "source": "aws.ssm-incidents.custom"  
    },  
    "lastModifiedBy": "arn:aws:iam::111122223333:user/draliatp",  
    "lastModifiedTime": "2021-05-21T18:16:59.149000+00:00",  
    "notificationTargets": [],  
    "status": "OPEN",  
    "title": "Example-Incident"  
  }  
}
```

For more information, see [Incident details](#) in the *Incident Manager User Guide*.

- For API details, see [GetIncidentRecord](#) in *AWS CLI Command Reference*.

get-replication-set

The following code example shows how to use `get-replication-set`.

AWS CLI

To get the replication set

The following `get-replication-set` example gets the details of the replication set Incident Manager uses to replicate and encrypt data in your Amazon Web Services account.

```
aws ssm-incidents get-replication-set \  
  --arn "arn:aws:ssm-incidents::111122223333:replication-set/c4bcb603-4bf9-  
bb3f-413c-08df53673b57"
```

Output:

```
{  
  "replicationSet": {  
    "createdBy": "arn:aws:sts::111122223333:assumed-role/Admin/username",  
    "createdTime": "2021-05-14T17:57:22.010000+00:00",  
    "deletionProtected": false,  
    "lastModifiedBy": "arn:aws:sts::111122223333:assumed-role/Admin/username",  
    "lastModifiedTime": "2021-05-14T17:57:22.010000+00:00",  
    "regionMap": {  
      "us-east-1": {  
        "sseKmsKeyId": "DefaultKey",  
        "status": "ACTIVE"  
      },  
      "us-east-2": {  
        "sseKmsKeyId": "DefaultKey",  
        "status": "ACTIVE",  
        "statusMessage": "Tagging inaccessible"  
      }  
    },  
    "status": "ACTIVE"  
  }  
}
```

For more information, see [Using the Incident Manager replication set](#) in the *Incident Manager User Guide*.

- For API details, see [GetReplicationSet](#) in *AWS CLI Command Reference*.

get-resource-policies

The following code example shows how to use `get-resource-policies`.

AWS CLI

To list resource policies for a response plan

The following command-name example lists the resource policies associated with the specified response plan.

```
aws ssm-incidents get-resource-policies \
--resource-arn "arn:aws:ssm-incidents::111122223333:response-plan/Example-Response-Plan"
```

Output:

```
{
  "resourcePolicies": [
    {
      "policyDocument": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Sid\":\"d901b37a-dbb0-458a-8842-75575c464219-external-principals\",\"Effect\":\"Allow\",\"Principal\":{\"AWS\":\"arn:aws:iam::222233334444:root\"},\"Action\":[\"ssm-incidents:GetResponsePlan\",\"ssm-incidents:StartIncident\",\"ssm-incidents:UpdateIncidentRecord\",\"ssm-incidents:GetIncidentRecord\",\"ssm-incidents:CreateTimelineEvent\",\"ssm-incidents:UpdateTimelineEvent\",\"ssm-incidents:GetTimelineEvent\",\"ssm-incidents:ListTimelineEvents\",\"ssm-incidents:UpdateRelatedItems\",\"ssm-incidents:ListRelatedItems\"]},\"Resource\":[\"arn:aws:ssm-incidents:*:111122223333:response-plan/Example-Response-Plan\",\"arn:aws:ssm-incidents:*:111122223333:incident-record/Example-Response-Plan/*\"]}]}",
      "policyId": "be8b57191f0371f1c6827341aa3f0a03",
      "ramResourceShareRegion": "us-east-1"
    }
  ]
}
```

For more information, see [Working with shared contacts and response plans](#) in the *Incident Manager User Guide*.

- For API details, see [GetResourcePolicies](#) in *AWS CLI Command Reference*.

get-response-plan

The following code example shows how to use `get-response-plan`.

AWS CLI

To get details of a response plan

The following command-name example gets details about a specified response plan in your AWS account.

```
aws ssm-incidents get-response-plan \
  --arn "arn:aws:ssm-incidents::111122223333:response-plan/Example-Response-Plan"
```

Output:

```
{
  "actions": [
    {
      "ssmAutomation": {
        "documentName": "AWSIncidents-CriticalIncidentRunbookTemplate",
        "documentVersion": "$DEFAULT",
        "roleArn": "arn:aws:iam::111122223333:role/aws-service-role/ssm-
incidents.amazonaws.com/AWSServiceRoleForIncidentManager",
        "targetAccount": "RESPONSE_PLAN_OWNER_ACCOUNT"
      }
    }
  ],
  "arn": "arn:aws:ssm-incidents::111122223333:response-plan/Example-Response-
Plan",
  "chatChannel": {
    "chatbotSns": [
      "arn:aws:sns:us-east-1:111122223333:Standard_User"
    ]
  },
  "displayName": "Example response plan",
  "engagements": [
    "arn:aws:ssm-contacts:us-east-1:111122223333:contact/example"
  ],
  "incidentTemplate": {
    "impact": 5,
    "title": "Example-Incident"
  },
}
```

```
"name": "Example-Response-Plan"
}
```

For more information, see [Incident preparation](#) in the *Incident Manager User Guide*.

- For API details, see [GetResponsePlan](#) in *AWS CLI Command Reference*.

get-timeline-event

The following code example shows how to use `get-timeline-event`.

AWS CLI

To get details of a timeline event

The following `get-timeline-event` example returns details of the specified timeline event.

```
aws ssm-incidents get-timeline-event \
  --event-id 20bcc812-8a94-4cd7-520c-0ff742111424 \
  --incident-record-arn "arn:aws:ssm-incidents::111122223333:incident-record/
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
```

Output:

```
{
  "event": {
    "eventData": "\"Incident Started\"",
    "eventId": "20bcc812-8a94-4cd7-520c-0ff742111424",
    "eventTime": "2021-05-21T18:16:57+00:00",
    "eventType": "Custom Event",
    "eventUpdatedTime": "2021-05-21T18:16:59.944000+00:00",
    "incidentRecordArn": "arn:aws:ssm-incidents::111122223333:incident-record/
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
  }
}
```

For more information, see [Incident details](#) in the *Incident Manager User Guide*.

- For API details, see [GetTimelineEvent](#) in *AWS CLI Command Reference*.

list-incident-records

The following code example shows how to use `list-incident-records`.

AWS CLI

To list incident records

The following command-name example lists the incident records in your Amazon Web Services account.

```
aws ssm-incidents list-incident-records
```

Output:

```
{
  "incidentRecordSummaries": [
    {
      "arn": "arn:aws:ssm-incidents::111122223333:incident-record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308",
      "creationTime": "2021-05-21T18:16:57.579000+00:00",
      "impact": 5,
      "incidentRecordSource": {
        "createdBy": "arn:aws:iam::111122223333:user/draliatp",
        "invokedBy": "arn:aws:iam::111122223333:user/draliatp",
        "source": "aws.ssm-incidents.custom"
      },
      "status": "OPEN",
      "title": "Example-Incident"
    }
  ]
}
```

For more information, see [Incident list](#) in the *Incident Manager User Guide*.

- For API details, see [ListIncidentRecords](#) in *AWS CLI Command Reference*.

list-related-items

The following code example shows how to use `list-related-items`.

AWS CLI

To list related items

The following `list-related-items` example lists the related items of the specified incident.

```
aws ssm-incidents list-related-items \
  --incident-record-arn "arn:aws:ssm-incidents::111122223333:incident-record/
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
```

Output:

```
{
  "relatedItems": [
    {
      "identifier": {
        "type": "OTHER",
        "value": {
          "url": "https://console.aws.amazon.com/systems-manager/opsitems/
oi-8ef82158e190/workbench?region=us-east-1"
        }
      },
      "title": "Example related item"
    },
    {
      "identifier": {
        "type": "PARENT",
        "value": {
          "arn": "arn:aws:ssm:us-east-1:111122223333:opsitem/
oi-8084126392ac"
        }
      },
      "title": "parentItem"
    }
  ]
}
```

For more information, see [Incident details](#) in the *Incident Manager User Guide*.

- For API details, see [ListRelatedItems](#) in *AWS CLI Command Reference*.

list-replication-sets

The following code example shows how to use `list-replication-sets`.

AWS CLI

To list the replication set

The following `list-replication-set` example lists the replication set Incident Manager uses to replicate and encrypt data in your AWS account.

```
aws ssm-incidents list-replication-sets
```

Output:

```
{
  "replicationSetArns": [
    "arn:aws:ssm-incidents::111122223333:replication-set/c4bcb603-4bf9-
    bb3f-413c-08df53673b57"
  ]
}
```

For more information, see [Using the Incident Manager replication set](#) in the *Incident Manager User Guide*.

- For API details, see [ListReplicationSets](#) in *AWS CLI Command Reference*.

list-response-plans

The following code example shows how to use `list-response-plans`.

AWS CLI

To list the available response plans

The following `list-response-plans` example lists the available response plans in your Amazon Web Services account.

```
aws ssm-incidents list-response-plans
```

Output:

```
{
  "responsePlanSummaries": [
    {
      "arn": "arn:aws:ssm-incidents::111122223333:response-plan/Example-
      Response-Plan",
      "displayName": "Example response plan",
      "name": "Example-Response-Plan"
    }
  ]
}
```

```
    }  
  ]  
}
```

For more information, see [Incident preparation](#) in the *Incident Manager User Guide*.

- For API details, see [ListResponsePlans](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags for a response plan

The following `list-tags-for-resource` example lists the tags associated with the specified response plan.

```
aws ssm-incidents list-tags-for-resource \  
    --resource-arn "arn:aws:ssm-incidents::111122223333:response-plan/Example-  
Response-Plan"
```

Output:

```
{  
  "tags": {  
    "group1": "1"  
  }  
}
```

For more information, see [Tagging](#) in the *Incident Manager User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

list-timeline-events

The following code example shows how to use `list-timeline-events`.

AWS CLI

To list timeline events of an incident

The following command-name example lists the timeline events of the specified incident.

```
aws ssm-incidents list-timeline-events \  
  --incident-record-arn "arn:aws:ssm-incidents::111122223333:incident-record/  
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
```

Output:

```
{  
  "eventSummaries": [  
    {  
      "eventId": "8cbcc889-35e1-a42d-2429-d6f100799915",  
      "eventTime": "2021-05-21T22:36:13.766000+00:00",  
      "eventType": "SSM Incident Record Update",  
      "eventUpdatedTime": "2021-05-21T22:36:13.766000+00:00",  
      "incidentRecordArn": "arn:aws:ssm-incidents::111122223333:incident-  
record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"  
    },  
    {  
      "eventId": "a2bcc825-aab5-1787-c605-f9bb2640d85b",  
      "eventTime": "2021-05-21T18:58:46.443000+00:00",  
      "eventType": "SSM Incident Record Update",  
      "eventUpdatedTime": "2021-05-21T18:58:46.443000+00:00",  
      "incidentRecordArn": "arn:aws:ssm-incidents::111122223333:incident-  
record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"  
    },  
    {  
      "eventId": "5abcc812-89c0-b0a8-9437-1c74223d4685",  
      "eventTime": "2021-05-21T18:16:59.149000+00:00",  
      "eventType": "SSM Incident Record Update",  
      "eventUpdatedTime": "2021-05-21T18:16:59.149000+00:00",  
      "incidentRecordArn": "arn:aws:ssm-incidents::111122223333:incident-  
record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"  
    },  
    {  
      "eventId": "06bcc812-8820-405e-4065-8d2b14d29b92",  
      "eventTime": "2021-05-21T18:16:58+00:00",  
      "eventType": "SSM Automation Execution Start Failure for Incident",  
      "eventUpdatedTime": "2021-05-21T18:16:58.689000+00:00",  
      "incidentRecordArn": "arn:aws:ssm-incidents::111122223333:incident-  
record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"  
    },  
    {
```

```

    "eventId": "20bcc812-8a94-4cd7-520c-0ff742111424",
    "eventTime": "2021-05-21T18:16:57+00:00",
    "eventType": "Custom Event",
    "eventUpdatedTime": "2021-05-21T18:16:59.944000+00:00",
    "incidentRecordArn": "arn:aws:ssm-incidents::111122223333:incident-
record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
  },
  {
    "eventId": "c0bcc885-a41d-eb01-b4ab-9d2de193643c",
    "eventTime": "2020-10-01T20:30:00+00:00",
    "eventType": "Custom Event",
    "eventUpdatedTime": "2021-05-21T22:28:26.299000+00:00",
    "incidentRecordArn": "arn:aws:ssm-incidents::111122223333:incident-
record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"
  }
]
}

```

For more information, see [Incident details](#) in the *Incident Manager User Guide*.

- For API details, see [ListTimelineEvents](#) in *AWS CLI Command Reference*.

put-resource-policy

The following code example shows how to use `put-resource-policy`.

AWS CLI

To share a response plan and incidents

The following command-name example adds a resource policy to the `Example-Response-Plan` that shares the response plan and associated incidents with the specified principal.

```

aws ssm-incidents put-resource-policy \
  --resource-arn "arn:aws:ssm-incidents::111122223333:response-plan/Example-
Response-Plan" \
  --policy "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Sid\":
\"ExampleResourcePolciy\",\"Effect\":\"Allow\",\"Principal\":{\"AWS\":
\"arn:aws:iam::222233334444:root\"},\"Action\":[\"ssm-incidents:GetResponsePlan
\",\"ssm-incidents:StartIncident\",\"ssm-incidents:UpdateIncidentRecord
\",\"ssm-incidents:GetIncidentRecord\",\"ssm-incidents:CreateTimelineEvent
\",\"ssm-incidents:UpdateTimelineEvent\",\"ssm-incidents:GetTimelineEvent
\",\"ssm-incidents:ListTimelineEvents\",\"ssm-incidents:UpdateRelatedItems

```

```
\",\"ssm-incidents:ListRelatedItems\"],\"Resource\":[\"arn:aws:ssm-incidents:*:111122223333:response-plan/Example-Response-Plan\", \"arn:aws:ssm-incidents:*:111122223333:incident-record/Example-Response-Plan/*\"]}]}"
```

Output:

```
{  
  "policyId": "be8b57191f0371f1c6827341aa3f0a03"  
}
```

For more information, see [Working with shared contacts and response plans](#) in the *Incident Manager User Guide*.

- For API details, see [PutResourcePolicy](#) in *AWS CLI Command Reference*.

start-incident

The following code example shows how to use `start-incident`.

AWS CLI

To start an incident

The following `start-incident` example starts an incident using the specified response plan.

```
aws ssm-incidents start-incident \  
  --response-plan-arn "arn:aws:ssm-incidents::111122223333:response-plan/Example-Response-Plan"
```

Output:

```
{  
  "incidentRecordArn": "arn:aws:ssm-incidents::682428703967:incident-record/Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308"  
}
```

For more information, see [Incident creation](#) in the *Incident Manager User Guide*.

- For API details, see [StartIncident](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To tag a response plan

The following `tag-resource` example tags a specified response plan with the provided tag key-value pair.

```
aws ssm-incidents tag-resource \  
  --resource-arn "arn:aws:ssm-incidents::111122223333:response-plan/Example-  
Response-Plan" \  
  --tags '{"group1":"1"}'
```

This command produces no output.

For more information, see [Tagging](#) in the *Incident Manager User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from a response plan

The following `untag-resource` example removes the specified tags from the response plan.

```
aws ssm-incidents untag-resource \  
  --resource-arn "arn:aws:ssm-incidents::111122223333:response-plan/Example-  
Response-Plan" \  
  --tag-keys '["group1"]'
```

This command produces no output.

For more information, see [Tagging](#) in the *Incident Manager User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-deletion-protection

The following code example shows how to use update-deletion-protection.

AWS CLI

To update replication set deletion protection

The following update-deletion-protection example updates the deletion protection in your account to protect you from deleting the last Region in your replication set.

```
aws ssm-incidents update-deletion-protection \  
  --arn "arn:aws:ssm-incidents::111122223333:replication-set/  
a2bcc5c9-0f53-8047-7fef-c20749989b40" \  
  --deletion-protected
```

This command produces no output.

For more information, see [Using the Incident Manager replication set](#) in the *Incident Manager User Guide*.

- For API details, see [UpdateDeletionProtection](#) in *AWS CLI Command Reference*.

update-incident-record

The following code example shows how to use update-incident-record.

AWS CLI

To update an incident record

The following command-name example resolves the specified incident.

```
aws ssm-incidents update-incident-record \  
  --arn "arn:aws:ssm-incidents::111122223333:incident-record/Example-Response-  
Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308" \  
  --status "RESOLVED"
```

This command produces no output.

For more information, see [Incident details](#) in the *Incident Manager User Guide*.

- For API details, see [UpdateIncidentRecord](#) in *AWS CLI Command Reference*.

update-related-items

The following code example shows how to use `update-related-items`.

AWS CLI

To update an incidents related item

The following `update-related-item` example removes a related item from the specified incident record.

```
aws ssm-incidents update-related-items \  
  --incident-record-arn "arn:aws:ssm-incidents::111122223333:incident-record/  
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308" \  
  --related-items-update '{"itemToRemove": {"type": "OTHER", "value": {"url":  
"https://console.aws.amazon.com/systems-manager/opsitems/oi-8ef82158e190/workbench?  
region=us-east-1"}}}'
```

This command produces no output.

For more information, see [Incident details](#) in the *Incident Manager User Guide*.

- For API details, see [UpdateRelatedItems](#) in *AWS CLI Command Reference*.

update-replication-set

The following code example shows how to use `update-replication-set`.

AWS CLI

To update a replication set

The following `command-name` example deletes the `us-east-2` Region from the replication set.

```
aws ssm-incidents update-replication-set \  
  --arn "arn:aws:ssm-incidents::111122223333:replication-set/  
a2bcc5c9-0f53-8047-7fef-c20749989b40" \  
  --actions '[{"deleteRegionAction": {"regionName": "us-east-2"}}]'
```

This command produces no output.

For more information, see [Using the Incident Manager replication set](#) in the *Incident Manager User Guide*.

- For API details, see [UpdateReplicationSet](#) in *AWS CLI Command Reference*.

update-response-plan

The following code example shows how to use `update-response-plan`.

AWS CLI

To update a response plan

The following `update-response-plan` example removes a chat channel from the specified response plan.

```
aws ssm-incidents update-response-plan \  
  --arn "arn:aws:ssm-incidents::111122223333:response-plan/Example-Response-Plan" \  
  \  
  --chat-channel '{"empty":{}}'
```

This command produces no output.

For more information, see [Incident preparation](#) in the *Incident Manager User Guide*.

- For API details, see [UpdateResponsePlan](#) in *AWS CLI Command Reference*.

update-timeline-event

The following code example shows how to use `update-timeline-event`.

AWS CLI

To update a timeline event

The following `update-timeline-event` example updates the time that the event occurred.

```
aws ssm-incidents update-timeline-event \  
  --event-id 20bcc812-8a94-4cd7-520c-0ff742111424 \  
  --incident-record-arn "arn:aws:ssm-incidents::111122223333:incident-record/  
Example-Response-Plan/6ebcc812-85f5-b7eb-8b2f-283e4d844308" \  
  --event-time "2021-05-21T18:10:57+00:00"
```

This command produces no output.

For more information, see [Incident details](#) in the *Incident Manager User Guide*.

- For API details, see [UpdateTimelineEvent](#) in *AWS CLI Command Reference*.

Incident Manager Contacts examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Incident Manager Contacts.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

accept-page

The following code example shows how to use `accept-page`.

AWS CLI

To accept a page during and engagement

The following `accept-page` example uses an accept code sent to the contact channel to accept a page.

```
aws ssm-contacts accept-page \  
  --page-id "arn:aws:ssm-contacts:us-east-2:682428703967:page/  
akuam/94ea0c7b-56d9-46c3-b84a-a37c8b067ad3" \  
  --accept-type READ \  
  --accept-code 425440
```

This command produces no output.

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [AcceptPage](#) in *AWS CLI Command Reference*.

activate-contact-channel

The following code example shows how to use `activate-contact-channel`.

AWS CLI

Activate a contact's contact channel

The following `activate-contact-channel` example activates a contact channel and makes it usable as part of an incident.

```
aws ssm-contacts activate-contact-channel \
  --contact-channel-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact-
channel/akuam/fc7405c4-46b2-48b7-87b2-93e2f225b90d" \
  --activation-code "466136"
```

This command produces no output.

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [ActivateContactChannel](#) in *AWS CLI Command Reference*.

command-name

The following code example shows how to use `command-name`.

AWS CLI

To delete a contact

The following `command-name` example deletes a contact. The contact will no longer be reachable from any escalation plan that refers to them.

```
aws ssm-contacts delete-contact \
  --contact-id "arn:aws:ssm-contacts:us-east-1:682428703967:contact/alejrr"
```

This command produces no output.

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [CommandName](#) in *AWS CLI Command Reference*.

create-contact-channel

The following code example shows how to use `create-contact-channel`.

AWS CLI

To create a contact channel

Creates a contact channel of type SMS for the contact Akua Mansa. Contact channels can be created of type SMS, EMAIL, or VOICE.

```
aws ssm-contacts create-contact-channel \
  --contact-id "arn:aws:ssm-contacts:us-east-1:111122223333:contact/akuam" \
  --name "akuas sms-test" \
  --type SMS \
  --delivery-address '{"SimpleAddress": "+15005550199"}'
```

Output:

```
{
  "ContactChannelArn": "arn:aws:ssm-contacts:us-east-1:111122223333:contact-
channel/akuam/02f506b9-ea5d-4764-af89-2daa793ff024"
}
```

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [CreateContactChannel](#) in *AWS CLI Command Reference*.

create-contact

The following code example shows how to use `create-contact`.

AWS CLI

To create a contact

The following `create-contact` example creates a contact in your environment with a blank plan. The plan can be updated after creating contact channels. Use the `create-contact-channel` command with the output ARN of this command. After you have created contact channels for this contact use `update-contact` to update the plan.

```
aws ssm-contacts create-contact \  
  --alias "akuam" \  
  --display-name "Akua Mansa" \  
  --type PERSONAL \  
  --plan '{"Stages": []}'
```

Output:

```
{  
  "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/akuam"  
}
```

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [CreateContact](#) in *AWS CLI Command Reference*.

deactivate-contact-channel

The following code example shows how to use `deactivate-contact-channel`.

AWS CLI

To deactivate a contact channel

The following `deactivate-contact-channel` example deactivates a contact channel. Deactivating a contact channel means the contact channel will no longer be paged during an incident. You can also reactivate a contact channel at any time using the `activate-contact-channel` command.

```
aws ssm-contacts deactivate-contact-channel \  
  --contact-channel-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact-  
channel/akuam/fc7405c4-46b2-48b7-87b2-93e2f225b90d"
```

This command produces no output.

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [DeactivateContactChannel](#) in *AWS CLI Command Reference*.

delete-contact-channel

The following code example shows how to use `delete-contact-channel`.

AWS CLI

To delete a contact channel

The following `delete-contact-channel` example deletes a contact channel. Deleting a contact channel ensures the contact channel will not be paged during an incident.

```
aws ssm-contacts delete-contact-channel \  
  --contact-channel-id "arn:aws:ssm-contacts:us-east-1:111122223333:contact-  
channel/akuam/13149bad-52ee-45ea-ae1e-45857f78f9b2"
```

This command produces no output.

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [DeleteContactChannel](#) in *AWS CLI Command Reference*.

delete-contact

The following code example shows how to use `delete-contact`.

AWS CLI

To delete a contact

The following `delete-contact` example deletes a contact. The contact will no longer be reachable from any escalation plan that refers to them.

```
aws ssm-contacts delete-contact \  
  --contact-id "arn:aws:ssm-contacts:us-east-1:111122223333:contact/alejrr"
```

This command produces no output.

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [DeleteContact](#) in *AWS CLI Command Reference*.

describe-engagement

The following code example shows how to use describe-engagement.

AWS CLI

To describe the details of an engagement

The following describe-engagement example lists the details of an engagement to a contact or escalation plan. The subject and content are sent to the contact channels.

```
aws ssm-contacts describe-engagement \  
  --engagement-id "arn:aws:ssm-contacts:us-east-2:111122223333:engagement/  
example_escalation/69e40ce1-8dbb-4d57-8962-5fbe7fc53356"
```

Output:

```
{  
  "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/  
example_escalation",  
  "EngagementArn": "arn:aws:ssm-contacts:us-east-2:111122223333:engagement/  
example_escalation/69e40ce1-8dbb-4d57-8962-5fbe7fc53356",  
  "Sender": "cli",  
  "Subject": "cli-test",  
  "Content": "Testing engagements via CLI",  
  "PublicSubject": "cli-test",  
  "PublicContent": "Testing engagements va CLI",  
  "StartTime": "2021-05-18T18:25:41.151000+00:00"  
}
```

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [DescribeEngagement](#) in *AWS CLI Command Reference*.

describe-page

The following code example shows how to use describe-page.

AWS CLI

To list the details of a page to a contact channel

The following describe-page example lists details of a page to a contact channel. The page will include the subject and content provided.

```
aws ssm-contacts describe-page \  
  --page-id "arn:aws:ssm-contacts:us-east-2:111122223333:page/akuam/ad0052bd-  
e606-498a-861b-25726292eb93"
```

Output:

```
{  
  "PageArn": "arn:aws:ssm-contacts:us-east-2:111122223333:page/akuam/ad0052bd-  
e606-498a-861b-25726292eb93",  
  "EngagementArn": "arn:aws:ssm-contacts:us-east-2:111122223333:engagement/  
akuam/78a29753-3674-4ac5-9f83-0468563567f0",  
  "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/akuam",  
  "Sender": "cli",  
  "Subject": "cli-test",  
  "Content": "Testing engagements via CLI",  
  "PublicSubject": "cli-test",  
  "PublicContent": "Testing engagements va CLI",  
  "SentTime": "2021-05-18T18:43:29.301000+00:00",  
  "ReadTime": "2021-05-18T18:43:55.708000+00:00",  
  "DeliveryTime": "2021-05-18T18:43:55.265000+00:00"  
}
```

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [DescribePage](#) in *AWS CLI Command Reference*.

get-contact-channel

The following code example shows how to use get-contact-channel.

AWS CLI

To list the details of a contact channel

The following get-contact-channel example lists the details of a contact channel.

```
aws ssm-contacts get-contact-channel \  
  --contact-channel-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact-  
channel/akuam/fc7405c4-46b2-48b7-87b2-93e2f225b90d"
```

Output:

```
{
  "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/akuam",
  "ContactChannelArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact-channel/akuam/fc7405c4-46b2-48b7-87b2-93e2f225b90d",
  "Name": "akuas sms",
  "Type": "SMS",
  "DeliveryAddress": {
    "SimpleAddress": "+15005550199"
  },
  "ActivationStatus": "ACTIVATED"
}
```

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [GetContactChannel](#) in *AWS CLI Command Reference*.

get-contact-policy

The following code example shows how to use `get-contact-policy`.

AWS CLI**To list the resource policies of a contact**

The following `get-contact-policy` example lists the resource policies associated with the specified contact.

```
aws ssm-contacts get-contact-policy \
  --contact-arn "arn:aws:ssm-contacts:us-east-1:111122223333:contact/akuam"
```

Output:

```
{
  "ContactArn": "arn:aws:ssm-contacts:us-east-1:111122223333:contact/akuam",
  "Policy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Sid\":\"SharePolicyForDocumentationDrAlia\",\"Effect\":\"Allow\",\"Principal\":{\"AWS\":\"222233334444\"},\"Action\":[\"ssm-contacts:GetContact\",\"ssm-contacts:StartEngagement\",\"ssm-contacts:DescribeEngagement\",\"ssm-contacts:ListPagesByEngagement\",\"ssm-contacts:StopEngagement\"],\"Resource\":[\"arn:aws:ssm-contacts:*:111122223333:contact/akuam\",\"arn:aws:ssm-contacts:*:111122223333:engagement/akuam/*\"]}]}"
```

```
}
```

For more information, see [Working with shared contacts and response plans](#) in the *Incident Manager User Guide*.

- For API details, see [GetContactPolicy](#) in *AWS CLI Command Reference*.

get-contact

The following code example shows how to use `get-contact`.

AWS CLI

Example 1: To describe a contact plan

The following `get-contact` example describes a contact.

```
aws ssm-contacts get-contact \
  --contact-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact/akuam"
```

Output:

```
{
  "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/akuam",
  "Alias": "akuam",
  "DisplayName": "Akua Mansa",
  "Type": "PERSONAL",
  "Plan": {
    "Stages": [
      {
        "DurationInMinutes": 5,
        "Targets": [
          {
            "ChannelTargetInfo": {
              "ContactChannelId": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact-channel/akuam/beb25840-5ac8-4644-95cc-7a8de390fa65",
              "RetryIntervalInMinutes": 1
            }
          }
        ]
      }
    ],
    "DurationInMinutes": 5,
```

```

        "Targets": [
            {
                "ChannelTargetInfo": {
                    "ContactChannelId": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact-channel/akuam/49f3c24d-5f9f-4638-ae25-3f49e04229ad",
                    "RetryIntervalInMinutes": 1
                }
            }
        ],
        {
            "DurationInMinutes": 5,
            "Targets": [
                {
                    "ChannelTargetInfo": {
                        "ContactChannelId": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact-channel/akuam/77d4f447-f619-4954-afff-85551e369c2a",
                        "RetryIntervalInMinutes": 1
                    }
                }
            ]
        }
    ]
}

```

Example 2: To describe an escalation plan

The following `get-contact` example describes an escalation plan.

```

aws ssm-contacts get-contact \
--contact-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
example_escalation"

```

Output:

```

{
    "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
example_escalation",
    "Alias": "example_escalation",
    "DisplayName": "Example Escalation",
    "Type": "ESCALATION",
    "Plan": {

```

```
    "Stages": [
      {
        "DurationInMinutes": 5,
        "Targets": [
          {
            "ContactTargetInfo": {
              "ContactId": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact/akuam",
              "IsEssential": true
            }
          }
        ]
      },
      {
        "DurationInMinutes": 5,
        "Targets": [
          {
            "ContactTargetInfo": {
              "ContactId": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact/alejr",
              "IsEssential": false
            }
          }
        ]
      },
      {
        "DurationInMinutes": 0,
        "Targets": [
          {
            "ContactTargetInfo": {
              "ContactId": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact/anasi",
              "IsEssential": false
            }
          }
        ]
      }
    ]
  }
}
```

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [GetContact](#) in *AWS CLI Command Reference*.

list-contact-channels

The following code example shows how to use `list-contact-channels`.

AWS CLI

To list the contact channels of a contact

The following `list-contact-channels` example lists the available contact channels of the specified contact.

```
aws ssm-contacts list-contact-channels \
  --contact-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact/akuam"
```

Output:

```
{
  [
    {
      "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
akuam",
      "Name": "akuas email",
      "Type": "EMAIL",
      "DeliveryAddress": {
        "SimpleAddress": "akuam@example.com"
      },
      "ActivationStatus": "NOT_ACTIVATED"
    },
    {
      "ContactChannelArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact-channel/akuam/fc7405c4-46b2-48b7-87b2-93e2f225b90d",
      "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
akuam",
      "Name": "akuas sms",
      "Type": "SMS",
      "DeliveryAddress": {
        "SimpleAddress": "+15005550100"
      },
      "ActivationStatus": "ACTIVATED"
    }
  ]
}
```

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [ListContactChannels](#) in *AWS CLI Command Reference*.

list-contacts

The following code example shows how to use `list-contacts`.

AWS CLI

To list all escalation plans and contacts

The following `list-contacts` example lists the contacts and escalation plans in your account.

```
aws ssm-contacts list-contacts
```

Output:

```
{
  "Contacts": [
    {
      "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
akuam",
      "Alias": "akuam",
      "DisplayName": "Akua Mansa",
      "Type": "PERSONAL"
    },
    {
      "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
alejr",
      "Alias": "alejr",
      "DisplayName": "Alejandro Rosalez",
      "Type": "PERSONAL"
    },
    {
      "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
anasi",
      "Alias": "anasi",
      "DisplayName": "Ana Carolina Silva",
      "Type": "PERSONAL"
    },
    {
```



```

        "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
example_escalation",
        "Alias": "example_escalation",
        "DisplayName": "Example Escalation",
        "Type": "ESCALATION"
    }
]
}

```

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [ListContacts](#) in *AWS CLI Command Reference*.

list-engagements

The following code example shows how to use `list-engagements`.

AWS CLI

To list all engagements

The following `list-engagements` example lists engagements to escalation plans and contacts. You can also list engagements for a single incident.

```
aws ssm-contacts list-engagements
```

Output:

```

{
  "Engagements": [
    {
      "EngagementArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:engagement/akuam/91792571-0b53-4821-9f73-d25d13d9e529",
      "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
akuam",
      "Sender": "cli",
      "StartTime": "2021-05-18T20:37:50.300000+00:00"
    },
    {
      "EngagementArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:engagement/akuam/78a29753-3674-4ac5-9f83-0468563567f0",
      "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
akuam",

```

```

        "Sender": "cli",
        "StartTime": "2021-05-18T18:40:26.666000+00:00"
    },
    {
        "EngagementArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:engagement/
example_escalation/69e40ce1-8dbb-4d57-8962-5fbe7fc53356",
        "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
example_escalation",
        "Sender": "cli",
        "StartTime": "2021-05-18T18:25:41.151000+00:00"
    },
    {
        "EngagementArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:engagement/akuam/607ced0e-e8fa-4ea7-8958-a237b8803f8f",
        "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
akuam",
        "Sender": "cli",
        "StartTime": "2021-05-18T18:20:58.093000+00:00"
    }
]
}

```

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [ListEngagements](#) in *AWS CLI Command Reference*.

list-page-receipts

The following code example shows how to use `list-page-receipts`.

AWS CLI

To list page receipts

The following command-name example lists whether a page was received or not by a contact.

```

aws ssm-contacts list-page-receipts \
  --page-id "arn:aws:ssm-contacts:us-east-2:111122223333:page/
akuam/94ea0c7b-56d9-46c3-b84a-a37c8b067ad3"

```

Output:

```
{
  "Receipts": [
    {
      "ContactChannelArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact-channel/akuam/fc7405c4-46b2-48b7-87b2-93e2f225b90d",
      "ReceiptType": "DELIVERED",
      "ReceiptInfo": "425440",
      "ReceiptTime": "2021-05-18T20:42:57.485000+00:00"
    },
    {
      "ContactChannelArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact-channel/akuam/fc7405c4-46b2-48b7-87b2-93e2f225b90d",
      "ReceiptType": "READ",
      "ReceiptInfo": "425440",
      "ReceiptTime": "2021-05-18T20:42:57.907000+00:00"
    },
    {
      "ContactChannelArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:contact-channel/akuam/fc7405c4-46b2-48b7-87b2-93e2f225b90d",
      "ReceiptType": "SENT",
      "ReceiptInfo": "SM6656c19132f1465f9c9c1123a5dde7c9",
      "ReceiptTime": "2021-05-18T20:40:52.962000+00:00"
    }
  ]
}
```

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [ListPageReceipts](#) in *AWS CLI Command Reference*.

list-pages-by-contact

The following code example shows how to use `list-pages-by-contact`.

AWS CLI

To list pages by contact

The following `list-pages-by-contact` example lists all pages to the specified contact.

```
aws ssm-contacts list-pages-by-contact \
  --contact-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact/akuam"
```

Output:

```
{
  "Pages": [
    {
      "PageArn": "arn:aws:ssm-contacts:us-east-2:111122223333:page/akuam/
ad0052bd-e606-498a-861b-25726292eb93",
      "EngagementArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:engagement/akuam/78a29753-3674-4ac5-9f83-0468563567f0",
      "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
akuam",
      "Sender": "cli",
      "SentTime": "2021-05-18T18:43:29.301000+00:00",
      "DeliveryTime": "2021-05-18T18:43:55.265000+00:00",
      "ReadTime": "2021-05-18T18:43:55.708000+00:00"
    }
  ]
}
```

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [ListPagesByContact](#) in *AWS CLI Command Reference*.

list-pages-by-engagement

The following code example shows how to use `list-pages-by-engagement`.

AWS CLI**To list pages to contact channels started from an engagement.**

The following `list-pages-by-engagement` example lists the pages that occurred while engaging the defined engagement plan.

```
aws ssm-contacts list-pages-by-engagement \
  --engagement-id "arn:aws:ssm-contacts:us-east-2:111122223333:engagement/
akuam/78a29753-3674-4ac5-9f83-0468563567f0"
```

Output:

```
{
  "Pages": [
```

```
{
  "PageArn": "arn:aws:ssm-contacts:us-east-2:111122223333:page/akuam/
ad0052bd-e606-498a-861b-25726292eb93",
  "EngagementArn": "arn:aws:ssm-contacts:us-
east-2:111122223333:engagement/akuam/78a29753-3674-4ac5-9f83-0468563567f0",
  "ContactArn": "arn:aws:ssm-contacts:us-east-2:111122223333:contact/
akuam",
  "Sender": "cli",
  "SentTime": "2021-05-18T18:40:27.245000+00:00"
}
]
```

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [ListPagesByEngagement](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags for a contact

The following `list-tags-for-resource` example lists the tags of the specified contact.

```
aws ssm-contacts list-tags-for-resource \
  --resource-arn "arn:aws:ssm-contacts:us-east-1:111122223333:contact/akuam"
```

Output:

```
{
  "Tags": [
    {
      "Key": "group1",
      "Value": "1"
    }
  ]
}
```

For more information, see [Tagging](#) in the *Incident Manager User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

put-contact-policy

The following code example shows how to use `put-contact-policy`.

AWS CLI

To share a contact and engagements

The following `put-contact-policy` example adds a resource policy to the contact Akua that shares the contact and related engagements with the principal.

```
aws ssm-contacts put-contact-policy \  
  --contact-arn "arn:aws:ssm-contacts:us-east-1:111122223333:contact/akua" \  
  --policy "{\"Version\":\"2012-10-17\",\"Statement\":[{\n\"Sid\":\n\"ExampleResourcePolicy\",\"Action\":[\"ssm-contacts:GetContact\",\"ssm-contacts:StartEngagement\",\"ssm-contacts:DescribeEngagement\",\"ssm-contacts:ListPagesByEngagement\",\"ssm-contacts:StopEngagement\"],\n\"Principal\":{\"AWS\":\"222233334444\"},\"Effect\":\"Allow\",\"Resource\":[\"arn:aws:ssm-contacts:*:111122223333:contact/akua\",\"arn:aws:ssm-contacts:*:111122223333:engagement/akua/*\"]}]}"
```

This command produces no output.

For more information, see [Working with shared contacts and response plans](#) in the *Incident Manager User Guide*.

- For API details, see [PutContactPolicy](#) in *AWS CLI Command Reference*.

send-activation-code

The following code example shows how to use `send-activation-code`.

AWS CLI

To send an activation code

The following `send-activation-code` example sends an activation code and message to the specified contact channel.

```
aws ssm-contacts send-activation-code \  
  --contact-arn "arn:aws:ssm-contacts:us-east-1:111122223333:contact/akua" \  
  --channel "channel" \  
  --message "message" \  
  --subject "subject"
```

```
--contact-channel-id "arn:aws:ssm-contacts:us-east-1:111122223333:contact-channel/akuam/8ddae2d1-12c8-4e45-b852-c8587266c400"
```

This command produces no output.

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [SendActivationCode](#) in *AWS CLI Command Reference*.

start-engagement

The following code example shows how to use start-engagement.

AWS CLI

Example 1: To page a contact's contact channels

The following start-engagement pages contact's contact channels. Sender, subject, public-subject, and public-content are all free from fields. Incident Manager sends the subject and content to the provided VOICE or EMAIL contact channels. Incident Manager sends the public-subject and public-content to the provided SMS contact channels. Sender is used to track who started the engagement.

```
aws ssm-contacts start-engagement \  
  --contact-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact/akuam" \  
  --sender "cli" \  
  --subject "cli-test" \  
  --content "Testing engagements via CLI" \  
  --public-subject "cli-test" \  
  --public-content "Testing engagements va CLI"
```

Output:

```
{  
  "EngagementArn": "arn:aws:ssm-contacts:us-east-2:111122223333:engagement/  
akuam/607ced0e-e8fa-4ea7-8958-a237b8803f8f"  
}
```

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

Example 2: To page a contact in the provided escalation plan.

The following `start-engagement` engages contact's through an escalation plan. Each contact is paged according to their engagement plan.

```
aws ssm-contacts start-engagement \  
  --contact-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact/  
example_escalation" \  
  --sender "cli" \  
  --subject "cli-test" \  
  --content "Testing engagements via CLI" \  
  --public-subject "cli-test" \  
  --public-content "Testing engagements va CLI"
```

Output:

```
{  
  "EngagementArn": "arn:aws:ssm-contacts:us-east-2:111122223333:engagement/  
example_escalation/69e40ce1-8dbb-4d57-8962-5fbe7fc53356"  
}
```

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [StartEngagement](#) in *AWS CLI Command Reference*.

stop-engagement

The following code example shows how to use `stop-engagement`.

AWS CLI

To stop an engagement

The following `stop-engagement` example stops an engagement from paging further contacts and contact channels.

```
aws ssm-contacts stop-engagement \  
  --engagement-id "arn:aws:ssm-contacts:us-east-2:111122223333:engagement/  
example_escalation/69e40ce1-8dbb-4d57-8962-5fbe7fc53356"
```

This command produces no output.

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [StopEngagement](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To tag a contact

The following `tag-resource` example tags a specified contact with the provided tag key value pair.

```
aws ssm-contacts tag-resource \  
  --resource-arn "arn:aws:ssm-contacts:us-east-1:111122223333:contact/akuam" \  
  --tags '[{"Key":"group1","Value":"1"}]'
```

This command produces no output.

For more information, see [Tagging](#) in the *Incident Manager User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from a contact

The following `untag-resource` example removes the `group1` tag from the specified contact.

```
aws ssm-contacts untag-resource \  
  --resource-arn "arn:aws:ssm-contacts:us-east-1:111122223333:contact/akuam" \  
  --tag-keys "group1"
```

This command produces no output.

For more information, see [Tagging](#) in the *Incident Manager User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-contact-channel

The following code example shows how to use `update-contact-channel`.

AWS CLI

To update a contact channel

The following `update-contact-channel` example updates the name and delivery address of a contact channel.

```
aws ssm-contacts update-contact-channel \  
  --contact-channel-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact-  
channel/akuam/49f3c24d-5f9f-4638-ae25-3f49e04229ad" \  
  --name "akuas voice channel" \  
  --delivery-address '{"SimpleAddress": "+15005550198"}'
```

This command produces no output.

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [UpdateContactChannel](#) in *AWS CLI Command Reference*.

update-contact

The following code example shows how to use `update-contact`.

AWS CLI

To update the engagement plan of contact

The following `update-contact` example updates the engagement plan of the contact Akua to include the three types of contacts channels. This is done after creating contact channels for Akua.

```
aws ssm-contacts update-contact \  
  --contact-id "arn:aws:ssm-contacts:us-east-2:111122223333:contact/akuam" \  
  --plan '{"Stages": [{"DurationInMinutes": 5, "Targets": [{"ChannelTargetInfo":  
{"ContactChannelId": "arn:aws:ssm-contacts:us-east-2:111122223333:contact-  
channel/akuam/beb25840-5ac8-4644-95cc-7a8de390fa65", "RetryIntervalInMinutes":  
1 }]}], {"DurationInMinutes": 5, "Targets": [{"ChannelTargetInfo":  
{"ContactChannelId": "arn:aws:ssm-contacts:us-east-2:111122223333:contact-channel/  
akuam/49f3c24d-5f9f-4638-ae25-3f49e04229ad", "RetryIntervalInMinutes": 1}]}}],
```

```
{"DurationInMinutes": 5, "Targets": [{"ChannelTargetInfo": {"ContactChannelId": "arn:aws:ssm-contacts:us-east-2:111122223333:contact-channel/akuam/77d4f447-f619-4954-afff-85551e369c2a", "RetryIntervalInMinutes": 1 }]}]}
```

This command produces no output.

For more information, see [Contacts](#) in the *Incident Manager User Guide*.

- For API details, see [UpdateContact](#) in *AWS CLI Command Reference*.

Amazon Inspector examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon Inspector.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-attributes-to-findings

The following code example shows how to use `add-attributes-to-findings`.

AWS CLI

To add attributes to findings

The following `add-attribute-to-finding` command assigns an attribute with the key of `example` and value of `example` to the finding with the ARN of `arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-811VIE0D/run/0-Z02cjjug/finding/0-T8yM9mEU`:

```
aws inspector add-attributes-to-findings --finding-arns arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-811VIE0D/run/0-Z02cjjug/finding/0-T8yM9mEU --attributes key=Example,value=example
```

Output:

```
{
  "failedItems": {}
}
```

For more information, see Amazon Inspector Findings in the *Amazon Inspector* guide.

- For API details, see [AddAttributesToFindings](#) in *AWS CLI Command Reference*.

create-assessment-target

The following code example shows how to use create-assessment-target.

AWS CLI

To create an assessment target

The following create-assessment-target command creates an assessment target named ExampleAssessmentTarget using the resource group with the ARN of arn:aws:inspector:us-west-2:123456789012:resourcegroup/0-AB6DMKnv:

```
aws inspector create-assessment-target --assessment-target-name
ExampleAssessmentTarget --resource-group-arn arn:aws:inspector:us-
west-2:123456789012:resourcegroup/0-AB6DMKnv
```

Output:

```
{
  "assessmentTargetArn": "arn:aws:inspector:us-west-2:123456789012:target/0-
nvgVhaxX"
}
```

For more information, see Amazon Inspector Assessment Targets in the *Amazon Inspector* guide.

- For API details, see [CreateAssessmentTarget](#) in *AWS CLI Command Reference*.

create-assessment-template

The following code example shows how to use `create-assessment-template`.

AWS CLI

To create an assessment template

The following `create-assessment-template` command creates an assessment template called `ExampleAssessmentTemplate` for the assessment target with the ARN of `arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX`:

```
aws inspector create-assessment-template --assessment-target-arn
arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX --assessment-template-
name ExampleAssessmentTemplate --duration-in-seconds 180 --rules-package-arns
arn:aws:inspector:us-west-2:758058086616:rulespackage/0-9hgA516p --user-attributes-
for-findings key=ExampleTag,value=examplevalue
```

Output:

```
{
  "assessmentTemplateArn": "arn:aws:inspector:us-west-2:123456789012:target/0-
nvgVhaxX/template/0-it5r2S4T"
}
```

For more information, see Amazon Inspector Assessment Templates and Assessment Runs in the *Amazon Inspector* guide.

- For API details, see [CreateAssessmentTemplate](#) in *AWS CLI Command Reference*.

create-resource-group

The following code example shows how to use `create-resource-group`.

AWS CLI

To create a resource group

The following `create-resource-group` command creates a resource group using the tag key of `Name` and value of `example`:

```
aws inspector create-resource-group --resource-group-tags key=Name,value=example
```

Output:

```
{
  "resourceGroupArn": "arn:aws:inspector:us-west-2:123456789012:resourcegroup/0-AB6DMKnv"
}
```

For more information, see Amazon Inspector Assessment Targets in the *Amazon Inspector* guide.

- For API details, see [CreateResourceGroup](#) in *AWS CLI Command Reference*.

delete-assessment-run

The following code example shows how to use `delete-assessment-run`.

AWS CLI**To delete an assessment run**

The following `delete-assessment-run` command deletes the assessment run with the ARN of `arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-it5r2S4T/run/0-11LMTAVe`:

```
aws inspector delete-assessment-run --assessment-run-arn arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-it5r2S4T/run/0-11LMTAVe
```

For more information, see Amazon Inspector Assessment Templates and Assessment Runs in the *Amazon Inspector* guide.

- For API details, see [DeleteAssessmentRun](#) in *AWS CLI Command Reference*.

delete-assessment-target

The following code example shows how to use `delete-assessment-target`.

AWS CLI**To delete an assessment target**

The following `delete-assessment-target` command deletes the assessment target with the ARN of `arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq`:

```
aws inspector delete-assessment-target --assessment-target-arn arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq
```

For more information, see Amazon Inspector Assessment Targets in the *Amazon Inspector* guide.

- For API details, see [DeleteAssessmentTarget](#) in *AWS CLI Command Reference*.

delete-assessment-template

The following code example shows how to use `delete-assessment-template`.

AWS CLI

To delete an assessment template

The following `delete-assessment-template` command deletes the assessment template with the ARN of `arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-it5r2S4T`:

```
aws inspector delete-assessment-template --assessment-template-arn arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-it5r2S4T
```

For more information, see Amazon Inspector Assessment Templates and Assessment Runs in the *Amazon Inspector* guide.

- For API details, see [DeleteAssessmentTemplate](#) in *AWS CLI Command Reference*.

describe-assessment-runs

The following code example shows how to use `describe-assessment-runs`.

AWS CLI

To describe assessment runs

The following `describe-assessment-run` command describes an assessment run with the ARN of `arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw/run/0-MKkpXXPE`:

```
aws inspector describe-assessment-runs --assessment-run-arns arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw/run/0-MKkpXXPE
```

Output:

```
{
  "assessmentRuns": [
    {
      "arn": "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/
template/0-4r1V2mAw/run/0-MKkpXXPE",
      "assessmentTemplateArn": "arn:aws:inspector:us-
west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw",
      "completedAt": 1458680301.4,
      "createdAt": 1458680170.035,
      "dataCollected": true,
      "durationInSeconds": 3600,
      "name": "Run 1 for ExampleAssessmentTemplate",
      "notifications": [],
      "rulesPackageArns": [
        "arn:aws:inspector:us-west-2:758058086616:rulespackage/0-X1KXtawP"
      ],
      "startedAt": 1458680170.161,
      "state": "COMPLETED",
      "stateChangedAt": 1458680301.4,
      "stateChanges": [
        {
          "state": "CREATED",
          "stateChangedAt": 1458680170.035
        },
        {
          "state": "START_DATA_COLLECTION_PENDING",
          "stateChangedAt": 1458680170.065
        },
        {
          "state": "START_DATA_COLLECTION_IN_PROGRESS",
          "stateChangedAt": 1458680170.096
        },
        {
          "state": "COLLECTING_DATA",
          "stateChangedAt": 1458680170.161
        },
        {
          "state": "STOP_DATA_COLLECTION_PENDING",
          "stateChangedAt": 1458680239.883
        },
        {
          "state": "DATA_COLLECTED",
```



```

        "stateChangedAt": 1458680299.847
      },
      {
        "state": "EVALUATING_RULES",
        "stateChangedAt": 1458680300.099
      },
      {
        "state": "COMPLETED",
        "stateChangedAt": 1458680301.4
      }
    ],
    "userAttributesForFindings": []
  }
],
"failedItems": {}
}

```

For more information, see Amazon Inspector Assessment Templates and Assessment Runs in the *Amazon Inspector* guide.

- For API details, see [DescribeAssessmentRuns](#) in *AWS CLI Command Reference*.

describe-assessment-targets

The following code example shows how to use `describe-assessment-targets`.

AWS CLI

To describe assessment targets

The following `describe-assessment-targets` command describes the assessment target with the ARN of `arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq`:

```
aws inspector describe-assessment-targets --assessment-target-arns
arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq
```

Output:

```
{
  "assessmentTargets": [
    {
      "arn": "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq",
```

```

        "createdAt": 1458074191.459,
        "name": "ExampleAssessmentTarget",
        "resourceGroupArn": "arn:aws:inspector:us-
west-2:123456789012:resourcegroup/0-PyGXopAI",
        "updatedAt": 1458074191.459
    }
  ],
  "failedItems": {}
}

```

For more information, see Amazon Inspector Assessment Targets in the *Amazon Inspector* guide.

- For API details, see [DescribeAssessmentTargets](#) in *AWS CLI Command Reference*.

describe-assessment-templates

The following code example shows how to use describe-assessment-templates.

AWS CLI

To describe assessment templates

The following describe-assessment-templates command describes the assessment template with the ARN of arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw:

```
aws inspector describe-assessment-templates --assessment-template-arns
arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw
```

Output:

```

{
  "assessmentTemplates": [
    {
      "arn": "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/
template/0-4r1V2mAw",
      "assessmentTargetArn": "arn:aws:inspector:us-
west-2:123456789012:target/0-0kFIPusq",
      "createdAt": 1458074191.844,
      "durationInSeconds": 3600,
      "name": "ExampleAssessmentTemplate",
      "rulesPackageArns": [

```

```
        "arn:aws:inspector:us-west-2:758058086616:rulespackage/0-X1KXtawP"  
    ],  
    "userAttributesForFindings": []  
  }  
],  
"failedItems": {}  
}
```

For more information, see Amazon Inspector Assessment Templates and Assessment Runs in the *Amazon Inspector* guide.

- For API details, see [DescribeAssessmentTemplates](#) in *AWS CLI Command Reference*.

describe-cross-account-access-role

The following code example shows how to use `describe-cross-account-access-role`.

AWS CLI

To describe the cross account access role

The following `describe-cross-account-access-role` command describes the IAM role that enables Amazon Inspector to access your AWS account:

```
aws inspector describe-cross-account-access-role
```

Output:

```
{  
    "registeredAt": 1458069182.826,  
    "roleArn": "arn:aws:iam::123456789012:role/inspector",  
    "valid": true  
}
```

For more information, see Setting up Amazon Inspector in the *Amazon Inspector* guide.

- For API details, see [DescribeCrossAccountAccessRole](#) in *AWS CLI Command Reference*.

describe-findings

The following code example shows how to use `describe-findings`.

AWS CLI

To describe findings

The following `describe-findings` command describes the finding with the ARN of `arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw/run/0-MKkpXXPE/finding/0-HwPnsDm4`:

```
aws inspector describe-findings --finding-arns arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw/run/0-MKkpXXPE/finding/0-HwPnsDm4
```

Output:

```
{
  "failedItems": {},
  "findings": [
    {
      "arn": "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw/run/0-MKkpXXPE/finding/0-HwPnsDm4",
      "assetAttributes": {
        "ipv4Addresses": [],
        "schemaVersion": 1
      },
      "assetType": "ec2-instance",
      "attributes": [],
      "confidence": 10,
      "createdAt": 1458680301.37,
      "description": "Amazon Inspector did not find any potential security issues during this assessment.",
      "indicatorOfCompromise": false,
      "numericSeverity": 0,
      "recommendation": "No remediation needed.",
      "schemaVersion": 1,
      "service": "Inspector",
      "serviceAttributes": {
        "assessmentRunArn": "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw/run/0-MKkpXXPE",
        "rulesPackageArn": "arn:aws:inspector:us-west-2:758058086616:rulespackage/0-X1KXtawP",
        "schemaVersion": 1
      },
      "severity": "Informational",
    }
  ]
}
```

```
        "title": "No potential security issues found",
        "updatedAt": 1458680301.37,
        "userAttributes": []
      }
    ]
  }
```

For more information, see Amazon Inspector Findings in the *Amazon Inspector* guide.

- For API details, see [DescribeFindings](#) in *AWS CLI Command Reference*.

describe-resource-groups

The following code example shows how to use `describe-resource-groups`.

AWS CLI

To describe resource groups

The following `describe-resource-groups` command describes the resource group with the ARN of `arn:aws:inspector:us-west-2:123456789012:resourcegroup/0-PyGXopAI`:

```
aws inspector describe-resource-groups --resource-group-arns arn:aws:inspector:us-west-2:123456789012:resourcegroup/0-PyGXopAI
```

Output:

```
{
  "failedItems": {},
  "resourceGroups": [
    {
      "arn": "arn:aws:inspector:us-west-2:123456789012:resourcegroup/0-PyGXopAI",
      "createdAt": 1458074191.098,
      "tags": [
        {
          "key": "Name",
          "value": "example"
        }
      ]
    }
  ]
}
```

```
}
```

For more information, see Amazon Inspector Assessment Targets in the *Amazon Inspector* guide.

- For API details, see [DescribeResourceGroups](#) in *AWS CLI Command Reference*.

describe-rules-packages

The following code example shows how to use `describe-rules-packages`.

AWS CLI

To describe rules packages

The following `describe-rules-packages` command describes the rules package with the ARN of `arn:aws:inspector:us-west-2:758058086616:rulespackage/0-9hgA516p`:

```
aws inspector describe-rules-packages --rules-package-arns arn:aws:inspector:us-west-2:758058086616:rulespackage/0-9hgA516p
```

Output:

```
{
  "failedItems": {},
  "rulesPackages": [
    {
      "arn": "arn:aws:inspector:us-west-2:758058086616:rulespackage/0-9hgA516p",
      "description": "The rules in this package help verify whether the EC2 instances in your application are exposed to Common Vulnerabilities and Exposures (CVEs). Attacks can exploit unpatched vulnerabilities to compromise the confidentiality, integrity, or availability of your service or data. The CVE system provides a reference for publicly known information security vulnerabilities and exposures. For more information, see [https://cve.mitre.org/](https://cve.mitre.org/). If a particular CVE appears in one of the produced Findings at the end of a completed Inspector assessment, you can search [https://cve.mitre.org/](https://cve.mitre.org/) using the CVE's ID (for example, \"CVE-2009-0021\") to find detailed information about this CVE, its severity, and how to mitigate it. ",
      "name": "Common Vulnerabilities and Exposures",
      "provider": "Amazon Web Services, Inc.",
      "version": "1.1"
    }
  ]
}
```

```
    }
  ]
}
```

For more information, see Amazon Inspector Rules Packages and Rules in the *Amazon Inspector* guide.

- For API details, see [DescribeRulesPackages](#) in *AWS CLI Command Reference*.

get-telemetry-metadata

The following code example shows how to use `get-telemetry-metadata`.

AWS CLI

To get the telemetry metadata

The following `get-telemetry-metadata` command generates information about the data that is collected for the assessment run with the ARN of `arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw/run/0-MKkpXXPE`:

```
aws inspector get-telemetry-metadata --assessment-run-arn arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw/run/0-MKkpXXPE
```

Output:

```
{
  "telemetryMetadata": [
    {
      "count": 2,
      "dataSize": 345,
      "messageType": "InspectorDuplicateProcess"
    },
    {
      "count": 3,
      "dataSize": 255,
      "messageType": "InspectorTimeEventMsg"
    },
    {
      "count": 4,
      "dataSize": 1082,
      "messageType": "InspectorNetworkInterface"
    }
  ]
}
```

```
  },
  {
    "count": 2,
    "dataSize": 349,
    "messageType": "InspectorDnsEntry"
  },
  {
    "count": 11,
    "dataSize": 2514,
    "messageType": "InspectorDirectoryInfoMsg"
  },
  {
    "count": 1,
    "dataSize": 179,
    "messageType": "InspectorTcpV6ListeningPort"
  },
  {
    "count": 101,
    "dataSize": 10949,
    "messageType": "InspectorTerminal"
  },
  {
    "count": 26,
    "dataSize": 5916,
    "messageType": "InspectorUser"
  },
  {
    "count": 282,
    "dataSize": 32148,
    "messageType": "InspectorDynamicallyLoadedCodeModule"
  },
  {
    "count": 18,
    "dataSize": 10172,
    "messageType": "InspectorCreateProcess"
  },
  {
    "count": 3,
    "dataSize": 8001,
    "messageType": "InspectorProcessPerformance"
  },
  {
    "count": 1,
    "dataSize": 360,
```



```
    "messageType": "InspectorOperatingSystem"
  },
  {
    "count": 6,
    "dataSize": 546,
    "messageType": "InspectorStopProcess"
  },
  {
    "count": 1,
    "dataSize": 1553,
    "messageType": "InspectorInstanceMetaData"
  },
  {
    "count": 2,
    "dataSize": 434,
    "messageType": "InspectorTcpV4Connection"
  },
  {
    "count": 474,
    "dataSize": 2960322,
    "messageType": "InspectorPackageInfo"
  },
  {
    "count": 3,
    "dataSize": 2235,
    "messageType": "InspectorSystemPerformance"
  },
  {
    "count": 105,
    "dataSize": 46048,
    "messageType": "InspectorCodeModule"
  },
  {
    "count": 1,
    "dataSize": 182,
    "messageType": "InspectorUdpV6ListeningPort"
  },
  {
    "count": 2,
    "dataSize": 371,
    "messageType": "InspectorUdpV4ListeningPort"
  },
  {
    "count": 18,
```

```
    "dataSize": 8362,
    "messageType": "InspectorKernelModule"
  },
  {
    "count": 29,
    "dataSize": 48788,
    "messageType": "InspectorConfigurationInfo"
  },
  {
    "count": 1,
    "dataSize": 79,
    "messageType": "InspectorMonitoringStart"
  },
  {
    "count": 5,
    "dataSize": 0,
    "messageType": "InspectorSplitMsgBegin"
  },
  {
    "count": 51,
    "dataSize": 4593,
    "messageType": "InspectorGroup"
  },
  {
    "count": 1,
    "dataSize": 184,
    "messageType": "InspectorTcpV4ListeningPort"
  },
  {
    "count": 1159,
    "dataSize": 3146579,
    "messageType": "Total"
  },
  {
    "count": 5,
    "dataSize": 0,
    "messageType": "InspectorSplitMsgEnd"
  },
  {
    "count": 1,
    "dataSize": 612,
    "messageType": "InspectorLoadImageInProgress"
  }
]
```

```
}
```

- For API details, see [GetTelemetryMetadata](#) in *AWS CLI Command Reference*.

list-assessment-run-agents

The following code example shows how to use `list-assessment-run-agents`.

AWS CLI

To list assessment run agents

The following `list-assessment-run-agents` command lists the agents of the assessment run with the specified ARN.

```
aws inspector list-assessment-run-agents \
  --assessment-run-arn arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/
  template/0-4r1V2mAw/run/0-MKkpXXPE
```

Output:

```
{
  "assessmentRunAgents": [
    {
      "agentHealth": "HEALTHY",
      "agentHealthCode": "HEALTHY",
      "agentId": "i-49113b93",
      "assessmentRunArn": "arn:aws:inspector:us-
west-2:123456789012:target/0-0kFIPusq/template/0-4r1V2mAw/run/0-MKkpXXPE",
      "telemetryMetadata": [
        {
          "count": 2,
          "dataSize": 345,
          "messageType": "InspectorDuplicateProcess"
        },
        {
          "count": 3,
          "dataSize": 255,
          "messageType": "InspectorTimeEventMsg"
        },
        {
          "count": 4,
```

```
        "dataSize": 1082,  
        "messageType": "InspectorNetworkInterface"  
    },  
    {  
        "count": 2,  
        "dataSize": 349,  
        "messageType": "InspectorDnsEntry"  
    },  
    {  
        "count": 11,  
        "dataSize": 2514,  
        "messageType": "InspectorDirectoryInfoMsg"  
    },  
    {  
        "count": 1,  
        "dataSize": 179,  
        "messageType": "InspectorTcpV6ListeningPort"  
    },  
    {  
        "count": 101,  
        "dataSize": 10949,  
        "messageType": "InspectorTerminal"  
    },  
    {  
        "count": 26,  
        "dataSize": 5916,  
        "messageType": "InspectorUser"  
    },  
    {  
        "count": 282,  
        "dataSize": 32148,  
        "messageType": "InspectorDynamicallyLoadedCodeModule"  
    },  
    {  
        "count": 18,  
        "dataSize": 10172,  
        "messageType": "InspectorCreateProcess"  
    },  
    {  
        "count": 3,  
        "dataSize": 8001,  
        "messageType": "InspectorProcessPerformance"  
    },  
    {
```

```
    "count": 1,  
    "dataSize": 360,  
    "messageType": "InspectorOperatingSystem"  
  },  
  {  
    "count": 6,  
    "dataSize": 546,  
    "messageType": "InspectorStopProcess"  
  },  
  {  
    "count": 1,  
    "dataSize": 1553,  
    "messageType": "InspectorInstanceMetaData"  
  },  
  {  
    "count": 2,  
    "dataSize": 434,  
    "messageType": "InspectorTcpV4Connection"  
  },  
  {  
    "count": 474,  
    "dataSize": 2960322,  
    "messageType": "InspectorPackageInfo"  
  },  
  {  
    "count": 3,  
    "dataSize": 2235,  
    "messageType": "InspectorSystemPerformance"  
  },  
  {  
    "count": 105,  
    "dataSize": 46048,  
    "messageType": "InspectorCodeModule"  
  },  
  {  
    "count": 1,  
    "dataSize": 182,  
    "messageType": "InspectorUdpV6ListeningPort"  
  },  
  {  
    "count": 2,  
    "dataSize": 371,  
    "messageType": "InspectorUdpV4ListeningPort"  
  },  
}
```

```
{
  "count": 18,
  "dataSize": 8362,
  "messageType": "InspectorKernelModule"
},
{
  "count": 29,
  "dataSize": 48788,
  "messageType": "InspectorConfigurationInfo"
},
{
  "count": 1,
  "dataSize": 79,
  "messageType": "InspectorMonitoringStart"
},
{
  "count": 5,
  "dataSize": 0,
  "messageType": "InspectorSplitMsgBegin"
},
{
  "count": 51,
  "dataSize": 4593,
  "messageType": "InspectorGroup"
},
{
  "count": 1,
  "dataSize": 184,
  "messageType": "InspectorTcpV4ListeningPort"
},
{
  "count": 1159,
  "dataSize": 3146579,
  "messageType": "Total"
},
{
  "count": 5,
  "dataSize": 0,
  "messageType": "InspectorSplitMsgEnd"
},
{
  "count": 1,
  "dataSize": 612,
  "messageType": "InspectorLoadImageInProgress"
}
```

```
}
  ]
}
]
```

For more information, see [AWS Agents](#) in the *Amazon Inspector User Guide*.

- For API details, see [ListAssessmentRunAgents](#) in *AWS CLI Command Reference*.

list-assessment-runs

The following code example shows how to use `list-assessment-runs`.

AWS CLI

To list assessment runs

The following `list-assessment-runs` command lists all existing assessment runs.

```
aws inspector list-assessment-runs
```

Output:

```
{
  "assessmentRunArns": [
    "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/
template/0-4r1V2mAw/run/0-MKkpXXPE",
    "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/
template/0-4r1V2mAw/run/0-v5D6fI3v"
  ]
}
```

For more information, see [Amazon Inspector Assessment Templates and Assessment Runs](#) in the *Amazon Inspector User Guide*.

- For API details, see [ListAssessmentRuns](#) in *AWS CLI Command Reference*.

list-assessment-targets

The following code example shows how to use `list-assessment-targets`.

AWS CLI

To list assessment targets

The following `list-assessment-targets` command lists all existing assessment targets:

```
aws inspector list-assessment-targets
```

Output:

```
{
  "assessmentTargetArns": [
    "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq"
  ]
}
```

For more information, see Amazon Inspector Assessment Targets in the *Amazon Inspector* guide.

- For API details, see [ListAssessmentTargets](#) in *AWS CLI Command Reference*.

list-assessment-templates

The following code example shows how to use `list-assessment-templates`.

AWS CLI

To list assessment templates

The following `list-assessment-templates` command lists all existing assessment templates:

```
aws inspector list-assessment-templates
```

Output:

```
{
  "assessmentTemplateArns": [
    "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/
template/0-4r1V2mAw",
    "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-
Uza6ihLh"
  ]
}
```



```
}
```

For more information, see Amazon Inspector Assessment Templates and Assessment Runs in the *Amazon Inspector* guide.

- For API details, see [ListAssessmentTemplates](#) in *AWS CLI Command Reference*.

list-event-subscriptions

The following code example shows how to use `list-event-subscriptions`.

AWS CLI

To list event subscriptions

The following `list-event-subscriptions` command lists all the event subscriptions for the assessment template with the ARN of `arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-7sbz2Kz0`:

```
aws inspector list-event-subscriptions --resource-arn arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-7sbz2Kz0
```

Output:

```
{
  "subscriptions": [
    {
      "eventSubscriptions": [
        {
          "event": "ASSESSMENT_RUN_COMPLETED",
          "subscribedAt": 1459455440.867
        }
      ],
      "resourceArn": "arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-7sbz2Kz0",
      "topicArn": "arn:aws:sns:us-west-2:123456789012:exampletopic"
    }
  ]
}
```

For more information, see Amazon Inspector Assessment Templates and Assessment Runs in the *Amazon Inspector* guide.

- For API details, see [ListEventSubscriptions](#) in *AWS CLI Command Reference*.

list-findings

The following code example shows how to use `list-findings`.

AWS CLI

To list findings

The following `list-findings` command lists all of the generated findings:

```
aws inspector list-findings
```

Output:

```
{
  "findingArns": [
    "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/
template/0-4r1V2mAw/run/0-MKkpXXPE/finding/0-HwPnsDm4",
    "arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/
template/0-4r1V2mAw/run/0-v5D6fI3v/finding/0-tyvmqBLy"
  ]
}
```

For more information, see Amazon Inspector Findings in the *Amazon Inspector* guide.

- For API details, see [ListFindings](#) in *AWS CLI Command Reference*.

list-rules-packages

The following code example shows how to use `list-rules-packages`.

AWS CLI

To list rules packages

The following `list-rules-packages` command lists all available Inspector rules packages:

```
aws inspector list-rules-packages
```

Output:

```
{
  "rulesPackageArns": [
    "arn:aws:inspector:us-west-2:758058086616:rulespackage/0-9hgA516p",
    "arn:aws:inspector:us-west-2:758058086616:rulespackage/0-H5hpSawc",
    "arn:aws:inspector:us-west-2:758058086616:rulespackage/0-JJ0tZiqQ",
    "arn:aws:inspector:us-west-2:758058086616:rulespackage/0-vg5GGHSD"
  ]
}
```

For more information, see Amazon Inspector Rules Packages and Rules in the *Amazon Inspector* guide.

- For API details, see [ListRulesPackages](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI**To list tags for resource**

The following `list-tags-for-resource` command lists all tags associated with the assessment template with the ARN of `arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-gcwFliYu`:

```
aws inspector list-tags-for-resource --resource-arn arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-gcwFliYu
```

Output:

```
{
  "tags": [
    {
      "key": "Name",
      "value": "Example"
    }
  ]
}
```

For more information, see Amazon Inspector Assessment Templates and Assessment Runs in the *Amazon Inspector* guide.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

preview-agents

The following code example shows how to use `preview-agents`.

AWS CLI

To preview agents

The following `preview-agents` command previews the agents installed on the EC2 instances that are part of the assessment target with the ARN of `arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq`:

```
aws inspector preview-agents --preview-agents-arn arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq
```

Output:

```
{
  "agentPreviews": [
    {
      "agentId": "i-49113b93"
    }
  ]
}
```

For more information, see Amazon Inspector Assessment Targets in the *Amazon Inspector* guide.

- For API details, see [PreviewAgents](#) in *AWS CLI Command Reference*.

register-cross-account-access-role

The following code example shows how to use `register-cross-account-access-role`.

AWS CLI

To register the cross account access role

The following `register-cross-account-access-role` command registers the IAM role with the ARN of `arn:aws:iam::123456789012:role/inspector` that Amazon Inspector uses to list your EC2 instances at the start of the assessment run of when you call the `preview-agents` command:

```
aws inspector register-cross-account-access-role --role-arn
arn:aws:iam::123456789012:role/inspector
```

For more information, see *Setting up Amazon Inspector* in the *Amazon Inspector* guide.

- For API details, see [RegisterCrossAccountAccessRole](#) in *AWS CLI Command Reference*.

remove-attributes-from-findings

The following code example shows how to use `remove-attributes-from-findings`.

AWS CLI

To remove attributes from findings

The following `remove-attributes-from-finding` command removes the attribute with the key of `Example` and value of `example` from the finding with the ARN of `arn:aws:inspector:us-west-2:123456789012:target/0-0kFIPusq/template/0-811VIE0D/run/0-Z02cjjug/finding/0-T8yM9mEU`:

```
aws inspector remove-attributes-from-findings --finding-arns arn:aws:inspector:us-
west-2:123456789012:target/0-0kFIPusq/template/0-811VIE0D/run/0-Z02cjjug/finding/0-
T8yM9mEU --attribute-keys key=Example,value=example
```

Output:

```
{
  "failedItems": {}
}
```

For more information, see *Amazon Inspector Findings* in the *Amazon Inspector* guide.

- For API details, see [RemoveAttributesFromFindings](#) in *AWS CLI Command Reference*.

set-tags-for-resource

The following code example shows how to use `set-tags-for-resource`.

AWS CLI

To set tags for a resource

The following `set-tags-for-resource` command sets the tag with the key of `Example` and value of `example` to the assessment template with the ARN of `arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-7sbz2Kz0`:

```
aws inspector set-tags-for-resource --resource-arn arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-7sbz2Kz0 --tags key=Example,value=example
```

For more information, see *Amazon Inspector Assessment Templates and Assessment Runs* in the *Amazon Inspector* guide.

- For API details, see [SetTagsForResource](#) in *AWS CLI Command Reference*.

start-assessment-run

The following code example shows how to use `start-assessment-run`.

AWS CLI

To start an assessment run

The following `start-assessment-run` command starts the assessment run named `examplerrun` using the assessment template with the ARN of `arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-it5r2S4T`:

```
aws inspector start-assessment-run --assessment-run-name examplerrun --assessment-template-arn arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-it5r2S4T
```

Output:

```
{
  "assessmentRunArn": "arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-it5r2S4T/run/0-j0o0oxyY"
```

```
}
```

For more information, see Amazon Inspector Assessment Templates and Assessment Runs in the *Amazon Inspector* guide.

- For API details, see [StartAssessmentRun](#) in *AWS CLI Command Reference*.

stop-assessment-run

The following code example shows how to use `stop-assessment-run`.

AWS CLI

To stop an assessment run

The following `stop-assessment-run` command stops the assessment run with the ARN of `arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-it5r2S4T/run/0-j0oroxyY`:

```
aws inspector stop-assessment-run --assessment-run-arn arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-it5r2S4T/run/0-j0oroxyY
```

For more information, see Amazon Inspector Assessment Templates and Assessment Runs in the *Amazon Inspector* guide.

- For API details, see [StopAssessmentRun](#) in *AWS CLI Command Reference*.

subscribe-to-event

The following code example shows how to use `subscribe-to-event`.

AWS CLI

To subscribe to an event

The following example enables the process of sending Amazon SNS notifications about the `ASSESSMENT_RUN_COMPLETED` event to the topic with the ARN of `arn:aws:sns:us-west-2:123456789012:exampletopic`

```
aws inspector subscribe-to-event \  
  --event ASSESSMENT_RUN_COMPLETED \  
  --topic-arn arn:aws:sns:us-west-2:123456789012:exampletopic
```

```
--resource-arn arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/  
template/0-7sbz2Kz0 \  
--topic-arn arn:aws:sns:us-west-2:123456789012:exampletopic
```

This command produces no output.

For more information, see [Amazon Inspector Assessment Templates and Assessment Runs](#) in the *Amazon Inspector* guide.

- For API details, see [SubscribeToEvent](#) in *AWS CLI Command Reference*.

unsubscribe-from-event

The following code example shows how to use `unsubscribe-from-event`.

AWS CLI

To unsubscribe from an event

The following `unsubscribe-from-event` command disables the process of sending Amazon SNS notifications about the `ASSESSMENT_RUN_COMPLETED` event to the topic with the ARN of `arn:aws:sns:us-west-2:123456789012:exampletopic`:

```
aws inspector unsubscribe-from-event --event ASSESSMENT_RUN_COMPLETED --resource-arn  
arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX/template/0-7sbz2Kz0 --  
topic arn:aws:sns:us-west-2:123456789012:exampletopic
```

For more information, see [Amazon Inspector Assessment Templates and Assessment Runs](#) in the *Amazon Inspector* guide.

- For API details, see [UnsubscribeFromEvent](#) in *AWS CLI Command Reference*.

update-assessment-target

The following code example shows how to use `update-assessment-target`.

AWS CLI

To update an assessment target

The following `update-assessment-target` command updates the assessment target with the ARN of `arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX` and

the name of Example, and the resource group with the ARN of `arn:aws:inspector:us-west-2:123456789012:resourcegroup/0-yNbgL5Pt`:

```
aws inspector update-assessment-target --assessment-target-arn arn:aws:inspector:us-west-2:123456789012:target/0-nvgVhaxX --assessment-target-name Example --resource-group-arn arn:aws:inspector:us-west-2:123456789012:resourcegroup/0-yNbgL5Pt
```

For more information, see Amazon Inspector Assessment Targets in the *Amazon Inspector* guide.

- For API details, see [UpdateAssessmentTarget](#) in *AWS CLI Command Reference*.

AWS IoT examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS IoT.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

accept-certificate-transfer

The following code example shows how to use `accept-certificate-transfer`.

AWS CLI

To accept a device certificate transferred from a different AWS account

The following `accept-certificate-transfer` example accepts a device certificate transferred from another AWS account. The certificate is identified by its ID.

```
aws iot accept-certificate-transfer \  
  --certificate-id  
  488b6a7f2acdeb00a77384e63c4e40b18bEXAMPLEe57b7272ba44c45e3448142
```

This command does not produce any output.

For more information, see [Transfer a certificate to another account](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [AcceptCertificateTransfer](#) in *AWS CLI Command Reference*.

add-thing-to-billing-group

The following code example shows how to use `add-thing-to-billing-group`.

AWS CLI

Example 1: To add a thing by name to a billing group

The following `add-thing-to-billing-group` example adds the thing named `MyLightBulb` to the billing group named `GroupOne`.

```
aws iot add-thing-to-billing-group \  
  --billing-group-name GroupOne \  
  --thing-name MyLightBulb
```

This command produces no output.

Example 2: To add a thing by ARN to a billing group

The following `add-thing-to-billing-group` example adds a thing with a specified ARN to a billing group with the specified ARN. Specifying an ARN is helpful if you work with multiple AWS Regions or accounts. It can help ensure that you are adding to the right Region and account.

```
aws iot add-thing-to-thing-group \  
  --billing-group-arn "arn:aws:iot:us-west-2:123456789012:billinggroup/GroupOne" \  
  --thing-arn "arn:aws:iot:us-west-2:123456789012:thing/MyOtherLightBulb"
```

This command produces no output.

For more information, see [Billing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [AddThingToBillingGroup](#) in *AWS CLI Command Reference*.

add-thing-to-thing-group

The following code example shows how to use `add-thing-to-thing-group`.

AWS CLI

To add a thing to a group

The following `add-thing-to-thing-group` example adds the specified thing to the specified thing group.

```
aws iot add-thing-to-thing-group \  
  --thing-name MyLightBulb \  
  --thing-group-name LightBulbs
```

This command produces no output.

For more information, see [Thing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [AddThingToThingGroup](#) in *AWS CLI Command Reference*.

associate-targets-with-job

The following code example shows how to use `associate-targets-with-job`.

AWS CLI

To associate a thing group with a continuous job

The following `associate-targets-with-job` example associates the specified thing group with the specified continuous job.

```
aws iot associate-targets-with-job \  
  --targets "arn:aws:iot:us-west-2:123456789012:thinggroup/LightBulbs" \  
  --job-id "example-job-04"
```

Output:

```
{
  "jobArn": "arn:aws:iot:us-west-2:123456789012:job/example-job-04",
  "jobId": "example-job-04",
  "description": "example continuous job"
}
```

For more information, see [Creating and Managing Jobs \(CLI\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [AssociateTargetsWithJob](#) in *AWS CLI Command Reference*.

attach-policy

The following code example shows how to use `attach-policy`.

AWS CLI

Example 1: To attach a policy to a thing group

The following `attach-policy` example attaches the specified policy to a thing group identified by its ARN.

```
aws iot attach-policy \
  --target "arn:aws:iot:us-west-2:123456789012:thinggroup/LightBulbs" \
  --policy-name "UpdateDeviceCertPolicy"
```

This command does not produce any output.

For more information, see [Thing Groups](#) in the *AWS IoT Developers Guide*.

Example 2: To attach a policy to a certificate

The following `attach-policy` example attaches the policy `UpdateDeviceCertPolicy` to the principal specified by a certificate.

```
aws iot attach-policy \
  --policy-name UpdateDeviceCertPolicy \
  --target "arn:aws:iot:us-
west-2:123456789012:cert/4f0ba725787aa94d67d2fca420eca022242532e8b3c58e7465c7778b443fd65e"
```

This command does not produce any output.

For more information, see [Attach an AWS IoT Policy to a Device Certificate](#) in the *AWS IoT Developers Guide*.

- For API details, see [AttachPolicy](#) in *AWS CLI Command Reference*.

attach-security-profile

The following code example shows how to use `attach-security-profile`.

AWS CLI

To associate a security profile with all unregistered devices

The following `attach-security-profile` example associates the AWS IoT Device Defender security profile named `Testprofile` with all unregistered devices in the `us-west-2` region for this AWS account.

```
aws iot attach-security-profile \  
  --security-profile-name Testprofile \  
  --security-profile-target-arn "arn:aws:iot:us-west-2:123456789012:all/  
unregistered-things"
```

This command produces no output.

For more information, see [Detect Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [AttachSecurityProfile](#) in *AWS CLI Command Reference*.

attach-thing-principal

The following code example shows how to use `attach-thing-principal`.

AWS CLI

To attach a certificate to your thing

The following `attach-thing-principal` example attaches a certificate to the `MyTemperatureSensor` thing. The certificate is identified by an ARN. You can find the ARN for a certificate in the AWS IoT console.

```
aws iot attach-thing-principal \  
  --certificate-arn arn:aws:iot:us-west-2:123456789012:certificate:  
MyCertificate
```

```
--thing-name MyTemperatureSensor \  
--principal arn:aws:iot:us-  
west-2:123456789012:cert/2e1eb273792174ec2b9bf4e9b37e6c6c692345499506002a35159767055278e8
```

This command produces no output.

For more information, see [How to Manage Things with the Registry](#) in the *AWS IoT Developers Guide*.

- For API details, see [AttachThingPrincipal](#) in *AWS CLI Command Reference*.

cancel-audit-mitigation-actions-task

The following code example shows how to use `cancel-audit-mitigation-actions-task`.

AWS CLI

To cancel an audit mitigation actions task

The following `cancel-audit-mitigation-actions-task` example cancels the application of mitigation actions for the specified task. You cannot cancel tasks that are already completed.

```
aws iot cancel-audit-mitigation-actions-task  
--task-id "myActionsTaskId"
```

This command produces no output.

For more information, see [CancelAuditMitigationActionsTask \(Mitigation Action Commands\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [CancelAuditMitigationActionsTask](#) in *AWS CLI Command Reference*.

cancel-audit-task

The following code example shows how to use `cancel-audit-task`.

AWS CLI

To cancel an audit task

The following `cancel-audit-task` example cancels an audit task with the specified task ID. You cannot cancel a task that is complete.

```
aws iot cancel-audit-task \  
  --task-id a3aea009955e501a31b764abe1bebd3d
```

This command produces no output.

For more information, see [Audit Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [CancelAuditTask](#) in *AWS CLI Command Reference*.

cancel-certificate-transfer

The following code example shows how to use `cancel-certificate-transfer`.

AWS CLI

To cancel the transfer a certificate to a different AWS account

The following `cancel-certificate-transfer` example cancels the transfer of the specified certificate transfer. The certificate is identified by a certificate ID. You can find the ID for a certificate in the AWS IoT console.

```
aws iot cancel-certificate-transfer \  
  --certificate-id  
  f0f33678c7c9a046e5cc87b2b1a58dfa0beec26db78add5e605d630e05c7fc8
```

This command produces no output.

For more information, see [Transfer a certificate to another account](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [CancelCertificateTransfer](#) in *AWS CLI Command Reference*.

cancel-job-execution

The following code example shows how to use `cancel-job-execution`.

AWS CLI

To cancel a job execution on a device

The following `cancel-job-execution` example cancels the execution of the specified job on a device. If the job is not in the QUEUED state, you must add the `--force` parameter.

```
aws iot cancel-job-execution \  
  --job-id "example-job-03" \  
  --thing-name "MyRPi"
```

This command produces no output.

For more information, see [Creating and Managing Jobs \(CLI\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [CancelJobExecution](#) in *AWS CLI Command Reference*.

cancel-job

The following code example shows how to use `cancel-job`.

AWS CLI

To cancel a job

The following `cancel-job` example cancels the specified job.

```
aws iot cancel-job \  
  --job-id "example-job-03"
```

Output:

```
{  
  "jobArn": "arn:aws:iot:us-west-2:123456789012:job/example-job-03",  
  "jobId": "example-job-03",  
  "description": "example job test"  
}
```

For more information, see [Creating and Managing Jobs \(CLI\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [CancelJob](#) in *AWS CLI Command Reference*.

clear-default-authorizer

The following code example shows how to use `clear-default-authorizer`.

AWS CLI

To clear the default authorizer

The following `clear-default-authorizer` example clears the currently configured default custom authorizer. After you run this command, there is no default authorizer. When you use a custom authorizer, you must specify it by name in the HTTP request headers.

```
aws iot clear-default-authorizer
```

This command produces no output.

For more information, see [ClearDefaultAuthorizer](#) in the *AWS IoT API Reference*.

- For API details, see [ClearDefaultAuthorizer](#) in *AWS CLI Command Reference*.

confirm-topic-rule-destination

The following code example shows how to use `confirm-topic-rule-destination`.

AWS CLI

To confirm a topic rule destination

The following `confirm-topic-rule-destination` example confirms a topic rule destination with a confirmation token received at an HTTP endpoint.

```
aws iot confirm-topic-rule-destination \  
  --confirmation-token "AYADeIcmtq-  
ZkxfpiWlQqHWM5ucAXwABABVhd3MtY3J5cHRvLXB1YmxpYy1rZXkAREFyY1E0Um1GeDg0V21BZWZ1VjZtZWFRVUJJUkt  
aywpPqg8YEsa11D4B40aJ2s1wEHKMybiF1Ro0ZzYisI0IvslzQY5UmCkqq3tV-3f7-  
nKfosgIAAAAADAAAEEAAAAAAAAAAAAAAAAAAAAAAi9RMgy-  
V19V9m6Iw2xfbw_____wAAAAEAAAAAAAAAAAAAAAAAAEAAAB1hw4SokgUcxiJ3gT06n50NLJVpzyQR1UmPIj5sShqXEQGcC  
iufgrzTePl8RZY0Wr006Aj9DiVzJZx-1iD6Pu-  
G6PUw1ka07Knzs2B4AD0qfrHUF4pYRTvyUgBnMGUCMQC8ZRmhKqntd_c6Kgrow3bMUDbvNqo2qZr8Z8Jm2rZgseR01An  
PIetJ803Z4I1I1F8xX1cdPGP-PV1d0XFemyL8g"
```

This command produces no output.

For more information, see [Confirming a topic rule destination](#) in the *AWS IoT Developer Guide*.

- For API details, see [ConfirmTopicRuleDestination](#) in *AWS CLI Command Reference*.

create-audit-suppression

The following code example shows how to use `create-audit-suppression`.

AWS CLI

To create an audit finding suppression

The following `create-audit-suppression` example creates an audit finding suppression for a policy named "virtualMachinePolicy" that has been flagged for being overly permissive.

```
aws iot create-audit-suppression \  
  --check-name IOT_POLICY_OVERLY_PERMISSIVE_CHECK \  
  --resource-identifier  
policyVersionIdentifier={"policyName"="virtualMachinePolicy","policyVersionId"="1"} \  
 \  
  --no-suppress-indefinitely \  
  --expiration-date 2020-10-20
```

This command produces no output.

For more information, see [Audit finding suppressions](#) in the *AWS IoT Developers Guide*.

- For API details, see [CreateAuditSuppression](#) in *AWS CLI Command Reference*.

create-authorizer

The following code example shows how to use `create-authorizer`.

AWS CLI

To create a custom authorizer

The following `create-authorizer` example creates a custom authorizer that uses the specified Lambda function as part of a custom authentication service.

```
aws iot create-authorizer \  
  --authorizer-name "CustomAuthorizer" \  
  --authorizer-function-arn "arn:aws:lambda:us-  
west-2:123456789012:function:CustomAuthorizerFunction" \  
  --token-key-name "MyAuthToken" \  
  --status ACTIVE \  
  --token-signing-public-keys FIRST_KEY="-----BEGIN PUBLIC KEY-----  
MIIBIjANBgkqhkiG9w0BAQEFAA0CAQ8AMIIBCgKCAQEA1uJOB4lQPgG/1M6ZfIwo  
Z+7ENxAio9q6QD4FFqjGZsvjtYwjoe1RKK0U8Eq9xb503kRSmyIwTzwm/f4Gf0Y  
ZUloJ+t3PUUwHrmbYTAgrCUgRFygjfgVwGCPs5ZAX4Eyqt5cr+AIHIiUDbxSa7p  
zw0BKPeic0asNJpqT8PkBbRaKyleJh5oo81NDHmVtbBm5A5YiJjqYXLaVAowKzZ
```

```
+GqsNvAQ9Jy1wI2VrEa10fL8f1DB/BJLm7zjpfPOHDJQgID0XnZwA1NnZc0hCwIx
50g2LW20y9R/dmqtDmJiVP97Z4GykxPvwlyHrUXY0iW1R3AR/Ac1NhCTGZMwVDB1
1QIDAQAB
-----END PUBLIC KEY-----"
```

Output:

```
{
  "authorizerName": "CustomAuthorizer",
  "authorizerArn": "arn:aws:iot:us-west-2:123456789012:authorizer/
CustomAuthorizer2"
}
```

For more information, see [CreateAuthorizer](#) in the *AWS IoT API Reference*.

- For API details, see [CreateAuthorizer](#) in *AWS CLI Command Reference*.

create-billing-group

The following code example shows how to use `create-billing-group`.

AWS CLI

To create a billing group

The following `create-billing-group` example creates a simple billing group named `GroupOne`.

```
aws iot create-billing-group \
  --billing-group-name GroupOne
```

Output:

```
{
  "billingGroupName": "GroupOne",
  "billingGroupArn": "arn:aws:iot:us-west-2:123456789012:billinggroup/GroupOne",
  "billingGroupId": "103de383-114b-4f51-8266-18f209ef5562"
}
```

For more information, see [Billing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [CreateBillingGroup](#) in *AWS CLI Command Reference*.

create-certificate-from-csr

The following code example shows how to use `create-certificate-from-csr`.

AWS CLI

To create a device certificate from a certificate signing request (CSR)

The following `create-certificate-from-csr` example creates a device certificate from a CSR. You can use the `openssl` command to create a CSR.

```
aws iot create-certificate-from-csr \  
  --certificate-signing-request=file://certificate.csr
```

Output:

```
{  
  "certificateArn": "arn:aws:iot:us-west-2:123456789012:cert/  
c0c57bbc8baaf4631a9a0345c957657f5e710473e3ddbbee1428d216d54d53ac9",  
  "certificateId":  
"c0c57bbc8baaf4631a9a0345c957657f5e710473e3ddbbee1428d216d54d53ac9",  
  "certificatePem": "<certificate-text>"  
}
```

For more information, see [CreateCertificateFromCSR](#) in the *AWS IoT API Reference*.

- For API details, see [CreateCertificateFromCsr](#) in *AWS CLI Command Reference*.

create-custom-metric

The following code example shows how to use `create-custom-metric`.

AWS CLI

To create a custom metric published by your devices to Device Defender

The following `create-custom-metric` example creates a custom metric that measures battery percentage.

```
aws iot create-custom-metric \  
  --metric-name "batteryPercentage" \  
  --metric-type "number" \  
  --metric-unit "Percentage"
```

```
--display-name "Remaining battery percentage." \  
--region us-east-1 \  
--client-request-token "02ccb92b-33e8-4dfa-a0c1-35b181ed26b0"
```

Output:

```
{  
  "metricName": "batteryPercentage",  
  "metricArn": "arn:aws:iot:us-east-1:1234564789012:custommetric/  
batteryPercentage"  
}
```

For more information, see [Custom metrics](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [CreateCustomMetric](#) in *AWS CLI Command Reference*.

create-dimension

The following code example shows how to use create-dimension.

AWS CLI

To create a dimension

The following create-dimension creates a dimension with a single topic filter called TopicFilterForAuthMessages.

```
aws iot create-dimension \  
  --name TopicFilterForAuthMessages \  
  --type TOPIC_FILTER \  
  --string-values device/+/auth
```

Output:

```
{  
  "name": "TopicFilterForAuthMessages",  
  "arn": "arn:aws:iot:eu-west-2:123456789012:dimension/TopicFilterForAuthMessages"  
}
```

For more information, see [Detect Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [CreateDimension](#) in *AWS CLI Command Reference*.

create-domain-configuration

The following code example shows how to use `create-domain-configuration`.

AWS CLI

To create a domain configuration

The following `create-domain-configuration` example creates an AWS-managed domain configuration with a service type of DATA.

```
aws iot create-domain-configuration \  
  --domain-configuration-name "additionalDataDomain" \  
  --service-type "DATA"
```

Output:

```
{  
  "domainConfigurationName": "additionalDataDomain",  
  "domainConfigurationArn": "arn:aws:iot:us-  
west-2:123456789012:domainconfiguration/additionalDataDomain/dikMh"  
}
```

For more information, see [Configurable Endpoints](#) in the *AWS IoT Developer Guide*.

- For API details, see [CreateDomainConfiguration](#) in *AWS CLI Command Reference*.

create-dynamic-thing-group

The following code example shows how to use `create-dynamic-thing-group`.

AWS CLI

To create a dynamic thing group

The following `create-dynamic-thing-group` example creates a dynamic thing group that contains any thing with a temperature attribute that is greater than 60 degrees. You must enable AWS IoT fleet indexing before you can use dynamic thing groups.

```
aws iot create-dynamic-thing-group \  
  --thing-group-name "RoomTooWarm" \  
  --query-string "attributes.temperature>60"
```

Output:

```
{
  "thingGroupName": "RoomTooWarm",
  "thingGroupArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/RoomTooWarm",
  "thingGroupId": "9d52492a-fc87-43f4-b6e2-e571d2ffcad1",
  "indexName": "AWS_Things",
  "queryString": "attributes.temperature>60",
  "queryVersion": "2017-09-30"
}
```

For more information, see [Dynamic Thing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [CreateDynamicThingGroup](#) in *AWS CLI Command Reference*.

create-job

The following code example shows how to use `create-job`.

AWS CLI**Example 1: To create a job**

The following `create-job` example creates a simple AWS IoT job that sends a JSON document to the MyRaspberryPi device.

```
aws iot create-job \
  --job-id "example-job-01" \
  --targets "arn:aws:iot:us-west-2:123456789012:thing/MyRaspberryPi" \
  --document file://example-job.json \
  --description "example job test" \
  --target-selection SNAPSHOT
```

Output:

```
{
  "jobArn": "arn:aws:iot:us-west-2:123456789012:job/example-job-01",
  "jobId": "example-job-01",
  "description": "example job test"
}
```

Example 2: To create a continuous job

The following `create-job` example creates a job that continues to run after the things specified as targets have completed the job. In this example, the target is a thing group, so when new devices are added to the group, the continuous job runs on those new things.

```
aws iot create-job --job-id "example-job-04" --targets "arn:aws:iot:us-west-2:123456789012:thinggroup/DeadBulbs" --document file://example-job.json --description "example continuous job" --target-selection CONTINUOUS
```

Output:

```
{
  "jobArn": "arn:aws:iot:us-west-2:123456789012:job/example-job-04",
  "jobId": "example-job-04",
  "description": "example continuous job"
}
```

For more information, see [Creating and Managing Jobs \(CLI\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [CreateJob](#) in *AWS CLI Command Reference*.

create-keys-and-certificate

The following code example shows how to use `create-keys-and-certificate`.

AWS CLI

To create an RSA key pair and issue an X.509 certificate

The following `create-keys-and-certificate` creates a 2048-bit RSA key pair and issues an X.509 certificate using the issued public key. Because this is the only time that AWS IoT provides the private key for this certificate, be sure to keep it in a secure location.

```
aws iot create-keys-and-certificate \
  --certificate-pem-outfile "myTest.cert.pem" \
  --public-key-outfile "myTest.public.key" \
  --private-key-outfile "myTest.private.key"
```

Output:

```
{
```



```

    "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/9894ba17925e663f1d29c23af4582b8e3b7619c31f3fbd93adcb51ae54b83dc2",
    "certificateId":
    "9894ba17925e663f1d29c23af4582b8e3b7619c31f3fbd93adcb51ae54b83dc2",
    "certificatePem": "
-----BEGIN CERTIFICATE-----
MIICiTCCEXAMPLE6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMx CzAJBgNVBAgEXAMPLEAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
b24x FDA SBgNVBA sTC01BTSEXAMPLE2x1MRIwEAYDVQQDEw1UZ XN0Q21sYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb251QGFTYEXAMPLEb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCEXAMPLEJBgNVBAgTAldBMR AwDgYD
VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24x FDAEXAMPLEsTC01BTSBDb25z
b2x1MRIwEAYDVQQDEw1UZ XN0Q21sYWMxHzAdBgkqhkiG9w0BCQEXAMPLE251QGFT
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+aEXAMPLE
EXAMPLEfEvySwTc2XADZ4nB+BLyGVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
rDHudUZEXAMPLEELG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzSwY6786m86gpE
Ibb30hjZncvQAEXAMPLEEWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVvxYUntneD9+h8Mg9qEXAMPLEEyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFbjvSfpJI1J00zbhNYS5f6GuoEDEXAMPLEBHjJnyp3780D8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----\n",
    "keyPair": {
        "PublicKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQ0CAQ8AMIIBCGKCAQEAEXAMPLE1nnyJwKSMHw4h\nMMEXAMPLEEuuN/
dMAS3fyce8DW/4+EXAMPLEYjmoF/YVF/gHr99VEEXAMPLE5VF13\n59VK7cEXAMPLE67GK+y+jikqX0gHh/
xJTtwo
+sGpWEXAMPLEDz18x0d2ka4tCzuWEXAMPLEEahJbYkCPUBSU8opVkr7qkEXAMPLE1DR6sx2Hocli00Ltu6Fkw91swQWEX
\GB3ZPrNh0PzQYvjUSTzecyNCx2EXAMPLEv9mQ0UXP6p1fgxwKRX2fEXAMPLEDa
\nhJLXkX3rHU2xbxJSq7D+XEXAMPLEcw+LyFhI5mgFR188eGdsAEXAMPLE1nI9EesG\nFQIDAQAB\n-----
END PUBLIC KEY-----\n",
        "PrivateKey": "-----BEGIN RSA PRIVATE KEY-----\nkey omitted for security
reasons\n-----END RSA PRIVATE KEY-----\n"
    }
}

```

For more information, see [Create and Register an AWS IoT Device Certificate](#) in the **AWS IoT Developer Guide**.

- For API details, see [CreateKeysAndCertificate](#) in *AWS CLI Command Reference*.

create-mitigation-action

The following code example shows how to use create-mitigation-action.

AWS CLI

To create a mitigation action

The following `create-mitigation-action` example defines a mitigation action named `AddThingsToQuarantineGroup1Action` that, when applied, moves things into the thing group named `QuarantineGroup1`. This action overrides dynamic thing groups.

```
aws iot create-mitigation-action --cli-input-json file::params.json
```

Contents of `params.json`:

```
{
  "actionName": "AddThingsToQuarantineGroup1Action",
  "actionParams": {
    "addThingsToThingGroupParams": {
      "thingGroupNames": [
        "QuarantineGroup1"
      ],
      "overrideDynamicGroups": true
    }
  },
  "roleArn": "arn:aws:iam::123456789012:role/service-role/MoveThingsToQuarantineGroupRole"
}
```

Output:

```
{
  "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/AddThingsToQuarantineGroup1Action",
  "actionId": "992e9a63-a899-439a-aa50-4e20c52367e1"
}
```

For more information, see [CreateMitigationAction \(Mitigation Action Commands\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [CreateMitigationAction](#) in *AWS CLI Command Reference*.

`create-ota-update`

The following code example shows how to use `create-ota-update`.

AWS CLI

To create an OTA update for use with Amazon FreeRTOS

The following `create-ota-update` example creates an AWS IoT OTAUpdate on a target group of things or groups. This is part of an Amazon FreeRTOS over-the-air update which makes it possible for you to deploy new firmware images to a single device or a group of devices.

```
aws iot create-ota-update \  
  --cli-input-json file://create-ota-update.json
```

Contents of `create-ota-update.json`:

```
{  
  "otaUpdateId": "ota12345",  
  "description": "A critical update needed right away.",  
  "targets": [  
    "device1",  
    "device2",  
    "device3",  
    "device4"  
  ],  
  "targetSelection": "SNAPSHOT",  
  "awsJobExecutionsRolloutConfig": {  
    "maximumPerMinute": 10  
  },  
  "files": [  
    {  
      "fileName": "firmware.bin",  
      "fileLocation": {  
        "stream": {  
          "streamId": "004",  
          "fileId": 123  
        }  
      },  
      "codeSigning": {  
        "awsSignerJobId": "48c67f3c-63bb-4f92-a98a-4ee0fbc2bef6"  
      }  
    }  
  ]  
  "roleArn": "arn:aws:iam:123456789012:role/service-role/my_ota_role"  
}
```

Output:

```
{
  "otaUpdateId": "ota12345",
  "awsIotJobId": "job54321",
  "otaUpdateArn": "arn:aws:iot:us-west-2:123456789012:otaupdate/itsaupdate",
  "awsIotJobArn": "arn:aws:iot:us-west-2:123456789012:job/itsajob",
  "otaUpdateStatus": "CREATE_IN_PROGRESS"
}
```

For more information, see [CreateOTAUpdate](#) in the *AWS IoT API Reference*.

- For API details, see [CreateOtaUpdate](#) in *AWS CLI Command Reference*.

create-policy-version

The following code example shows how to use `create-policy-version`.

AWS CLI**To update a policy with a new version**

The following `create-policy-version` example updates a policy definition, creating a new policy version. This example also makes the new version the default.

```
aws iot create-policy-version \
  --policy-name UpdateDeviceCertPolicy \
  --policy-document file://policy.json \
  --set-as-default
```

Contents of `policy.json`:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iot:UpdateCertificate",
      "Resource": "*"
    }
  ]
}
```

Output:

```
{
  "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/UpdateDeviceCertPolicy",
  "policyDocument": "{ \"Version\": \"2012-10-17\", \"Statement\": [ { \"Effect\": \"Allow\", \"Action\": \"iot:UpdateCertificate\", \"Resource\": \"*\" } ] }",
  "policyVersionId": "2",
  "isDefaultVersion": true
}
```

For more information, see [AWS IoT Policies](#) in the *AWS IoT Developers Guide*.

- For API details, see [CreatePolicyVersion](#) in *AWS CLI Command Reference*.

create-policy

The following code example shows how to use `create-policy`.

AWS CLI**To create an AWS IoT policy**

The following `create-policy` example creates an AWS IoT policy named `TemperatureSensorPolicy`. The `policy.json` file contains statements that allow AWS IoT policy actions.

```
aws iot create-policy \
  --policy-name TemperatureSensorPolicy \
  --policy-document file://policy.json
```

Contents of `policy.json`:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iot:Publish",
        "iot:Receive"
      ],
      "Resource": [
```

```

        "arn:aws:iot:us-west-2:123456789012:topic/topic_1",
        "arn:aws:iot:us-west-2:123456789012:topic/topic_2"
    ]
},
{
    "Effect": "Allow",
    "Action": [
        "iot:Subscribe"
    ],
    "Resource": [
        "arn:aws:iot:us-west-2:123456789012:topicfilter/topic_1",
        "arn:aws:iot:us-west-2:123456789012:topicfilter/topic_2"
    ]
},
{
    "Effect": "Allow",
    "Action": [
        "iot:Connect"
    ],
    "Resource": [
        "arn:aws:iot:us-west-2:123456789012:client/basicPubSub"
    ]
}
]
}

```

Output:

```

{
    "policyName": "TemperatureSensorPolicy",
    "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/
TemperatureSensorPolicy",
    "policyDocument": "{
        \"Version\": \"2012-10-17\",
        \"Statement\": [
            {
                \"Effect\": \"Allow\",
                \"Action\": [
                    \"iot:Publish\",
                    \"iot:Receive\"
                ],
                \"Resource\": [
                    \"arn:aws:iot:us-west-2:123456789012:topic/topic_1\",

```

```

        \ "arn:aws:iot:us-west-2:123456789012:topic/topic_2\"
      ]
    },
    {
      \ "Effect\" : \ "Allow\" ,
      \ "Action\" : [
        \ "iot:Subscribe\"
      ],
      \ "Resource\" : [
        \ "arn:aws:iot:us-west-2:123456789012:topicfilter/topic_1\" ,
        \ "arn:aws:iot:us-west-2:123456789012:topicfilter/topic_2\"
      ]
    },
    {
      \ "Effect\" : \ "Allow\" ,
      \ "Action\" : [
        \ "iot:Connect\"
      ],
      \ "Resource\" : [
        \ "arn:aws:iot:us-west-2:123456789012:client/basicPubSub\"
      ]
    }
  ]
},
"policyVersionId": "1"
}

```

For more information, see [AWS IoT Policies](#) in the *AWS IoT Developers Guide*.

- For API details, see [CreatePolicy](#) in *AWS CLI Command Reference*.

create-provisioning-claim

The following code example shows how to use `create-provisioning-claim`.

AWS CLI

To create a provisioning claim

The following `create-provisioning-claim` example creates a provisioning claim from a provisioning template.

```
aws iot create-provisioning-claim \
```

```
--template-name MyTestProvisioningTemplate
```

Output:

```
{
  "certificateId":
    "78de02184b2ce80cf8fb709bda59e62b19fb83513590483eb0434589476ab09f",
  "certificatePem": "-----BEGIN CERTIFICATE-----\nMIIDdzCCA1
+gAwIBAgIUXSZHEBLztMLZ2fHG
14gV0NymYY0wDQYJKoZIhvcNAQEL
\nBQAwfjELMAKGA1UEBhMCMVVMxEzARBgNVBAGMCl dhc2hpbmd0b24xEDA0Bg
VBAcM\nB1N1YXR0bGUxGDAwBgNVBAoMD0FtYXpvbi5jb20gSW5jLjEgMB4GA1UECwwXQW1h
\nem9uIElvVCBQcm9
2aXNpb25pbmcxDDAKBgNVBAUTAzEuMDAeFw0yMDA3MjgxNjQ0\nMDZaFw0yMDA3MjgxNjUxMDZaMEsxBHBBGgNVB
AMMQDFhNDEyM2VkNmIxYjU3MzE3\nZTgzMTJmY2MzN2FiNTdhY2MzYTZkZGVjOGQ5OGY3NzUwMWR1Mjc0YjhmYTQ
xN2Iw\nnggEiMA0GCSqGSIb3EXAMPLEAA4IBDwAwggEKAoIBAQBDBhKI94ktKLqTwnj+ayOq1\nTAJt/
N6s6IJDZv1
rYjkC0E7wzaeY3TprWk03S29vUzVuE0XHXQXZbihgpg2m6fza\nkwm9/
wpjzE9ny5+xkPGVH4Wnwz7yK5m8S0agL
T96cRBSWnWmon0WdY0GKVzni0CA\n+iyGudgrFKm7Eae/
v18oXrf82Kt0AG04xG0KE2WKYHsT1fx3c9xZh1XP/eX
Lhv00\n+1Gp0WVw9PbhKfrxliKJ5q6sL5nVUAUhq6h1QPYwsATe0vAp3u0ak5zgTyL0fg7Y
\nPyKk6VYwLW62r+v
YBSForEM0Ahkq3LsP/rjxpEKmi2W41PVS6oFZRKcD+H1Kyil5\nAgMBAAGjIDAEAwGA1UdEwEB/
wQCMAAwDgYDV
R0PAQH/BAQDAgeAMA0GCSqGSIb3\nDQEBChUA4IBAQAQgix2k6nVqbZFKq97/fZBzLGS0dyz5rT/
E41cDIRX+1j
EPW41\nnw0D+2sXheCZLZZnSkvIiP74IToNeXDrjdcaodeGFVHIE1RjhMIq+4ZebPbRLtidF
\nRc2hfcTAlqq9Z6v
5Vk6BeM1tu0RqH1wPoVUccLPya8EjNCbnJZUmGd0frN/Y9pho\n5ikV+HPeZhG/k6dhE2GsQJyKfVHL/
uBgKSily
1bRyWU1r6qcpWBNBHjUoD7Hg0wD
\nnzMh4XRb2FQDsqaFalkCSYmeL8IVC49sgPD90typ5uteGMTy62usAAUQdq/f
ZvrWg\n0kFpwMVnGKVKT7Kg0kKOLzKWOBB2Jm4/gmrJ\n-----END CERTIFICATE-----\n",
  "keyPair": {
    "PublicKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCg
KCAQEAWYSiPeJLSi6k8J4/msjq
\nUwCbfer0iCQ2b5a2I5AtB08M2nmN06a1pNN0tvb1M1bhDlx10F2W4oYKYN
pun8\n2pFpvf8KY8xPZ8ufsZDx1R+Fp8M+8iuZvEtGoC0/enEQUl1pqJzlnWNBilc54tA
\nngPoshrnYKxSpuxGn
v79fKF63/NirTgBjuMRtChNlimEXAMPLE3PcWYZVz/3ly4b9\nNPPRqdFlcPT24Sn68ZYiieaurC
+Z1VGlB6uoZU
```



```

D2MLAE3jrwKd7tGp0c4E8i9H40\n2D8ip0lWMC1utq/
lWAUhaKxDDgIZKty7D/648aRCpotluJT1UuqBWUSnA/h9
Ssop\neQIDAQAB\n-----END PUBLIC KEY-----\n",
    "PrivateKey": "-----BEGIN RSA PRIVATE KEY-----
\nMIIIEowIBAAKCAQEAwYSiPeJLSi6k8J4/
msjqtUwCbfzer0iCQ2b5a2I5AtB08M2n
\nmN06a1pNN0tvb1M1bhD1x10F2W4oYKYnpun82pFpvf8KY8xPZ8ufsZ
Dx1R+Fp8M+\n8iuZvEtGoC0/enEQUlP1pqJzlnWNBilc54tAgPoshrnYKxSpuxGnv79fKF63/Nir
\nTgBjuMRtCh
NlimB7E9X8d3PcWYZVz/3ly4b9NPpRqdFlcPT24Sn68ZYiieaurC+Z
\n1VG1B6uoZUD2MLAE3jrwKd7tGp0c4E8i
9H402D8ip0lWMC1utq/lWAUhaKxDDgIZ\nKty7D/648aRCpotluJT1UuqBWUSnA/
h9SsopeQIDAQABAoIBAEAybn
QUtx9T2/nK\nTzT2pA4iugecxI4dz+DmT0XVxs5VJmrx/
nBSq6ejXExEpSIM04RY7LE3ZdJcnd56\nF7tQkkY7yR
VzfxHeXFU1kr0IPuxWebN0rRoPZr+1RSer+ww2aBC525+88pVuR6tM
\nm3pgkrR2ycCj9Fd0UoQxdjHBHaM5PDMj
9aSxCKdg3nReepeGwsR2TQA+m2vVxWk7\nnou0+91eTOP+/QfP7P8Zj0Ik02Xiv1RcVDyN/
E4QXPKuIkM/8vS8VK+
E9pATQ0MtB\n2lW8R/YU5AJd6j1EXAMPLEGU2UzRzInNWlLtkPPPqgqXXhx0f+mxByjcMa1VJk0L
\nh0G2R0UCgY
EA+R0cHNHy/XbsP7Fih0hEh+6Q2QxQ2ncBUPYbBazrR8Hn+7SCICQK
\nVyYfd8Ajfq3e7RsKVL5S1MBp7S1idxak
bIn28fKfPn62DaemGCIOyDgLf+eUxBx
\nngzbCiBZga8brfurza43UZjKZLpg3hq721+FeAiXi1Nma4Yr9YWEHEN
8CgYEAXuWt\nnpzdWwmsiFzfsAw0sy9ySDA/xr5WRWzJyAqUsjsks6rxNzWebpufnYHcmtW7pLdqM
\nkboHwN2pXa
kmZvrk2nKkEMq5brBYGDxuxDe+V369Bianx8aZFYIsckA70wXW1w1h
\nngRC5rQ4X0gp3+Jmw7eA08LRYDjaN846+
Qbt02KcCgYAWS0UL51bijQR0ZwI0dz27\nnFQVuCAYsp748aurcRTACCj8jbnK/
QbqTN1xWsaH7ssBjZKo2D5sAqY
BRtASW0Dab\nnAHxsDhVm2Jye+ESLoHMaClOyCkT31l8yqXIcEDStM07f01Ryag164EiJvSIRmfny\nnNL/
fXVjCSH
/udCxdzPt+7QKBgQC+LAD7rxdr4J9538hTqpc4XK9vxRbrMXEH55XH
\nHbMa2x0NZXpmeTgEQBukyohCVceyRhK9
i0e6irZTjVXgh0eoTpC8VXkzcnczouTiQ
\nnFQQSGfnp7Ioe6UIz23715pKduzSNkMSKrG924ktv7CyDBF1gBQI5g
aDoHndJBj\nnPRtIZQKBgA8MASxtTxQntRwXXzR92U0vAighiuRkB/mx9jQpUcK1qiqHbkAMqgNF
\nPFCBYIubFT
iYKKKeJNbyJQvjfsJcKAnaFJ+RnTxk0Q6Wjm20peJ/ii4QiDdnigoE\nnvd1c5cFQewWb4/
zqAtPdinkPlN94ileI
79XQdc7RlJ0jpgTimL+V\n-----END RSA PRIVATE KEY-----\n"
    },
    "expiration": 1595955066.0

```

```
}
```

For more information, see [Provisioning by trusted user](#) in the *AWS IoT Core Developers Guide*.

- For API details, see [CreateProvisioningClaim](#) in *AWS CLI Command Reference*.

create-provisioning-template-version

The following code example shows how to use `create-provisioning-template-version`.

AWS CLI

To create a provisioning template version

The following example creates a version for the specified provisioning template. The body of the new version is supplied in the file `template.json`.

```
aws iot create-provisioning-template-version \  
  --template-name widget-template \  
  --template-body file://template.json
```

Contents of `template.json`:

```
{  
  "Parameters" : {  
    "DeviceLocation": {  
      "Type": "String"  
    }  
  },  
  "Mappings": {  
    "LocationTable": {  
      "Seattle": {  
        "LocationUrl": "https://example.aws"  
      }  
    }  
  },  
  "Resources" : {  
    "thing" : {  
      "Type" : "AWS::IoT::Thing",  
      "Properties" : {  
        "AttributePayload" : {  
          "version" : "v1",  
          "serialNumber" : "serialNumber"  
        }  
      }  
    }  
  }  
}
```

```

        },
        "ThingName" : {"Fn::Join":["",["ThingPrefix_",
{"Ref":"SerialNumber"}]]},
        "ThingTypeName" : {"Fn::Join":["",["ThingTypePrefix_",
{"Ref":"SerialNumber"}]]},
        "ThingGroups" : ["widgets", "WA"],
        "BillingGroup": "BillingGroup"
    },
    "OverrideSettings" : {
        "AttributePayload" : "MERGE",
        "ThingTypeName" : "REPLACE",
        "ThingGroups" : "DO_NOTHING"
    }
},
"certificate" : {
    "Type" : "AWS::IoT::Certificate",
    "Properties" : {
        "CertificateId": {"Ref": "AWS::IoT::Certificate::Id"},
        "Status" : "Active"
    }
},
"policy" : {
    "Type" : "AWS::IoT::Policy",
    "Properties" : {
        "PolicyDocument" : {
            "Version": "2012-10-17",
            "Statement": [{
                "Effect": "Allow",
                "Action":["iot:Publish"],
                "Resource": ["arn:aws:iot:us-east-1:123456789012:topic/foo/
bar"]
            }]
        }
    }
},
"DeviceConfiguration": {
    "FallbackUrl": "https://www.example.com/test-site",
    "LocationUrl": {
        "Fn::FindInMap": ["LocationTable",{"Ref": "DeviceLocation"},
"LocationUrl"]}
    }
}
}

```

```
}
```

Output:

```
{
  "templateArn": "arn:aws:iot:us-east-1:123456789012:provisioningtemplate/widget-
template",
  "templateName": "widget-template",
  "versionId": 2,
  "isDefaultVersion": false
}
```

For more information, see [AWS IoT Secure Tunneling](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [CreateProvisioningTemplateVersion](#) in *AWS CLI Command Reference*.

create-provisioning-template

The following code example shows how to use `create-provisioning-template`.

AWS CLI**To create a provisioning template**

The following `create-provisioning-template` example creates a provisioning template as defined by the file `template.json`.

```
aws iot create-provisioning-template \
  --template-name widget-template \
  --description "A provisioning template for widgets" \
  --provisioning-role-arn arn:aws:iam::123456789012:role/Provision_role \
  --template-body file://template.json
```

Contents of `template.json`:

```
{
  "Parameters" : {
    "DeviceLocation": {
      "Type": "String"
    }
  },
  "Mappings": {
```

```

    "LocationTable": {
      "Seattle": {
        "LocationUrl": "https://example.aws"
      }
    },
    "Resources" : {
      "thing" : {
        "Type" : "AWS::IoT::Thing",
        "Properties" : {
          "AttributePayload" : {
            "version" : "v1",
            "serialNumber" : "serialNumber"
          },
          "ThingName" : {"Fn::Join":["",["ThingPrefix_",
{"Ref":"SerialNumber"}]]},
          "ThingTypeName" : {"Fn::Join":["",["ThingTypePrefix_",
{"Ref":"SerialNumber"}]]},
          "ThingGroups" : ["widgets", "WA"],
          "BillingGroup": "BillingGroup"
        },
        "OverrideSettings" : {
          "AttributePayload" : "MERGE",
          "ThingTypeName" : "REPLACE",
          "ThingGroups" : "DO_NOTHING"
        }
      },
      "certificate" : {
        "Type" : "AWS::IoT::Certificate",
        "Properties" : {
          "CertificateId": {"Ref": "AWS::IoT::Certificate::Id"},
          "Status" : "Active"
        }
      },
      "policy" : {
        "Type" : "AWS::IoT::Policy",
        "Properties" : {
          "PolicyDocument" : {
            "Version": "2012-10-17",
            "Statement": [{
              "Effect": "Allow",
              "Action":["iot:Publish"],
              "Resource": ["arn:aws:iot:us-east-1:504350838278:topic/foo/
bar"]
            }
          ]
        }
      }
    }
  }

```

```

    }
  }
},
"DeviceConfiguration": {
  "FallbackUrl": "https://www.example.com/test-site",
  "LocationUrl": {
    "Fn::FindInMap": ["LocationTable", {"Ref": "DeviceLocation"},
"LocationUrl"]}
  }
}
}
}

```

Output:

```

{
  "templateArn": "arn:aws:iot:us-east-1:123456789012:provisioningtemplate/widget-
template",
  "templateName": "widget-template",
  "defaultVersionId": 1
}

```

For more information, see [AWS IoT Secure Tunneling](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [CreateProvisioningTemplate](#) in *AWS CLI Command Reference*.

create-role-alias

The following code example shows how to use `create-role-alias`.

AWS CLI**To create a role alias**

The following `create-role-alias` example creates a role alias called `LightBulbRole` for the specified role.

```

aws iot create-role-alias \
  --role-alias LightBulbRole \
  --role-arn arn:aws:iam::123456789012:role/lightbulbrole-001

```

Output:

```
{
  "roleAlias": "LightBulbRole",
  "roleAliasArn": "arn:aws:iot:us-west-2:123456789012:rolealias/LightBulbRole"
}
```

For more information, see [CreateRoleAlias](#) in the *AWS IoT API Reference*.

- For API details, see [CreateRoleAlias](#) in *AWS CLI Command Reference*.

create-scheduled-audit

The following code example shows how to use `create-scheduled-audit`.

AWS CLI

To create a scheduled audit

The following `create-scheduled-audit` example creates a scheduled audit that runs weekly, on Wednesday, to check if CA certificates or device certificates are expiring.

```
aws iot create-scheduled-audit \
  --scheduled-audit-name WednesdayCertCheck \
  --frequency WEEKLY \
  --day-of-week WED \
  --target-check-names CA_CERTIFICATE_EXPIRING_CHECK
  DEVICE_CERTIFICATE_EXPIRING_CHECK
```

Output:

```
{
  "scheduledAuditArn": "arn:aws:iot:us-west-2:123456789012:scheduledaudit/
  WednesdayCertCheck"
}
```

For more information, see [Audit Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [CreateScheduledAudit](#) in *AWS CLI Command Reference*.

create-security-profile

The following code example shows how to use `create-security-profile`.

AWS CLI

To create a security profile

The following `create-security-profile` example creates a security profile that checks if cellular bandwidth exceeds a threshold or if more than 10 authorization failures occur within a five-minute period.

```
aws iot create-security-profile \
  --security-profile-name PossibleIssue \
  --security-profile-description "Check to see if authorization fails 10 times in
  5 minutes or if cellular bandwidth exceeds 128" \
  --behaviors "[{"name":"CellularBandwidth","metric":"aws:message-byte-size
  \","criteria":{"comparisonOperator":"greater-than","value":{"count":128},
  "consecutiveDatapointsToAlarm":1,"consecutiveDatapointsToClear":1}},{"name
  \":"Authorization","metric":"aws:num-authorization-failures","criteria":
  {"comparisonOperator":"less-than","value":{"count":10},"durationSeconds
  \":300,"consecutiveDatapointsToAlarm":1,"consecutiveDatapointsToClear":1}]]"
```

Output:

```
{
  "securityProfileName": "PossibleIssue",
  "securityProfileArn": "arn:aws:iot:us-west-2:123456789012:securityprofile/
  PossibleIssue"
}
```

For more information, see [Detect Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [CreateSecurityProfile](#) in *AWS CLI Command Reference*.

create-stream

The following code example shows how to use `create-stream`.

AWS CLI

To create a stream for delivering one or more large files in chunks over MQTT

The following `create-stream` example creates a stream for delivering one or more large files in chunks over MQTT. A stream transports data bytes in chunks or blocks packaged as MQTT messages from a source like S3. You can have one or more files associated with a stream.


```
aws iot create-stream \  
  --cli-input-json file://create-stream.json
```

Contents of create-stream.json:

```
{  
  "streamId": "stream12345",  
  "description": "This stream is used for Amazon FreeRTOS OTA Update 12345.",  
  "files": [  
    {  
      "fileId": 123,  
      "s3Location": {  
        "bucket": "codesign-ota-bucket",  
        "key": "48c67f3c-63bb-4f92-a98a-4ee0fbc2bef6"  
      }  
    }  
  ],  
  "roleArn": "arn:aws:iam:123456789012:role/service-role/my_ota_stream_role"  
}
```

Output:

```
{  
  "streamId": "stream12345",  
  "streamArn": "arn:aws:iot:us-west-2:123456789012:stream/stream12345",  
  "description": "This stream is used for Amazon FreeRTOS OTA Update 12345.",  
  "streamVersion": "1"  
}
```

For more information, see [CreateStream](#) in the *AWS IoT API Reference*.

- For API details, see [CreateStream](#) in *AWS CLI Command Reference*.

create-thing-group

The following code example shows how to use create-thing-group.

AWS CLI

Example 1: To create a thing group

The following `create-thing-group` example creates a thing group named `LightBulbs` with a description and two attributes.

```
aws iot create-thing-group \  
  --thing-group-name LightBulbs \  
  --thing-group-properties "thingGroupDescription=\"Generic bulb group\  
attributePayload={attributes={Manufacturer=AnyCompany,wattage=60}}"
```

Output:

```
{  
  "thingGroupName": "LightBulbs",  
  "thingGroupArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/LightBulbs",  
  "thingGroupId": "9198bf9f-1e76-4a88-8e8c-e7140142c331"  
}
```

Example 2: To create a thing group that's part of a parent group

The following `create-thing-group` creates a thing group named `HalogenBulbs` that has a parent thing group named `LightBulbs`.

```
aws iot create-thing-group \  
  --thing-group-name HalogenBulbs \  
  --parent-group-name LightBulbs
```

Output:

```
{  
  "thingGroupName": "HalogenBulbs",  
  "thingGroupArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/HalogenBulbs",  
  "thingGroupId": "f4ec6b84-b42b-499d-9ce1-4dbd4d4f6f6e"  
}
```

For more information, see [Thing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [CreateThingGroup](#) in *AWS CLI Command Reference*.

create-thing-type

The following code example shows how to use `create-thing-type`.

AWS CLI

To define a thing type

The following `create-thing-type` example defines a thing type and associated attributes.

```
aws iot create-thing-type \  
  --thing-type-name "LightBulb" \  
  --thing-type-properties "thingTypeDescription=light bulb type,  
searchableAttributes=wattage,model"
```

Output:

```
{  
  "thingTypeName": "LightBulb",  
  "thingTypeArn": "arn:aws:iot:us-west-2:123456789012:thingtype/LightBulb",  
  "thingTypeId": "ce3573b0-0a3c-45a7-ac93-4e0ce14cd190"  
}
```

For more information, see [Thing Types](#) in the *AWS IoT Developers Guide*.

- For API details, see [CreateThingType](#) in *AWS CLI Command Reference*.

create-thing

The following code example shows how to use `create-thing`.

AWS CLI

Example 1: To create a thing record in the registry

The following `create-thing` example creates an entry for a device in the AWS IoT thing registry.

```
aws iot create-thing \  
  --thing-name SampleIoTThing
```

Output:

```
{  
  "thingName": "SampleIoTThing",
```

```
"thingArn": "arn:aws:iot:us-west-2: 123456789012:thing/SampleIoTThing",
"thingId": " EXAMPLE1-90ab-cdef-fedc-ba987EXAMPLE "
}
```

Example 2: To define a thing that is associated with a thing type

The following `create-thing` example create a thing that has the specified thing type and its attributes.

```
aws iot create-thing \
  --thing-name "MyLightBulb" \
  --thing-type-name "LightBulb" \
  --attribute-payload '{"attributes": {"wattage": "75", "model": "123"}}'
```

Output:

```
{
  "thingName": "MyLightBulb",
  "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MyLightBulb",
  "thingId": "40da2e73-c6af-406e-b415-15acae538797"
}
```

For more information, see [How to Manage Things with the Registry](#) and [Thing Types](#) in the *AWS IoT Developers Guide*.

- For API details, see [CreateThing](#) in *AWS CLI Command Reference*.

create-topic-rule-destination

The following code example shows how to use `create-topic-rule-destination`.

AWS CLI

To create a topic rule destination

The following `create-topic-rule-destination` example creates a topic rule destination for an HTTP endpoint.

```
aws iot create-topic-rule-destination \
  --destination-configuration httpUrlConfiguration={confirmationUrl=https://
example.com}
```

Output:

```
{
  "topicRuleDestination": {
    "arn": "arn:aws:iot:us-west-2:123456789012:ruledestination/http/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "status": "IN_PROGRESS",
    "statusReason": "Awaiting confirmation. Confirmation message sent on
2020-07-09T22:47:54.154Z; no response received from the endpoint.",
    "httpUrlProperties": {
      "confirmationUrl": "https://example.com"
    }
  }
}
```

For more information, see [Creating a topic rule destination](#) in the *AWS IoT Developer Guide*.

- For API details, see [CreateTopicRuleDestination](#) in *AWS CLI Command Reference*.

create-topic-rule

The following code example shows how to use `create-topic-rule`.

AWS CLI**To create a rule that sends an Amazon SNS alert**

The following `create-topic-rule` example creates a rule that sends an Amazon SNS message when soil moisture level readings, as found in a device shadow, are low.

```
aws iot create-topic-rule \
  --rule-name "LowMoistureRule" \
  --topic-rule-payload file://plant-rule.json
```

The example requires the following JSON code to be saved to a file named `plant-rule.json`:

```
{
  "sql": "SELECT * FROM '$aws/things/MyRPi/shadow/update/accepted' WHERE
state.reported.moisture = 'low'\n",
  "description": "Sends an alert whenever soil moisture level readings are too
low.",
  "ruleDisabled": false,
```

```
"awsIotSqlVersion": "2016-03-23",
"actions": [{
  "sns": {
    "targetArn": "arn:aws:sns:us-
west-2:123456789012:MyRPIlowMoistureTopic",
    "roleArn": "arn:aws:iam::123456789012:role/service-role/
MyRPIlowMoistureTopicRole",
    "messageFormat": "RAW"
  }
}]
}
```

This command produces no output.

For more information, see [Creating an AWS IoT Rule](#) in the *AWS IoT Developers Guide*.

- For API details, see [CreateTopicRule](#) in *AWS CLI Command Reference*.

delete-account-audit-configuration

The following code example shows how to use `delete-account-audit-configuration`.

AWS CLI

To disable all audit checks for your AWS account

The following `delete-account-audit-configuration` example restores the default settings for AWS IoT Device Defender for this account, disabling all audit checks and clearing configuration data. It also deletes any scheduled audits for this account. **Use this command with caution.**

```
aws iot delete-account-audit-configuration \
  --delete-scheduled-audits
```

This command produces no output.

For more information, see [Audit Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [DeleteAccountAuditConfiguration](#) in *AWS CLI Command Reference*.

delete-audit-suppression

The following code example shows how to use `delete-audit-suppression`.

AWS CLI

To delete an audit finding suppression

The following `delete-audit-suppression` example deletes an audit finding suppression for `DEVICE_CERTIFICATE_EXPIRING_CHECK`.

```
aws iot delete-audit-suppression \  
  --check-name DEVICE_CERTIFICATE_EXPIRING_CHECK \  
  --resource-identifier deviceCertificateId="c7691e<shortened>"
```

This command produces no output.

For more information, see [Audit finding suppressions](#) in the *AWS IoT Developers Guide*.

- For API details, see [DeleteAuditSuppression](#) in *AWS CLI Command Reference*.

delete-authorizer

The following code example shows how to use `delete-authorizer`.

AWS CLI

To delete a custom authorizer

The following `delete-authorizer` example deletes the authorizer named `CustomAuthorizer`. A custom authorizer must be in the `INACTIVE` state before you can delete it.

```
aws iot delete-authorizer \  
  --authorizer-name CustomAuthorizer
```

This command produces no output.

For more information, see [DeleteAuthorizer](#) in the *AWS IoT Developer Guide*.

- For API details, see [DeleteAuthorizer](#) in *AWS CLI Command Reference*.

delete-billing-group

The following code example shows how to use `delete-billing-group`.

AWS CLI

To delete a billing group

The following `delete-billing-group` example deletes the specified billing group. You can delete a billing group even if it contains one or more things.

```
aws iot delete-billing-group \  
  --billing-group-name BillingGroupTwo
```

This command does not produce any output.

For more information, see [Billing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [DeleteBillingGroup](#) in *AWS CLI Command Reference*.

`delete-ca-certificate`

The following code example shows how to use `delete-ca-certificate`.

AWS CLI

To delete a CA certificate

The following `delete-ca-certificate` example deletes the CA certificate with the specified certificate ID.

```
aws iot delete-ca-certificate \  
  --certificate-id  
  f4efed62c0142f16af278166f61962501165c4f0536295207426460058cd1467
```

This command produces no output.

For more information, see [DeleteCACertificate](#) in the *AWS IoT API Reference*.

- For API details, see [DeleteCaCertificate](#) in *AWS CLI Command Reference*.

`delete-certificate`

The following code example shows how to use `delete-certificate`.

AWS CLI

To delete a device certificate

The following `delete-certificate` example deletes the device certificate with the specified ID.

```
aws iot delete-certificate \  
  --certificate-id  
  c0c57bbc8baaf4631a9a0345c957657f5e710473e3ddbbee1428d216d54d53ac9
```

This command produces no output.

For more information, see [DeleteCertificate](#) in the *AWS IoT API Reference*.

- For API details, see [DeleteCertificate](#) in *AWS CLI Command Reference*.

`delete-custom-metric`

The following code example shows how to use `delete-custom-metric`.

AWS CLI

To delete a custom metric

The following `delete-custom-metric` example deletes a custom metric.

```
aws iot delete-custom-metric \  
  --metric-name batteryPercentage \  
  --region us-east-1
```

Output:

```
HTTP 200
```

For more information, see [Custom metrics](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [DeleteCustomMetric](#) in *AWS CLI Command Reference*.

`delete-dimension`

The following code example shows how to use `delete-dimension`.

AWS CLI

To delete a dimension

The following `delete-dimension` example deletes a dimension called `TopicFilterForAuthMessages`.

```
aws iot delete-dimension \  
  --name TopicFilterForAuthMessages
```

This command produces no output.

For more information, see [Detect Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [DeleteDimension](#) in *AWS CLI Command Reference*.

`delete-domain-configuration`

The following code example shows how to use `delete-domain-configuration`.

AWS CLI

To delete a domain configuration

The following `delete-domain-configuration` example deletes a domain configuration named `additionalDataDomain` from your AWS account.

```
aws iot delete-domain-configuration \  
  --domain-configuration-name "additionalDataDomain" \  
  --domain-configuration-status "OK"
```

This command produces no output.

For more information, see [Configurable Endpoints](#) in the *AWS IoT Developer Guide*.

- For API details, see [DeleteDomainConfiguration](#) in *AWS CLI Command Reference*.

`delete-dynamic-thing-group`

The following code example shows how to use `delete-dynamic-thing-group`.

AWS CLI

To delete a dynamic thing group

The following `delete-dynamic-thing-group` example deletes the specified dynamic thing group.

```
aws iot delete-dynamic-thing-group \  
  --thing-group-name "RoomTooWarm"
```

This command produces no output.

For more information, see [Dynamic Thing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [DeleteDynamicThingGroup](#) in *AWS CLI Command Reference*.

delete-job-execution

The following code example shows how to use `delete-job-execution`.

AWS CLI

To delete a job execution

The following `delete-job-execution` example deletes the job execution of the specified job on a device. Use `describe-job-execution` to get the execution number.

```
aws iot delete-job-execution  
  --job-id "example-job-02"  
  --thing-name "MyRaspberryPi"  
  --execution-number 1
```

This command produces no output.

For more information, see [Creating and Managing Jobs \(CLI\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [DeleteJobExecution](#) in *AWS CLI Command Reference*.

delete-job

The following code example shows how to use `delete-job`.

AWS CLI

To delete a job

The following `delete-job` example deletes the specified job. By specifying the `--force` option, the job is deleted even if the status is `IN_PROGRESS`.

```
aws iot delete-job \  
  --job-id "example-job-04" \  
  --force
```

This command produces no output.

For more information, see [Creating and Managing Jobs \(CLI\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [DeleteJob](#) in *AWS CLI Command Reference*.

delete-mitigation-action

The following code example shows how to use `delete-mitigation-action`.

AWS CLI

To delete a mitigation action

The following `delete-mitigation-action` example deletes the specified mitigation action.

```
aws iot delete-mitigation-action \  
  --action-name AddThingsToQuarantineGroup1Action
```

This command produces no output.

For more information, see [DeleteMitigationAction \(Mitigation Action Commands\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [DeleteMitigationAction](#) in *AWS CLI Command Reference*.

delete-ota-update

The following code example shows how to use `delete-ota-update`.

AWS CLI

To delete an OTA update

The following `delete-ota-update` example deletes the specified OTA update.

```
aws iot delete-ota-update \  
  --ota-update-id ota12345 \  
  --delete-stream \  
  --force-delete-aws-job
```

This command produces no output.

For more information, see [DeleteOTAUpdate](#) in the *AWS IoT API Reference*.

- For API details, see [DeleteOtaUpdate](#) in *AWS CLI Command Reference*.

delete-policy-version

The following code example shows how to use `delete-policy-version`.

AWS CLI

To delete a version of policy

The following `delete-policy-version` example deletes version 2 of the specified policy from your AWS account.

```
aws iot delete-policy-version \  
  --policy-name UpdateDeviceCertPolicy \  
  --policy-version-id 2
```

This command produces no output.

For more information, see [AWS IoT Policies](#) in the *AWS IoT Developer Guide*.

- For API details, see [DeletePolicyVersion](#) in *AWS CLI Command Reference*.

delete-policy

The following code example shows how to use `delete-policy`.

AWS CLI

To delete a policy

The following `delete-policy` example deletes the specified policy from your AWS account.

```
aws iot delete-policy --policy-name UpdateDeviceCertPolicy
```

This command produces no output.

For more information, see [AWS IoT Policies](#) in the *AWS IoT Developers Guide*.

- For API details, see [DeletePolicy](#) in *AWS CLI Command Reference*.

delete-provisioning-template-version

The following code example shows how to use `delete-provisioning-template-version`.

AWS CLI

To delete a provisioning template version

The following `delete-provisioning-template-version` example deletes version 2 of the specified provisioning template.

```
aws iot delete-provisioning-template-version \  
  --version-id 2 \  
  --template-name "widget-template"
```

This command produces no output.

For more information, see [AWS IoT Secure Tunneling](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [DeleteProvisioningTemplateVersion](#) in *AWS CLI Command Reference*.

delete-provisioning-template

The following code example shows how to use `delete-provisioning-template`.

AWS CLI

To delete a provisioning template

The following `delete-provisioning-template` example deletes the specified provisioning template.

```
aws iot delete-provisioning-template \  
  --template-name widget-template
```

This command produces no output.

For more information, see [AWS IoT Secure Tunneling](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [DeleteProvisioningTemplate](#) in *AWS CLI Command Reference*.

delete-registration-code

The following code example shows how to use `delete-registration-code`.

AWS CLI

To delete your registration cod

The following `delete-registration-code` example deletes an AWS IoT account-specific registration code.

```
aws iot delete-registration-code
```

This command produces no output.

For more information, see [Use Your Own Certificate](#) in the *AWS IoT Developer Guide*.

- For API details, see [DeleteRegistrationCode](#) in *AWS CLI Command Reference*.

delete-role-alias

The following code example shows how to use `delete-role-alias`.

AWS CLI

To delete an AWS IoT role alias

The following `delete-role-alias` example deletes an AWS IoT role alias named `LightBulbRole`.

```
aws iot delete-role-alias \  
  --role-alias-name LightBulbRole
```

```
--role-alias LightBulbRole
```

This command produces no output.

For more information, see [Authorizing Direct Calls to AWS Services](#) in the *AWS IoT Developer Guide*.

- For API details, see [DeleteRoleAlias](#) in *AWS CLI Command Reference*.

delete-scheduled-audit

The following code example shows how to use `delete-scheduled-audit`.

AWS CLI

To delete a scheduled audit

The following `delete-scheduled-audit` example deletes the AWS IoT Device Defender scheduled audit named `AWSIoTDeviceDefenderDailyAudit`.

```
aws iot delete-scheduled-audit \  
  --scheduled-audit-name AWSIoTDeviceDefenderDailyAudit
```

This command produces no output.

For more information, see [Audit Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [DeleteScheduledAudit](#) in *AWS CLI Command Reference*.

delete-security-profile

The following code example shows how to use `delete-security-profile`.

AWS CLI

To delete a security profile

The following `delete-security-profile` example deletes a security profile named `PossibleIssue`.

```
aws iot delete-security-profile \  
  --security-profile-name PossibleIssue
```


This command produces no output.

For more information, see [Detect Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [DeleteSecurityProfile](#) in *AWS CLI Command Reference*.

delete-stream

The following code example shows how to use `delete-stream`.

AWS CLI

To delete a stream

The following `delete-stream` example deletes the specified stream.

```
aws iot delete-stream \  
  --stream-id stream12345
```

This command produces no output.

For more information, see [DeleteStream](#) in the *AWS IoT API Reference*.

- For API details, see [DeleteStream](#) in *AWS CLI Command Reference*.

delete-thing-group

The following code example shows how to use `delete-thing-group`.

AWS CLI

To delete a thing group

The following `delete-thing-group` example deletes the specified thing group. You cannot delete a thing group if it contains child thing groups.

```
aws iot delete-thing-group \  
  --thing-group-name DefectiveBulbs
```

This command produces no output.

For more information, see [Thing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [DeleteThingGroup](#) in *AWS CLI Command Reference*.

delete-thing-type

The following code example shows how to use delete-thing-type.

AWS CLI

Example 1: To delete a thing type

The following delete-thing-type example deletes a deprecated thing type.

```
aws iot delete-thing-type \  
  --thing-type-name "obsoleteThingType"
```

This command produces no output.

For more information, see [Thing Types](#) in the *AWS IoT Developers Guide*.

- For API details, see [DeleteThingType](#) in *AWS CLI Command Reference*.

delete-thing

The following code example shows how to use delete-thing.

AWS CLI

To display detailed information about a thing

The following delete-thing example deletes a thing from the AWS IoT registry for your AWS account.

```
aws iot delete-thing --thing-name "FourthBulb"
```

This command produces no output.

For more information, see [How to Manage Things with the Registry](#) in the *AWS IoT Developers Guide*.

- For API details, see [DeleteThing](#) in *AWS CLI Command Reference*.

delete-topic-rule-destination

The following code example shows how to use delete-topic-rule-destination.

AWS CLI

To delete a topic rule destination

The following `delete-topic-rule-destination` example deletes the specified topic rule destination.

```
aws iot delete-topic-rule-destination \  
  --arn "arn:aws:iot:us-west-2:123456789012:ruledestination/http/  
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"
```

This command produces no output.

For more information, see [Deleting a topic rule destination](#) in the *AWS IoT Developer Guide*.

- For API details, see [DeleteTopicRuleDestination](#) in *AWS CLI Command Reference*.

delete-topic-rule

The following code example shows how to use `delete-topic-rule`.

AWS CLI

To delete a rule

The following `delete-topic-rule` example deletes the specified rule.

```
aws iot delete-topic-rule \  
  --rule-name "LowMoistureRule"
```

This command produces no output.

For more information, see [Deleting a Rule](#) in the *AWS IoT Developers Guide*.

- For API details, see [DeleteTopicRule](#) in *AWS CLI Command Reference*.

delete-v2-logging-level

The following code example shows how to use `delete-v2-logging-level`.

AWS CLI

To delete the logging level for a thing group

The following `delete-v2-logging-level` example deletes the logging level for the specified thing group.

```
aws iot delete-v2-logging-level \  
  --target-type THING_GROUP \  
  --target-name LightBulbs
```

This command produces no output.

- For API details, see [DeleteV2LoggingLevel](#) in *AWS CLI Command Reference*.

deprecate-thing-type

The following code example shows how to use `deprecate-thing-type`.

AWS CLI

Example 1: To deprecate a thing type

The following `deprecate-thing-type` example deprecates a thing type so that users can't associate any new things with it.

```
aws iot deprecate-thing-type \  
  --thing-type-name "obsoleteThingType"
```

This command produces no output.

Example 2: To reverse the deprecation of a thing type

The following `deprecate-thing-type` example reverses the deprecation of a thing type, which makes it possible for users to associate new things with it again.

```
aws iot deprecate-thing-type \  
  --thing-type-name "obsoleteThingType" \  
  --undo-deprecate
```

This command produces no output.

For more information, see [Thing Types](#) in the *AWS IoT Developers Guide*.

- For API details, see [DeprecateThingType](#) in *AWS CLI Command Reference*.

describe-account-audit-configuration

The following code example shows how to use describe-account-audit-configuration.

AWS CLI

To view current audit configuration settings

The following describe-account-audit-configuration example lists the current settings for your AWS IoT Device Defender audit configuration.

```
aws iot describe-account-audit-configuration
```

Output:

```
{
  "roleArn": "arn:aws:iam::123456789012:role/service-role/
AWSIoTDeviceDefenderAudit_1551201085996",
  "auditNotificationTargetConfigurations": {
    "SNS": {
      "targetArn": "arn:aws:sns:us-west-2:123456789012:ddaudits",
      "roleArn": "arn:aws:iam::123456789012:role/service-role/
AWSIoTDeviceDefenderAudit",
      "enabled": true
    }
  },
  "auditCheckConfigurations": {
    "AUTHENTICATED_COGNITO_ROLE_OVERLY_PERMISSIVE_CHECK": {
      "enabled": true
    },
    "CA_CERTIFICATE_EXPIRING_CHECK": {
      "enabled": true
    },
    "CONFLICTING_CLIENT_IDS_CHECK": {
      "enabled": true
    },
    "DEVICE_CERTIFICATE_EXPIRING_CHECK": {
      "enabled": true
    },
    "DEVICE_CERTIFICATE_SHARED_CHECK": {
      "enabled": true
    },
    "IOT_POLICY_OVERLY_PERMISSIVE_CHECK": {
```

```

        "enabled": true
    },
    "LOGGING_DISABLED_CHECK": {
        "enabled": true
    },
    "REVOKED_CA_CERTIFICATE_STILL_ACTIVE_CHECK": {
        "enabled": true
    },
    "REVOKED_DEVICE_CERTIFICATE_STILL_ACTIVE_CHECK": {
        "enabled": true
    },
    "UNAUTHENTICATED_COGNITO_ROLE_OVERLY_PERMISSIVE_CHECK": {
        "enabled": true
    }
}
}
}

```

For more information, see [Audit Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [DescribeAccountAuditConfiguration](#) in *AWS CLI Command Reference*.

describe-audit-finding

The following code example shows how to use `describe-audit-finding`.

AWS CLI

To list details for an audit finding

The following `describe-audit-finding` example lists the details for the specified AWS IoT Device Defender audit finding. An audit can produce multiple findings. Use the `list-audit-findings` command to get a list of the findings from an audit to get the `findingId`.

```

aws iot describe-audit-finding \
  --finding-id "ef4826b8-e55a-44b9-b460-5c485355371b"

```

Output:

```

{
  "finding": {
    "findingId": "ef4826b8-e55a-44b9-b460-5c485355371b",

```

```

    "taskId": "873ed69c74a9ec8fa9b8e88e9abc4661",
    "checkName": "IOT_POLICY_OVERLY_PERMISSIVE_CHECK",
    "taskStartTime": 1576012045.745,
    "findingTime": 1576012046.168,
    "severity": "CRITICAL",
    "nonCompliantResource": {
      "resourceType": "IOT_POLICY",
      "resourceIdentifier": {
        "policyVersionIdentifier": {
          "policyName": "smp-ggrass-group_Core-policy",
          "policyVersionId": "1"
        }
      }
    },
    "reasonForNonCompliance": "Policy allows broad access to IoT data plane
actions: [iot:Subscribe, iot:Connect, iot:GetThingShadow, iot>DeleteThingShadow,
iot:UpdateThingShadow, iot:Publish].",
    "reasonForNonComplianceCode":
"ALLOWS_BROAD_ACCESS_TO_IOT_DATA_PLANE_ACTIONS"
  }
}

```

For more information, see [Check Audit Results \(Audit Commands\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [DescribeAuditFinding](#) in *AWS CLI Command Reference*.

describe-audit-mitigation-actions-task

The following code example shows how to use `describe-audit-mitigation-actions-task`.

AWS CLI

To show the details of an audit mitigation actions task

The following `describe-audit-mitigation-actions-task` example shows the details for the specified task, where the `ResetPolicyVersionAction` was applied to a finding. The results include when the task started and ended, how many findings were targeted (and the outcome), and the definition of the action that is applied as part of this task.

```

aws iot describe-audit-mitigation-actions-task \
  --task-id ResetPolicyTask01

```

Output:

```
{
  "taskStatus": "COMPLETED",
  "startTime": "2019-12-10T15:13:19.457000-08:00",
  "endTime": "2019-12-10T15:13:19.947000-08:00",
  "taskStatistics": {
    "IOT_POLICY_OVERLY_PERMISSIVE_CHECK": {
      "totalFindingsCount": 1,
      "failedFindingsCount": 0,
      "succeededFindingsCount": 1,
      "skippedFindingsCount": 0,
      "canceledFindingsCount": 0
    }
  },
  "target": {
    "findingIds": [
      "ef4826b8-e55a-44b9-b460-5c485355371b"
    ]
  },
  "auditCheckToActionsMapping": {
    "IOT_POLICY_OVERLY_PERMISSIVE_CHECK": [
      "ResetPolicyVersionAction"
    ]
  },
  "actionsDefinition": [
    {
      "name": "ResetPolicyVersionAction",
      "id": "1ea0b415-bef1-4a01-bd13-72fb63c59afb",
      "roleArn": "arn:aws:iam::123456789012:role/service-role/
ReplacePolicyVersionRole",
      "actionParams": {
        "replaceDefaultPolicyVersionParams": {
          "templateName": "BLANK_POLICY"
        }
      }
    }
  ]
}
```

For more information, see [DescribeAuditMitigationActionsTask \(Mitigation Action Commands\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [DescribeAuditMitigationActionsTask](#) in *AWS CLI Command Reference*.

describe-audit-suppression

The following code example shows how to use describe-audit-suppression.

AWS CLI

To get details about an audit finding suppression

The following describe-audit-suppression example lists details about an audit finding suppression.

```
aws iot describe-audit-task \  
  --task-id "787ed873b69cb4d6cdbae6ddd06996c5"
```

Output:

```
{  
  "taskStatus": "COMPLETED",  
  "taskType": "SCHEDULED_AUDIT_TASK",  
  "taskStartTime": 1596168096.157,  
  "taskStatistics": {  
    "totalChecks": 1,  
    "InProgressChecks": 0,  
    "waitingForDataCollectionChecks": 0,  
    "compliantChecks": 0,  
    "nonCompliantChecks": 1,  
    "failedChecks": 0,  
    "canceledChecks": 0  
  },  
  "scheduledAuditName": "AWSIoTDeviceDefenderDailyAudit",  
  "auditDetails": {  
    "DEVICE_CERTIFICATE_EXPIRING_CHECK": {  
      "checkRunStatus": "COMPLETED_NON_COMPLIANT",  
      "checkCompliant": false,  
      "totalResourcesCount": 195,  
      "nonCompliantResourcesCount": 2  
    }  
  }  
}
```

For more information, see [Audit finding suppressions](#) in the *AWS IoT Developers Guide*.

- For API details, see [DescribeAuditSuppression](#) in *AWS CLI Command Reference*.

describe-audit-task

The following code example shows how to use `describe-audit-task`.

AWS CLI

To get information about an audit instance

The following `describe-audit-task` example gets information about an instance of an AWS IoT Device Defender audit. If the audit is complete, summary statistics for the run are included in the results.

```
aws iot describe-audit-task \  
  --task-id a3aea009955e501a31b764abe1bebd3d
```

Output:

```
{  
  "taskStatus": "COMPLETED",  
  "taskType": "ON_DEMAND_AUDIT_TASK",  
  "taskStartTime": 1560356923.434,  
  "taskStatistics": {  
    "totalChecks": 3,  
    "inProgressChecks": 0,  
    "waitingForDataCollectionChecks": 0,  
    "compliantChecks": 3,  
    "nonCompliantChecks": 0,  
    "failedChecks": 0,  
    "canceledChecks": 0  
  },  
  "auditDetails": {  
    "CA_CERTIFICATE_EXPIRING_CHECK": {  
      "checkRunStatus": "COMPLETED_COMPLIANT",  
      "checkCompliant": true,  
      "totalResourcesCount": 0,  
      "nonCompliantResourcesCount": 0  
    },  
    "DEVICE_CERTIFICATE_EXPIRING_CHECK": {  
      "checkRunStatus": "COMPLETED_COMPLIANT",  
      "checkCompliant": true,  
      "totalResourcesCount": 6,  
      "nonCompliantResourcesCount": 0  
    }  
  }  
}
```

```

    },
    "REVOKED_CA_CERTIFICATE_STILL_ACTIVE_CHECK": {
      "checkRunStatus": "COMPLETED_COMPLIANT",
      "checkCompliant": true,
      "totalResourcesCount": 0,
      "nonCompliantResourcesCount": 0
    }
  }
}

```

For more information, see [Audit Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [DescribeAuditTask](#) in *AWS CLI Command Reference*.

describe-authorizer

The following code example shows how to use `describe-authorizer`.

AWS CLI

To get information about a custom authorizer

The following `describe-authorizer` example displays details for the specified custom authorizer.

```

aws iot describe-authorizer \
  --authorizer-name CustomAuthorizer

```

Output:

```

{
  "authorizerDescription": {
    "authorizerName": "CustomAuthorizer",
    "authorizerArn": "arn:aws:iot:us-west-2:123456789012:authorizer/
CustomAuthorizer",
    "authorizerFunctionArn": "arn:aws:lambda:us-
west-2:123456789012:function:CustomAuthorizerFunction",
    "tokenKeyName": "MyAuthToken",
    "tokenSigningPublicKeys": {
      "FIRST_KEY": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA1uJOB4lQPgG/1M6ZfIwo
\nZ+7ENxAio9q6QD4FFqjGZsvjtYwjoe1RKK0U8Eq9xb503kRSmyIwTzwzm/f4Gf0Y

```

```

\nZUloJ+t3PUUwHrmbYTAgrCUgRFyggjfgVwGCPs5ZAX4Eyqt5cr+AIHIiUDbxSa7p
\nzw0BKPeic0asNJpqT8PkBbRaKylEJh5oo81NDHHmVtbBm5A5YiJjqYXLaVAowKzZ\n
+GqsNvAQ9Jy1wI2VrEa10fL8f1DB/BJLm7zjpfPOHDJQgID0XnZwA1NnZc0hCwIx\n50g2LW20y9R/
dmqtDmJiVP97Z4GykxPvwLYHrUXY0iW1R3AR/Ac1NhCTGZMwVDB1\nlQIDAQAB\n-----END PUBLIC
KEY-----"
    },
    "status": "ACTIVE",
    "creationDate": 1571245658.069,
    "lastModifiedDate": 1571245658.069
  }
}

```

For more information, see [DescribeAuthorizer](#) in the *AWS IoT API Reference*.

- For API details, see [DescribeAuthorizer](#) in *AWS CLI Command Reference*.

describe-billing-group

The following code example shows how to use `describe-billing-group`.

AWS CLI

To get information about a billing group

The following `describe-billing-group` example gets information for the specified billing group.

```
aws iot describe-billing-group --billing-group-name GroupOne
```

Output:

```

{
  "billingGroupName": "GroupOne",
  "billingGroupId": "103de383-114b-4f51-8266-18f209ef5562",
  "billingGroupArn": "arn:aws:iot:us-west-2:123456789012:billinggroup/GroupOne",
  "version": 1,
  "billingGroupProperties": {},
  "billingGroupMetadata": {
    "creationDate": 1560199355.378
  }
}

```

For more information, see [Billing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [DescribeBillingGroup](#) in *AWS CLI Command Reference*.

describe-ca-certificate

The following code example shows how to use describe-ca-certificate.

AWS CLI

To get details about a CA certificate

The following describe-ca-certificate example displays the details for the specified CA certificate.

```
aws iot describe-ca-certificate \
  --certificate-id
  f4efed62c0142f16af278166f61962501165c4f0536295207426460058cd1467
```

Output:

```
{
  "certificateDescription": {
    "certificateArn": "arn:aws:iot:us-west-2:123456789012:cacert/
f4efed62c0142f16af278166f61962501165c4f0536295207426460058cd1467",
    "certificateId":
"f4efed62c0142f16af278166f61962501165c4f0536295207426460058cd1467",
    "status": "INACTIVE",
    "certificatePem": "-----BEGIN CERTIFICATE-----
\nMIICzzCCAbegEXAMPLEJANVEPWX18taPMA0GCSqGSIb3DQEBBQUAMB4xCzAJBgNV
\nBAYTA1VMTQ8wDQYDVQQKDAZBbWF6b24wHhcNMTEwMjEzMTU1WhcNMjEwMjEzMTU1
\nMjEzMTU1WjAeMQswCQYDVQQGEwJVUzEPMA0GA1UECgwGQW1hem9uMIIBIjANBgkq
\nhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAZd3R3ioa1CS0MhFwFBrVGR036EK07UAF
\nVdz9EXAMPLE1VczICbADnATK522kEIB51/18Vz1FtAhQL5V5eybXKnB7QebNer5m
\n4Yibx7shR5oqNzFsrXWxuugN5+w5gEfqNMaw0jhF4Lscu1KG49yuqjcDU19/13ua
\n3B2gxs1Pe7TiWwvUskzxb01F2WCshbEJvqY8fIwtGYCjTeJAgQ9hvZx/69XhKen
\nwV9LJw0QxrsUS0Ty8IHwbB8fRy72VM3u7fJoaU+n04jd5cqaoEPtzoEPUEXAMPLE
\nyVAJpqHwgbYbcUfn7V+AB6yh1+0Fa1rEQGuZDPGyJslxwr5vh8nRewIDAQABoxAw
\nDjAMBgNVHRMBETADAQH/MA0GCSqGSIb3DQEBBQUAA4IBAQA+3a5CV3IJg0nd0AgI
\nBgVMtmYzTvqAngx26aG9/spvCjXckh2SBF+EcB1CFwH1yakwjJL1dR4yarnrfxgI
\nEqP4A0YVimAVoQ5FBwnloHe16+3qtDib1U9DeXBUctS55EcfEXAMPLEYtXdqU5C
\nU9ia4KAjV0dxW1+EFYMwX5eGeb0gDTNHBy1V6B/f0SZiQAwDYp4x3B+gAP+a/bWB
\nu1um0qtBdWe6L6/83L+JhaTByqV25iVJ4c/UZUnG8926wU1DM9zQvEXuEVvzZ7+m\n4PSNqst/
```

```
nV0vnLpoG4e0WgcJgANuB33CSwtjWSuYsbhmqQRknGhREXAMPLEZT4fm\nfo0e\n-----END
CERTIFICATE-----\n",
  "ownedBy": "123456789012",
  "creationDate": 1569365372.053,
  "autoRegistrationStatus": "DISABLE",
  "lastModifiedDate": 1569365372.053,
  "customerVersion": 1,
  "generationId": "c5c2eb95-140b-4f49-9393-6aaac85b2a90",
  "validity": {
    "notBefore": 1569360675.0,
    "notAfter": 1884720675.0
  }
}
```

For more information, see [DescribeCACertificate](#) in the *AWS IoT API Reference*.

- For API details, see [DescribeCaCertificate](#) in *AWS CLI Command Reference*.

describe-certificate

The following code example shows how to use describe-certificate.

AWS CLI

To get information about a certificate

The following describe-certificate example displays the details for the specified certificate.

```
aws iot describe-certificate \
  --certificate-id
  "4f0ba725787aa94d67d2fca420eca022242532e8b3c58e7465c7778b443fd65e"
```

Output:

```
{
  "certificateDescription": {
    "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/4f0ba725787aa94d67d2fca420eca022242532e8b3c58e7465c7778b443fd65e",
    "certificateId":
    "4f0ba725787aa94d67d2fca420eca022242532e8b3c58e7465c7778b443fd65e",
    "status": "ACTIVE",
```

```

    "certificatePem": "-----BEGIN CERTIFICATE-----
MIICiTEXAMPLEQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBEXAMPLEMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBA5TC01BTSBDEXAMPLE1MRIwEAYDVQQDEwLUZXN0Q21sYWxhZAd
BgkqhkiG9w0BCQEWEG5vb251QGftYXpvbi5EXAMPLEcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNEXAMPLEdBMRwDgYD
VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBA5TC01BEXAMPEz
b2xEXAMPLEYDVQQDEwLUZXN0Q21sYWxhZAdBgkqhkiG9w0BCQEWEG5vb251QGft
YXpvbi5jb20wgZ8EXAMPLEZIHvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySwTc2XADZ4nB+BLYEXAMPLEpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7EXAMPLEGBzZswY6786m86gpE
Ibb30hjZnczvQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFEXAMPLEAtCu4
nUhVvxYUnEXAMPLE8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJIIJ00zbhNYS5f6GEXAMPLE10ZxBHjJnyp3780D8uTs7fLvJx79LjStb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----",
    "ownedBy": "123456789012",
    "creationDate": 1541022751.983,
    "lastModifiedDate": 1541022751.983,
    "customerVersion": 1,
    "transferData": {},
    "generationId": "6974fbed-2e61-4114-bc5e-4204cc79b045",
    "validity": {
        "notBefore": 1541022631.0,
        "notAfter": 2524607999.0
    }
}
}
}

```

For more information, see [DescribeCertificate](#) in the *AWS IoT API Reference*.

- For API details, see [DescribeCertificate](#) in *AWS CLI Command Reference*.

describe-custom-metric

The following code example shows how to use `describe-custom-metric`.

AWS CLI

To get information about a Device Defender custom metric

The following `describe-custom-metric` example gets information about a custom metric named `myCustomMetric`.

```
aws iot describe-custom-metric \  
  --metric-name myCustomMetric
```

Output:

```
{  
  "metricName": "myCustomMetric",  
  "metricArn": "arn:aws:iot:us-east-1:1234564789012:custommetric/myCustomMetric",  
  "metricType": "number",  
  "displayName": "My custom metric",  
  "creationDate": 2020-11-17T23:02:12.879000-09:00,  
  "lastModifiedDate": 2020-11-17T23:02:12.879000-09:00  
}
```

For more information, see [Custom metrics](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [DescribeCustomMetric](#) in *AWS CLI Command Reference*.

describe-default-authorizer

The following code example shows how to use `describe-default-authorizer`.

AWS CLI

To get information about the default custom authorizer

The following `describe-default-authorizer` example displays details for the default custom authorizer.

```
aws iot describe-default-authorizer
```

Output:

```
{  
  "authorizerName": "CustomAuthorizer",  
  "authorizerArn": "arn:aws:iot:us-west-2:123456789012:authorizer/  
CustomAuthorizer"  
}
```

For more information, see [DescribeDefaultAuthorizer](#) in the *AWS IoT API Reference*.

- For API details, see [DescribeDefaultAuthorizer](#) in *AWS CLI Command Reference*.

describe-dimension

The following code example shows how to use describe-dimension.

AWS CLI

To get information about a dimension

The following describe-dimension example gets information about a dimension named TopicFilterForAuthMessages.

```
aws iot describe-dimension \  
  --name TopicFilterForAuthMessages
```

Output:

```
{  
  "name": "TopicFilterForAuthMessages",  
  "arn": "arn:aws:iot:eu-west-2:123456789012:dimension/  
TopicFilterForAuthMessages",  
  "type": "TOPIC_FILTER",  
  "stringValues": [  
    "device/+/auth"  
  ],  
  "creationDate": 1578620223.255,  
  "lastModifiedDate": 1578620223.255  
}
```

For more information, see [Detect Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [DescribeDimension](#) in *AWS CLI Command Reference*.

describe-domain-configuration

The following code example shows how to use describe-domain-configuration.

AWS CLI

To describe a domain configuration

The following describe-domain-configuration example displays details about the specified domain configuration.

```
aws iot describe-domain-configuration \  
  --domain-configuration-name "additionalDataDomain"
```

Output:

```
{  
  "domainConfigurationName": "additionalDataDomain",  
  "domainConfigurationArn": "arn:aws:iot:us-  
east-1:758EXAMPLE143:domainconfiguration/additionalDataDomain/norpw",  
  "domainName": "d055exampleed74y71zfd-ats.beta.us-east-1.iot.amazonaws.com",  
  "serverCertificates": [],  
  "domainConfigurationStatus": "ENABLED",  
  "serviceType": "DATA",  
  "domainType": "AWS_MANAGED",  
  "lastStatusChangeDate": 1601923783.774  
}
```

For more information, see [Configurable Endpoints](#) in the *AWS IoT Developer Guide*.

- For API details, see [DescribeDomainConfiguration](#) in *AWS CLI Command Reference*.

describe-endpoint

The following code example shows how to use describe-endpoint.

AWS CLI

Example 1: To get your current AWS endpoint

The following describe-endpoint example retrieves the default AWS endpoint to which all commands are applied.

```
aws iot describe-endpoint
```

Output:

```
{  
  "endpointAddress": "abc123defghijk.iot.us-west-2.amazonaws.com"  
}
```

For more information, see [DescribeEndpoint](#) in the *AWS IoT Developer Guide*.

Example 2: To get your ATS endpoint

The following `describe-endpoint` example retrieves the Amazon Trust Services (ATS) endpoint.

```
aws iot describe-endpoint \  
  --endpoint-type iot:Data-ATS
```

Output:

```
{  
  "endpointAddress": "abc123defghijk-ats.iot.us-west-2.amazonaws.com"  
}
```

For more information, see [X.509 Certificates and AWS IoT](#) in the *AWS IoT Developer Guide*.

- For API details, see [DescribeEndpoint](#) in *AWS CLI Command Reference*.

describe-event-configurations

The following code example shows how to use `describe-event-configurations`.

AWS CLI

To show which event types are published

The following `describe-event-configurations` example lists the configuration that controls which events are generated when something is added, updated, or deleted.

```
aws iot describe-event-configurations
```

Output:

```
{  
  "eventConfigurations": {  
    "CA_CERTIFICATE": {  
      "Enabled": false  
    },  
    "CERTIFICATE": {  
      "Enabled": false  
    },  
  },  
}
```

```
"JOB": {
  "Enabled": false
},
"JOB_EXECUTION": {
  "Enabled": false
},
"POLICY": {
  "Enabled": false
},
"THING": {
  "Enabled": false
},
"THING_GROUP": {
  "Enabled": false
},
"THING_GROUP_HIERARCHY": {
  "Enabled": false
},
"THING_GROUP_MEMBERSHIP": {
  "Enabled": false
},
"THING_TYPE": {
  "Enabled": false
},
"THING_TYPE_ASSOCIATION": {
  "Enabled": false
}
}
```

For more information, see [Event Messages](#) in the *AWS IoT Developer Guide*.

- For API details, see [DescribeEventConfigurations](#) in *AWS CLI Command Reference*.

describe-index

The following code example shows how to use `describe-index`.

AWS CLI

To retrieve the current status of the thing index

The following `describe-index` example retrieves the current status of the thing index.

```
aws iot describe-index \  
  --index-name "AWS_Things"
```

Output:

```
{  
  "indexName": "AWS_Things",  
  "indexStatus": "ACTIVE",  
  "schema": "REGISTRY_AND_SHADOW_AND_CONNECTIVITY_STATUS"  
}
```

For more information, see [Managing Thing Indexing](#) in the *AWS IoT Developer Guide*.

- For API details, see [DescribeIndex](#) in *AWS CLI Command Reference*.

describe-job-execution

The following code example shows how to use `describe-job-execution`.

AWS CLI

To get execution details for a job on a device

The following `describe-job-execution` example gets execution details for the specified job.

```
aws iot describe-job-execution \  
  --job-id "example-job-01" \  
  --thing-name "MyRaspberryPi"
```

Output:

```
{  
  "execution": {  
    "jobId": "example-job-01",  
    "status": "QUEUED",  
    "statusDetails": {},  
    "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MyRaspberryPi",  
    "queuedAt": 1560787023.636,  
    "lastUpdatedAt": 1560787023.636,  
    "executionNumber": 1,  
    "versionNumber": 1  
  }  
}
```

```
}
```

For more information, see [Creating and Managing Jobs \(CLI\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [DescribeJobExecution](#) in *AWS CLI Command Reference*.

describe-job

The following code example shows how to use `describe-job`.

AWS CLI

To get detailed status for a job

The following `describe-job` example gets detailed status for the job whose ID is `example-job-01`.

```
aws iot describe-job \  
  --job-id "example-job-01"
```

Output:

```
{  
  "job": {  
    "jobArn": "arn:aws:iot:us-west-2:123456789012:job/example-job-01",  
    "jobId": "example-job-01",  
    "targetSelection": "SNAPSHOT",  
    "status": "IN_PROGRESS",  
    "targets": [  
      "arn:aws:iot:us-west-2:123456789012:thing/MyRaspberryPi"  
    ],  
    "description": "example job test",  
    "presignedUrlConfig": {},  
    "jobExecutionsRolloutConfig": {},  
    "createdAt": 1560787022.733,  
    "lastUpdatedAt": 1560787026.294,  
    "jobProcessDetails": {  
      "numberOfCanceledThings": 0,  
      "numberOfSucceededThings": 0,  
      "numberOfFailedThings": 0,  
      "numberOfRejectedThings": 0,  
      "numberOfQueuedThings": 1,  
    }  
  }  
}
```

```

        "numberOfInProgressThings": 0,
        "numberOfRemovedThings": 0,
        "numberOfTimedOutThings": 0
    },
    "timeoutConfig": {}
}
}

```

For more information, see [Creating and Managing Jobs \(CLI\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [DescribeJob](#) in *AWS CLI Command Reference*.

describe-mitigation-action

The following code example shows how to use `describe-mitigation-action`.

AWS CLI

To view the details for a defined mitigation action

The following `describe-mitigation-action` example displays details for the specified mitigation action.

```

aws iot describe-mitigation-action \
  --action-name AddThingsToQuarantineGroupAction

```

Output:

```

{
  "actionName": "AddThingsToQuarantineGroupAction",
  "actionType": "ADD_THINGS_TO_THING_GROUP",
  "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/AddThingsToQuarantineGroupAction",
  "actionId": "2fd2726d-98e1-4abf-b10f-09465ccd6bfa",
  "roleArn": "arn:aws:iam::123456789012:role/service-role/MoveThingsToQuarantineGroupRole",
  "actionParams": {
    "addThingsToThingGroupParams": {
      "thingGroupNames": [
        "QuarantineGroup1"
      ],
      "overrideDynamicGroups": true
    }
  }
}

```

```
  },
  "creationDate": "2019-12-10T11:09:35.999000-08:00",
  "lastModifiedDate": "2019-12-10T11:09:35.999000-08:00"
}
```

For more information, see [DescribeMitigationAction \(Mitigation Action Commands\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [DescribeMitigationAction](#) in *AWS CLI Command Reference*.

describe-provisioning-template-version

The following code example shows how to use `describe-provisioning-template-version`.

AWS CLI

To describe a provisioning template version

The following `describe-provisioning-template-version` example describes a provisioning template version.

```
aws iot describe-provisioning-template-version \
  --template-name MyTestProvisioningTemplate \
  --version-id 1
```

Output:

```
{
  "versionId": 1,
  "creationDate": 1589308310.574,
  "templateBody": "{
    \"Parameters\":{
      \"SerialNumber\":{
        \"Type\": \"String\"
      },
      \"AWS::IoT::Certificate::Id\":{
        \"Type\": \"String\"
      }
    },
    \"Resources\":{
      \"certificate\":{
        \"Properties\":{
```



```

        \\"CertificateId\\":{
            \\"Ref\\":\\"AWS::IoT::Certificate::Id\\"
        },
        \\"Status\\":\\"Active\\"
    },
    \\"Type\\":\\"AWS::IoT::Certificate\\"
},
\\"policy\\":{
    \\"Properties\\":{
        \\"PolicyName\\":\\"MyIotPolicy\\"
    },
    \\"Type\\":\\"AWS::IoT::Policy\\"
},
\\"thing\\":{
    \\"OverrideSettings\\":{
        \\"AttributePayload\\":\\"MERGE\\",
        \\"ThingGroups\\":\\"DO_NOTHING\\",
        \\"ThingTypeName\\":\\"REPLACE\\"
    },
    \\"Properties\\":{
        \\"AttributePayload\\":{},
        \\"ThingGroups\\":[],
        \\"ThingName\\":{
            \\"Fn::Join\\":[
                \\"\\",
                [
                    \\"DemoGroup_\\",
                    {\\"Ref\\":\\"SerialNumber\\"}
                ]
            ]
        },
        \\"ThingTypeName\\":\\"VirtualThings\\"
    },
    \\"Type\\":\\"AWS::IoT::Thing\\"
}
}
},
"setDefaultVersion": true
}

```

For more information, see [Provisioning devices that don't have device certificates using fleet provisioning](#) in the *AWS IoT Core Developers Guide*.

- For API details, see [DescribeProvisioningTemplateVersion](#) in *AWS CLI Command Reference*.

describe-provisioning-template

The following code example shows how to use describe-provisioning-template.

AWS CLI

To describe a provisioning template

The following describe-provisioning-template example describes a provisioning template.

```
aws iot describe-provisioning-template \  
  --template-name MyTestProvisioningTemplate
```

Output:

```
{  
  "templateArn": "arn:aws:iot:us-west-2:57EXAMPLE833:provisioningtemplate/  
MyTestProvisioningTemplate",  
  "templateName": "MyTestProvisioningTemplate",  
  "creationDate": 1589308310.574,  
  "lastModifiedDate": 1589308345.539,  
  "defaultVersionId": 1,  
  "templateBody": "{  
    \"Parameters\":{  
      \"SerialNumber\":{  
        \"Type\":\"String\"  
      },  
      \"AWS::IoT::Certificate::Id\":{  
        \"Type\":\"String\"  
      }  
    },  
    \"Resources\":{  
      \"certificate\":{  
        \"Properties\":{  
          \"CertificateId\":{  
            \"Ref\":\"AWS::IoT::Certificate::Id\"  
          },  
          \"Status\":\"Active\"  
        },  
        \"Type\":\"AWS::IoT::Certificate\"  
      },  
      \"policy\":{
```

```

        \ "Properties\":{
            \ "PolicyName\":\ "MyIotPolicy\ "
        },
        \ "Type\":\ "AWS::IoT::Policy\ "
    },
    \ "thing\":{
        \ "OverrideSettings\":{
            \ "AttributePayload\":\ "MERGE\ ",
            \ "ThingGroups\":\ "DO_NOTHING\ ",
            \ "ThingTypeName\":\ "REPLACE\ "
        },
        \ "Properties\":{
            \ "AttributePayload\":{ },
            \ "ThingGroups\":[ ],
            \ "ThingName\":{
                \ "Fn::Join\":[
                    \ "\ ",
                    [
                        \ "DemoGroup_\ ",
                        { \ "Ref\":\ "SerialNumber\ " }
                    ]
                ]
            },
            \ "ThingTypeName\":\ "VirtualThings\ "
        },
        \ "Type\":\ "AWS::IoT::Thing\ "
    }
}
},
"enabled": true,
"provisioningRoleArn": "arn:aws:iam::571032923833:role/service-role/IoT_access"
}

```

For more information, see [Provisioning devices that don't have device certificates using fleet provisioning](#) in the *AWS IoT Core Developers Guide*.

- For API details, see [DescribeProvisioningTemplate](#) in *AWS CLI Command Reference*.

describe-role-alias

The following code example shows how to use `describe-role-alias`.

AWS CLI

To get information about an AWS IoT role alias

The following `describe-role-alias` example displays details for the specified role alias.

```
aws iot describe-role-alias \  
  --role-alias LightBulbRole
```

Output:

```
{  
  "roleAliasDescription": {  
    "roleAlias": "LightBulbRole",  
    "roleAliasArn": "arn:aws:iot:us-west-2:123456789012:rolealias/  
LightBulbRole",  
    "roleArn": "arn:aws:iam::123456789012:role/light_bulb_role_001",  
    "owner": "123456789012",  
    "credentialDurationSeconds": 3600,  
    "creationDate": 1570558643.221,  
    "lastModifiedDate": 1570558643.221  
  }  
}
```

For more information, see [DescribeRoleAlias](#) in the *AWS IoT API Reference*.

- For API details, see [DescribeRoleAlias](#) in *AWS CLI Command Reference*.

describe-scheduled-audit

The following code example shows how to use `describe-scheduled-audit`.

AWS CLI

To get information about a scheduled audit

The following `describe-scheduled-audit` example gets detailed information about an AWS IoT Device Defender scheduled audit named `AWSIoTDeviceDefenderDailyAudit`.

```
aws iot describe-scheduled-audit \  
  --scheduled-audit-name AWSIoTDeviceDefenderDailyAudit
```

Output:

```
{
  "frequency": "DAILY",
  "targetCheckNames": [
    "AUTHENTICATED_COGNITO_ROLE_OVERLY_PERMISSIVE_CHECK",
    "CONFLICTING_CLIENT_IDS_CHECK",
    "DEVICE_CERTIFICATE_SHARED_CHECK",
    "IOT_POLICY_OVERLY_PERMISSIVE_CHECK",
    "REVOKED_CA_CERTIFICATE_STILL_ACTIVE_CHECK",
    "UNAUTHENTICATED_COGNITO_ROLE_OVERLY_PERMISSIVE_CHECK"
  ],
  "scheduledAuditName": "AWSIoTDeviceDefenderDailyAudit",
  "scheduledAuditArn": "arn:aws:iot:us-west-2:123456789012:scheduledaudit/
AWSIoTDeviceDefenderDailyAudit"
}
```

For more information, see [Audit Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [DescribeScheduledAudit](#) in *AWS CLI Command Reference*.

describe-security-profile

The following code example shows how to use describe-security-profile.

AWS CLI

To get information about a security profile

The following describe-security-profile example gets information about the AWS IoT Device Defender security profile named PossibleIssue.

```
aws iot describe-security-profile \
  --security-profile-name PossibleIssue
```

Output:

```
{
  "securityProfileName": "PossibleIssue",
  "securityProfileArn": "arn:aws:iot:us-west-2:123456789012:securityprofile/
PossibleIssue",
  "securityProfileDescription": "check to see if authorization fails 10 times in 5
minutes or if cellular bandwidth exceeds 128",
  "behaviors": [
```

```

    {
      "name": "CellularBandwidth",
      "metric": "aws:message-byte-size",
      "criteria": {
        "comparisonOperator": "greater-than",
        "value": {
          "count": 128
        },
        "consecutiveDatapointsToAlarm": 1,
        "consecutiveDatapointsToClear": 1
      }
    },
    {
      "name": "Authorization",
      "metric": "aws:num-authorization-failures",
      "criteria": {
        "comparisonOperator": "greater-than",
        "value": {
          "count": 10
        },
        "durationSeconds": 300,
        "consecutiveDatapointsToAlarm": 1,
        "consecutiveDatapointsToClear": 1
      }
    }
  ],
  "version": 1,
  "creationDate": 1560278102.528,
  "lastModifiedDate": 1560278102.528
}

```

For more information, see [Detect Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [DescribeSecurityProfile](#) in *AWS CLI Command Reference*.

describe-stream

The following code example shows how to use `describe-stream`.

AWS CLI

To get information about a stream

The following `describe-stream` example displays the details about the specified stream.

```
aws iot describe-stream \  
  --stream-id stream12345
```

Output:

```
{  
  "streamInfo": {  
    "streamId": "stream12345",  
    "streamArn": "arn:aws:iot:us-west-2:123456789012:stream/stream12345",  
    "streamVersion": 1,  
    "description": "This stream is used for Amazon FreeRTOS OTA Update 12345.",  
    "files": [  
      {  
        "fileId": "123",  
        "s3Location": {  
          "bucket": "codesign-ota-bucket",  
          "key": "48c67f3c-63bb-4f92-a98a-4ee0fbc2bef6"  
        }  
      }  
    ],  
    "createdAt": 1557863215.995,  
    "lastUpdatedAt": 1557863215.995,  
    "roleArn": "arn:aws:iam:123456789012:role/service-role/my_ota_stream_role"  
  }  
}
```

For more information, see [DescribeStream](#) in the *AWS IoT API Reference*.

- For API details, see [DescribeStream](#) in *AWS CLI Command Reference*.

describe-thing-group

The following code example shows how to use `describe-thing-group`.

AWS CLI

To get information about a thing group

The following `describe-thing-group` example gets information about the thing group named `HalogenBulbs`.

```
aws iot describe-thing-group \  
  --thing-group-name HalogenBulbs
```

```
--thing-group-name HalogenBulbs
```

Output:

```
{
  "thingGroupName": "HalogenBulbs",
  "thingGroupId": "f4ec6b84-b42b-499d-9ce1-4dbd4d4f6f6e",
  "thingGroupArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/HalogenBulbs",
  "version": 1,
  "thingGroupProperties": {},
  "thingGroupMetadata": {
    "parentGroupName": "LightBulbs",
    "rootToParentThingGroups": [
      {
        "groupName": "LightBulbs",
        "groupArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/
LightBulbs"
      }
    ],
    "creationDate": 1559927609.897
  }
}
```

For more information, see [Thing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [DescribeThingGroup](#) in *AWS CLI Command Reference*.

describe-thing-type

The following code example shows how to use `describe-thing-type`.

AWS CLI

To get information about a thing type

The following `describe-thing-type` example displays information about the specified thing type defined in your AWS account.

```
aws iot describe-thing-type \
  --thing-type-name "LightBulb"
```

Output:


```
{
  "thingTypeName": "LightBulb",
  "thingTypeId": "ce3573b0-0a3c-45a7-ac93-4e0ce14cd190",
  "thingTypeArn": "arn:aws:iot:us-west-2:123456789012:thingtype/LightBulb",
  "thingTypeProperties": {
    "thingTypeDescription": "light bulb type",
    "searchableAttributes": [
      "model",
      "wattage"
    ]
  },
  "thingTypeMetadata": {
    "deprecated": false,
    "creationDate": 1559772562.498
  }
}
```

For more information, see [Thing Types](#) in the *AWS IoT Developers Guide*.

- For API details, see [DescribeThingType](#) in *AWS CLI Command Reference*.

describe-thing

The following code example shows how to use describe-thing.

AWS CLI

To display detailed information about a thing

The following describe-thing example display information about a thing (device) that is defined in the AWS IoT registry for your AWS account.

```
aws iot describe-thing --thing-name "MyLightBulb"
```

Output:

```
{
  "defaultClientId": "MyLightBulb",
  "thingName": "MyLightBulb",
  "thingId": "40da2e73-c6af-406e-b415-15acae538797",
  "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MyLightBulb",
  "thingTypeName": "LightBulb",
  "attributes": {
```

```
    "model": "123",
    "wattage": "75"
  },
  "version": 1
}
```

For more information, see [How to Manage Things with the Registry](#) in the *AWS IoT Developers Guide*.

- For API details, see [DescribeThing](#) in *AWS CLI Command Reference*.

detach-policy

The following code example shows how to use `detach-policy`.

AWS CLI

Example 1: To detach an AWS IoT policy from a thing group

The following `detach-policy` example detaches the specified policy from a thing group and, by extension, from all things in that group and any of the group's child groups.

```
aws iot detach-policy \
  --target "arn:aws:iot:us-west-2:123456789012:thinggroup/LightBulbs" \
  --policy-name "MyFirstGroup_Core-policy"
```

This command produces no output.

For more information, see [Thing Groups](#) in the *AWS IoT Developers Guide*.

Example 2: To detach an AWS IoT policy from a device certificate

The following `detach-policy` example detaches the `TemperatureSensorPolicy` policy from a device certificate identified by ARN.

```
aws iot detach-policy \
  --policy-name TemperatureSensorPolicy \
  --target arn:aws:iot:us-
west-2:123456789012:cert/488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142
```

This command produces no output.

- For API details, see [DetachPolicy](#) in *AWS CLI Command Reference*.

detach-security-profile

The following code example shows how to use `detach-security-profile`.

AWS CLI

To disassociate a security profile from a target

The following `detach-security-profile` example removes the association between the AWS IoT Device Defender security profile named `Testprofile` and the all registered things target.

```
aws iot detach-security-profile \  
  --security-profile-name Testprofile \  
  --security-profile-target-arn "arn:aws:iot:us-west-2:123456789012:all/  
registered-things"
```

This command produces no output.

For more information, see [Detect Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [DetachSecurityProfile](#) in *AWS CLI Command Reference*.

detach-thing-principal

The following code example shows how to use `detach-thing-principal`.

AWS CLI

To detach a certificate/principal from a thing

The following `detach-thing-principal` example removes a certificate that represents a principal from the specified thing.

```
aws iot detach-thing-principal \  
  --thing-name "MyLightBulb" \  
  --principal "arn:aws:iot:us-  
west-2:123456789012:cert/604c48437a57b7d5fc5d137c5be75011c6ee67c9a6943683a1acb4b1626bac36"
```

This command produces no output.

For more information, see [How to Manage Things with the Registry](#) in the *AWS IoT Developers Guide*.

- For API details, see [DetachThingPrincipal](#) in *AWS CLI Command Reference*.

disable-topic-rule

The following code example shows how to use `disable-topic-rule`.

AWS CLI

To disable a topic rule

The following `disable-topic-rule` example disables the specified topic rule.

```
aws iot disable-topic-rule \  
  --rule-name "MyPlantPiMoistureAlertRule"
```

This command produces no output.

For more information, see [Viewing Your Rules](#) in the *AWS IoT Developer Guide*.

- For API details, see [DisableTopicRule](#) in *AWS CLI Command Reference*.

enable-topic-rule

The following code example shows how to use `enable-topic-rule`.

AWS CLI

To enable a topic rule

The following `enable-topic-rule` example enables (or re-enables) the specified topic rule.

```
aws iot enable-topic-rule \  
  --rule-name "MyPlantPiMoistureAlertRule"
```

This command produces no output.

For more information, see [Viewing Your Rules](#) in the *AWS IoT Developer Guide*.

- For API details, see [EnableTopicRule](#) in *AWS CLI Command Reference*.

get-behavior-model-training-summaries

The following code example shows how to use `get-behavior-model-training-summaries`.

AWS CLI

To list a Device Defender's ML Detect Security Profile training model's status

The following `get-behavior-model-training-summaries` example lists model training status for the configured behaviors in the chosen Security Profile. For each behavior, the name, model status, and percentage of datapoints collected are listed.

```
aws iot get-behavior-model-training-summaries \  
  --security-profile-name MySecuirtyProfileName
```

Output:

```
{  
  "summaries": [  
    {  
      "securityProfileName": "MySecuirtyProfileName",  
      "behaviorName": "Messages_sent_ML_behavior",  
      "modelStatus": "PENDING_BUILD",  
      "datapointsCollectionPercentage": 0.0  
    },  
    {  
      "securityProfileName": "MySecuirtyProfileName",  
      "behaviorName": "Messages_received_ML_behavior",  
      "modelStatus": "PENDING_BUILD",  
      "datapointsCollectionPercentage": 0.0  
    },  
    {  
      "securityProfileName": "MySecuirtyProfileName",  
      "behaviorName": "Authorization_failures_ML_behavior",  
      "modelStatus": "PENDING_BUILD",  
      "datapointsCollectionPercentage": 0.0  
    },  
    {  
      "securityProfileName": "MySecuirtyProfileName",  
      "behaviorName": "Message_size_ML_behavior",  
      "modelStatus": "PENDING_BUILD",  
      "datapointsCollectionPercentage": 0.0  
    },  
    {  
      "securityProfileName": "MySecuirtyProfileName",  
      "behaviorName": "Connection_attempts_ML_behavior",  
      "modelStatus": "PENDING_BUILD",  
      "datapointsCollectionPercentage": 0.0  
    }  
  ]  
}
```

```

        "datapointsCollectionPercentage": 0.0
    },
    {
        "securityProfileName": "MySPNoAlerts",
        "behaviorName": "Disconnects_ML_behavior",
        "modelStatus": "PENDING_BUILD",
        "datapointsCollectionPercentage": 0.0
    }
]
}

```

For more information, see [GetBehaviorModelTrainingSummaries \(Detect Commands\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [GetBehaviorModelTrainingSummaries](#) in *AWS CLI Command Reference*.

get-cardinality

The following code example shows how to use `get-cardinality`.

AWS CLI

To return the approximate count of unique values that match the query

You can use the following setup script to create 10 things representing 10 temperature sensors. Each new thing has 3 attributes.

```

# Bash script. If in other shells, type `bash` before running
Temperatures=(70 71 72 73 74 75 47 97 98 99)
Racks=(Rack1 Rack1 Rack2 Rack2 Rack3 Rack4 Rack5 Rack6 Rack6 Rack6)
IsNormal=(true true true true true true false false false false)
for ((i=0; i<10 ; i++))
do
    thing=$(aws iot create-thing --thing-name "TempSensor$i" --attribute-payload
attributes="{temperature=${Temperatures[i]},rackId=${Racks[i]},stateNormal=
${IsNormal[i]}}")
    aws iot describe-thing --thing-name "TempSensor$i"
done

```

Example output of the setup script:

```

{
    "version": 1,

```

```
"thingName": "TempSensor0",
"defaultClientId": "TempSensor0",
"attributes": {
  "rackId": "Rack1",
  "stateNormal": "true",
  "temperature": "70"
},
"thingArn": "arn:aws:iot:us-east-1:123456789012:thing/TempSensor0",
"thingId": "example1-90ab-cdef-fedc-ba987example"
}
```

The following `get-cardinality` example queries the 10 sensors created by the setup script and returns the number of racks that have temperature sensors reporting abnormal temperature values. If the temperature value is below 60 or above 80, the temperature sensor is in an abnormal state.

```
aws iot get-cardinality \
  --aggregation-field "attributes.rackId" \
  --query-string "thingName:TempSensor* AND attributes.stateNormal:false"
```

Output:

```
{
  "cardinality": 2
}
```

For more information, see [Querying for Aggregate Data](https://docs.aws.amazon.com/iot/latest/developerguide/index-aggregate.html) in the *AWS IoT Developer Guide*.

- For API details, see [GetCardinality](#) in *AWS CLI Command Reference*.

get-effective-policies

The following code example shows how to use `get-effective-policies`.

AWS CLI

To list the policies that effect a thing

The following `get-effective-policies` example lists the policies that effect the specified thing, including policies attached to any groups to which it belongs.

```
aws iot get-effective-policies \
  --thing-name TemperatureSensor-001 \
  --principal arn:aws:iot:us-
west-2:123456789012:cert/488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142
```

Output:

```
{
  "effectivePolicies": [
    {
      "policyName": "TemperatureSensorPolicy",
      "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/
TemperatureSensorPolicy",
      "policyDocument": "{
        \"Version\": \"2012-10-17\",
        \"Statement\": [
          {
            \"Effect\": \"Allow\",
            \"Action\": [
              \"iot:Publish\",
              \"iot:Receive\"
            ],
            \"Resource\": [
              \"arn:aws:iot:us-west-2:123456789012:topic/topic_1\",
              \"arn:aws:iot:us-west-2:123456789012:topic/topic_2\"
            ]
          },
          {
            \"Effect\": \"Allow\",
            \"Action\": [
              \"iot:Subscribe\"
            ],
            \"Resource\": [
              \"arn:aws:iot:us-west-2:123456789012:topicfilter/
topic_1\",
              \"arn:aws:iot:us-west-2:123456789012:topicfilter/
topic_2\"
            ]
          },
          {
            \"Effect\": \"Allow\",
            \"Action\": [
              \"iot:Connect\"
            ],
            \"Resource\": [
              \"arn:aws:iot:us-west-2:123456789012:topicfilter/
topic_1\",
              \"arn:aws:iot:us-west-2:123456789012:topicfilter/
topic_2\"
            ]
          }
        ]
      }
    }
  ]
}
```



```
    ],
    \"Resource\": [
        \"arn:aws:iot:us-west-2:123456789012:client/basicPubSub
    ]
}
]
}
]
}
```

For more information, see [Get Effective Policies for a Thing](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetEffectivePolicies](#) in *AWS CLI Command Reference*.

get-indexing-configuration

The following code example shows how to use `get-indexing-configuration`.

AWS CLI

To get the thing indexing configuration

The following `get-indexing-configuration` example gets the current configuration data for AWS IoT fleet indexing.

```
aws iot get-indexing-configuration
```

Output:

```
{
  "thingIndexingConfiguration": {
    "thingIndexingMode": "OFF",
    "thingConnectivityIndexingMode": "OFF"
  },
  "thingGroupIndexingConfiguration": {
    "thingGroupIndexingMode": "OFF"
  }
}
```

For more information, see [Managing Thing Indexing](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetIndexingConfiguration](#) in *AWS CLI Command Reference*.

get-job-document

The following code example shows how to use `get-job-document`.

AWS CLI

To retrieve the document for a job

The following `get-job-document` example displays details about the document for the job whose ID is `example-job-01`.

```
aws iot get-job-document \
  --job-id "example-job-01"
```

Output:

```
{
  "document": "\n{\n  \"operation\": \"customJob\", \n  \"otherInfo\": \n\"someValue\"\n}\n"
}
```

For more information, see [Creating and Managing Jobs \(CLI\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [GetJobDocument](#) in *AWS CLI Command Reference*.

get-logging-options

The following code example shows how to use `get-logging-options`.

AWS CLI

To get the logging options

The following `get-logging-options` example gets the current logging options for your AWS account.

```
aws iot get-logging-options
```

Output:

```
{
  "roleArn": "arn:aws:iam::123456789012:role/service-role/iotLoggingRole",
  "logLevel": "ERROR"
}
```

For more information, see title in the *AWS IoT Developer Guide*.

- For API details, see [GetLoggingOptions](#) in *AWS CLI Command Reference*.

get-ota-update

The following code example shows how to use `get-ota-update`.

AWS CLI**To retrieve information about an OTA Update**

The following `get-ota-update` example displays details about the specified OTA Update.

```
aws iot get-ota-update \
  --ota-update-id ota12345
```

Output:

```
{
  "otaUpdateInfo": {
    "otaUpdateId": "ota12345",
    "otaUpdateArn": "arn:aws:iot:us-west-2:123456789012:otaupdate/itsaupdate",
    "creationDate": 1557863215.995,
    "lastModifiedDate": 1557863215.995,
    "description": "A critical update needed right away.",
    "targets": [
      "device1",
      "device2",
      "device3",
      "device4"
    ],
    "targetSelection": "SNAPSHOT",
    "protocols": ["HTTP"],
    "awsJobExecutionsRolloutConfig": {
```

```

        "maximumPerMinute": 10
    },
    "otaUpdateFiles": [
        {
            "fileName": "firmware.bin",
            "fileLocation": {
                "stream": {
                    "streamId": "004",
                    "fileId": 123
                }
            },
            "codeSigning": {
                "awsSignerJobId": "48c67f3c-63bb-4f92-a98a-4ee0fbc2bef6"
            }
        }
    ],
    "roleArn": "arn:aws:iam:123456789012:role/service-role/my_ota_role",
    "otaUpdateStatus": "CREATE_COMPLETE",
    "awsIotJobId": "job54321",
    "awsIotJobArn": "arn:aws:iot:us-west-2:123456789012:job/job54321",
    "errorInfo": {
    }
}
}

```

For more information, see [GetOTAUpdate](#) in the *AWS IoT API Reference*.

- For API details, see [GetOtaUpdate](#) in *AWS CLI Command Reference*.

get-percentiles

The following code example shows how to use `get-percentiles`.

AWS CLI

To group the aggregated values that match the query into percentile groupings

You can use the following setup script to create 10 things representing 10 temperature sensors. Each new thing has 1 attribute.

```

# Bash script. If in other shells, type `bash` before running
Temperatures=(70 71 72 73 74 75 47 97 98 99)
for ((i=0; i<10 ; i++))

```

```
do
  thing=$(aws iot create-thing --thing-name "TempSensor$i" --attribute-payload
  attributes="{temperature=${Temperatures[i]}}")
  aws iot describe-thing --thing-name "TempSensor$i"
done
```

Example output of the setup script:

```
{
  "version": 1,
  "thingName": "TempSensor0",
  "defaultClientId": "TempSensor0",
  "attributes": {
    "temperature": "70"
  },
  "thingArn": "arn:aws:iot:us-east-1:123456789012:thing/TempSensor0",
  "thingId": "example1-90ab-cdef-fedc-ba987example"
}
```

The following `get-percentiles` example queries the 10 sensors created by the setup script and returns a value for each percentile group specified. The percentile group "10" contains the aggregated field value that occurs in approximately 10 percent of the values that match the query. In the following output, `{"percent": 10.0, "value": 67.7}` means approximately 10.0% of the temperature values are below 67.7.

```
aws iot get-percentiles \
  --aggregation-field "attributes.temperature" \
  --query-string "thingName:TempSensor*" \
  --percents 10 25 50 75 90
```

Output:

```
{
  "percentiles": [
    {
      "percent": 10.0,
      "value": 67.7
    },
    {
      "percent": 25.0,
      "value": 71.25
    }
  ]
}
```

```
    },
    {
      "percent": 50.0,
      "value": 73.5
    },
    {
      "percent": 75.0,
      "value": 91.5
    },
    {
      "percent": 90.0,
      "value": 98.1
    }
  ]
}
```

For more information, see [Querying for Aggregate Data](#) in the *AWS IoT Developer Guide*.

- For API details, see [GetPercentiles](#) in *AWS CLI Command Reference*.

get-policy-version

The following code example shows how to use `get-policy-version`.

AWS CLI

To get information about a specific version of a policy

The following `get-policy-version` example gets information about the first version of the specified policy.

```
aws iot get-policy \
  --policy-name UpdateDeviceCertPolicy
  --policy-version-id "1"
```

Output:

```
{
  "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/UpdateDeviceCertPolicy",
  "policyName": "UpdateDeviceCertPolicy",
  "policyDocument": "{ \"Version\": \"2012-10-17\", \"Statement\": [ { \"Effect\": \"Allow\", \"Action\": \"iot:UpdateCertificate\", \"Resource\": \"*\" } ] }",
```

```
"policyVersionId": "1",
"isDefaultVersion": false,
"creationDate": 1559925941.924,
"lastModifiedDate": 1559926175.458,
"generationId":
"5066f1b6712ce9d2a1e56399771649a272d6a921762fead080e24fe52f24e042"
}
```

For more information, see [AWS IoT Policies](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetPolicyVersion](#) in *AWS CLI Command Reference*.

get-policy

The following code example shows how to use `get-policy`.

AWS CLI

To get information about the default version of a policy

The following `get-policy` example retrieves information about the default version of the specified policy.

```
aws iot get-policy \
  --policy-name UpdateDeviceCertPolicy
```

Output:

```
{
  "policyName": "UpdateDeviceCertPolicy",
  "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/UpdateDeviceCertPolicy",
  "policyDocument": "{ \"Version\": \"2012-10-17\", \"Statement\": [ { \"Effect\": \"Allow\", \"Action\": \"iot:UpdateCertificate\", \"Resource\": \"*\" } ] }",
  "defaultVersionId": "2",
  "creationDate": 1559925941.924,
  "lastModifiedDate": 1559925941.924,
  "generationId":
  "5066f1b6712ce9d2a1e56399771649a272d6a921762fead080e24fe52f24e042"
}
```

For more information, see [AWS IoT Policies](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetPolicy](#) in *AWS CLI Command Reference*.

get-registration-code

The following code example shows how to use `get-registration-code`.

AWS CLI

To get your AWS account-specific registration code

The following `get-registration-code` example retrieves your AWS account-specific registration code.

```
aws iot get-registration-code
```

Output:

```
{
  "registrationCode":
  "15c51ae5e36ba59ba77042df1115862076bea4bd15841c838fcb68d5010a614c"
}
```

For more information, see [Use Your Own Certificate](#) in the *AWS IoT Developer Guide*.

- For API details, see [GetRegistrationCode](#) in *AWS CLI Command Reference*.

get-statistics

The following code example shows how to use `get-statistics`.

AWS CLI

To search the device index for aggregate data

The following `get-statistics` example returns the number of things that have a property called `connectivity.connected` set to `false` (that is, the number of devices that are not connected) in their device shadow.

```
aws iot get-statistics \
  --index-name AWS_Things \
  --query-string "connectivity.connected:false"
```


Output:

```
{
  "statistics": {
    "count": 6
  }
}
```

For more information, see [Getting Statistics About Your Device Fleet](#) in the *AWS IoT Developer Guide*.

- For API details, see [GetStatistics](#) in *AWS CLI Command Reference*.

get-topic-rule-destination

The following code example shows how to use `get-topic-rule-destination`.

AWS CLI**To get a topic rule destination**

The following `get-topic-rule-destination` example gets information about a topic rule destination.

```
aws iot get-topic-rule-destination \
  --arn "arn:aws:iot:us-west-2:123456789012:ruledestination/http/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"
```

Output:

```
{
  "topicRuleDestination": {
    "arn": "arn:aws:iot:us-west-2:123456789012:ruledestination/http/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "status": "DISABLED",
    "httpUrlProperties": {
      "confirmationUrl": "https://example.com"
    }
  }
}
```

For more information, see [Working with topic rule destinations](#) in the *AWS IoT Developer Guide*.

- For API details, see [GetTopicRuleDestination](#) in *AWS CLI Command Reference*.

get-topic-rule

The following code example shows how to use `get-topic-rule`.

AWS CLI

To get information about a rule

The following `get-topic-rule` example gets information about the specified rule.

```
aws iot get-topic-rule \  
  --rule-name MyRPiLowMoistureAlertRule
```

Output:

```
{  
  "ruleArn": "arn:aws:iot:us-west-2:123456789012:rule/MyRPiLowMoistureAlertRule",  
  "rule": {  
    "ruleName": "MyRPiLowMoistureAlertRule",  
    "sql": "SELECT * FROM '$aws/things/MyRPi/shadow/update/accepted' WHERE  
state.reported.moisture = 'low'\n          ",  
    "description": "Sends an alert whenever soil moisture level readings are too  
low.",  
    "createdAt": 1558624363.0,  
    "actions": [  
      {  
        "sns": {  
          "targetArn": "arn:aws:sns:us-  
west-2:123456789012:MyRPiLowMoistureTopic",  
          "roleArn": "arn:aws:iam::123456789012:role/service-role/  
MyRPiLowMoistureTopicRole",  
          "messageFormat": "RAW"  
        }  
      }  
    ],  
    "ruleDisabled": false,  
    "awsIotSqlVersion": "2016-03-23"  
  }  
}
```

For more information, see [Viewing Your Rules](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetTopicRule](#) in *AWS CLI Command Reference*.

get-v2-logging-options

The following code example shows how to use `get-v2-logging-options`.

AWS CLI

To list the current logging options

The following `get-v2-logging-options` example lists the current logging options for AWS IoT.

```
aws iot get-v2-logging-options
```

Output:

```
{
  "roleArn": "arn:aws:iam::094249569039:role/service-role/iotLoggingRole",
  "defaultLogLevel": "WARN",
  "disableAllLogs": false
}
```

For more information, see title in the *AWS IoT Developer Guide*.

- For API details, see [GetV2LoggingOptions](#) in *AWS CLI Command Reference*.

list-active-violations

The following code example shows how to use `list-active-violations`.

AWS CLI

To list the active violations

The following `list-active-violations` example lists all violations for the specified security profile.

```
aws iot list-active-violations \
  --security-profile-name Testprofile
```

Output:

```
{
  "activeViolations": [
    {
      "violationId": "174db59167fa474c80a652ad1583fd44",
      "thingName": "iotconsole-1560269126751-1",
      "securityProfileName": "Testprofile",
      "behavior": {
        "name": "Authorization",
        "metric": "aws:num-authorization-failures",
        "criteria": {
          "comparisonOperator": "greater-than",
          "value": {
            "count": 10
          },
          "durationSeconds": 300,
          "consecutiveDatapointsToAlarm": 1,
          "consecutiveDatapointsToClear": 1
        }
      },
      "lastViolationValue": {
        "count": 0
      },
      "lastViolationTime": 1560293700.0,
      "violationStartTime": 1560279000.0
    },
    {
      "violationId": "c8a9466a093d3b7b35cd44ca58bdbeab",
      "thingName": "TvnQoEoU",
      "securityProfileName": "Testprofile",
      "behavior": {
        "name": "CellularBandwidth",
        "metric": "aws:message-byte-size",
        "criteria": {
          "comparisonOperator": "greater-than",
          "value": {
            "count": 128
          },
          "consecutiveDatapointsToAlarm": 1,
          "consecutiveDatapointsToClear": 1
        }
      },
      "lastViolationValue": {
```

```
        "count": 110
      },
      "lastViolationTime": 1560369000.0,
      "violationStartTime": 1560276600.0
    },
    {
      "violationId": "74aa393adea02e6648f3ac362beed55e",
      "thingName": "iotconsole-1560269232412-2",
      "securityProfileName": "Testprofile",
      "behavior": {
        "name": "Authorization",
        "metric": "aws:num-authorization-failures",
        "criteria": {
          "comparisonOperator": "greater-than",
          "value": {
            "count": 10
          },
        },
        "durationSeconds": 300,
        "consecutiveDatapointsToAlarm": 1,
        "consecutiveDatapointsToClear": 1
      }
    },
    "lastViolationValue": {
      "count": 0
    },
    "lastViolationTime": 1560276600.0,
    "violationStartTime": 1560276600.0
  },
  {
    "violationId": "1e6ab5f7cf39a1466fcd154e1377e406",
    "thingName": "TvnQoEoU",
    "securityProfileName": "Testprofile",
    "behavior": {
      "name": "Authorization",
      "metric": "aws:num-authorization-failures",
      "criteria": {
        "comparisonOperator": "greater-than",
        "value": {
          "count": 10
        },
      },
      "durationSeconds": 300,
      "consecutiveDatapointsToAlarm": 1,
      "consecutiveDatapointsToClear": 1
    }
  }
}
```

```
    },
    "lastViolationValue": {
      "count": 0
    },
    "lastViolationTime": 1560369000.0,
    "violationStartTime": 1560276600.0
  }
]
}
```

- For API details, see [ListActiveViolations](#) in *AWS CLI Command Reference*.

list-attached-policies

The following code example shows how to use `list-attached-policies`.

AWS CLI

Example 1: To list the policies attached to a group

The following `list-attached-policies` example lists the policies that are attached to the specified group.

```
aws iot list-attached-policies \
  --target "arn:aws:iot:us-west-2:123456789012:thinggroup/LightBulbs"
```

Output:

```
{
  "policies": [
    {
      "policyName": "UpdateDeviceCertPolicy",
      "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/
UpdateDeviceCertPolicy"
    }
  ]
}
```

For more information, see [Thing Groups](#) in the *AWS IoT Developers Guide*.

Example 2: To list the policies attached to a device certificate

The following `list-attached-policies` example lists the AWS IoT policies attached to the device certificate. The certificate is identified by its ARN.

```
aws iot list-attached-policies \  
  --target arn:aws:iot:us-  
west-2:123456789012:cert/488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142
```

Output:

```
{  
  "policies": [  
    {  
      "policyName": "TemperatureSensorPolicy",  
      "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/  
TemperatureSensorPolicy"  
    }  
  ]  
}
```

For more information, see [Thing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListAttachedPolicies](#) in *AWS CLI Command Reference*.

list-audit-findings

The following code example shows how to use `list-audit-findings`.

AWS CLI

Example 1: To list all findings from an audit

The following `list-audit-findings` example lists all findings from an AWS IoT Device Defender audit with a specified task ID.

```
aws iot list-audit-findings \  
  --task-id a3aea009955e501a31b764abe1bebd3d
```

Output:

```
{  
  "findings": []
```

```
}
```

Example 2: To list findings for an audit check type

The following `list-audit-findings` example shows findings from AWS IoT Device Defender audits that ran between June 5, 2019 and June 19, 2019 in which devices are sharing a device certificate. When you specify a check name, you must provide a start and end time.

```
aws iot list-audit-findings \  
  --check-name DEVICE_CERTIFICATE_SHARED_CHECK \  
  --start-time 1559747125 \  
  --end-time 1560962028
```

Output:

```
{  
  "findings": [  
    {  
      "taskId": "eeef61068b0eb03c456d746c5a26ee04",  
      "checkName": "DEVICE_CERTIFICATE_SHARED_CHECK",  
      "taskStartTime": 1560161017.172,  
      "findingTime": 1560161017.592,  
      "severity": "CRITICAL",  
      "nonCompliantResource": {  
        "resourceType": "DEVICE_CERTIFICATE",  
        "resourceIdentifier": {  
          "deviceCertificateId":  
"b193ab7162c0fadca83246d24fa090300a1236fe58137e121b011804d8ac1d6b"  
        }  
      },  
      "relatedResources": [  
        {  
          "resourceType": "CLIENT_ID",  
          "resourceIdentifier": {  
            "clientId": "ZipxgAII"  
          },  
          "additionalInfo": {  
            "CONNECTION_TIME": "1560086374068"  
          }  
        },  
        {  
          "resourceType": "CLIENT_ID",  
          "resourceIdentifier": {
```



```

        "clientId": "ZipxgAII"
      },
      "additionalInfo": {
        "CONNECTION_TIME": "1560081552187",
        "DISCONNECTION_TIME": "1560086371552"
      }
    },
    {
      "resourceType": "CLIENT_ID",
      "resourceIdentifier": {
        "clientId": "ZipxgAII"
      },
      "additionalInfo": {
        "CONNECTION_TIME": "1559289863631",
        "DISCONNECTION_TIME": "1560081532716"
      }
    }
  ],
  "reasonForNonCompliance": "Certificate shared by one or more devices.",
  "reasonForNonComplianceCode": "CERTIFICATE_SHARED_BY_MULTIPLE_DEVICES"
},
{
  "taskId": "bade6b5efd2e1b1569822f6021b39cf5",
  "checkName": "DEVICE_CERTIFICATE_SHARED_CHECK",
  "taskStartTime": 1559988217.27,
  "findingTime": 1559988217.655,
  "severity": "CRITICAL",
  "nonCompliantResource": {
    "resourceType": "DEVICE_CERTIFICATE",
    "resourceIdentifier": {
      "deviceCertificateId":
"b193ab7162c0fadca83246d24fa090300a1236fe58137e121b011804d8ac1d6b"
    }
  },
  "relatedResources": [
    {
      "resourceType": "CLIENT_ID",
      "resourceIdentifier": {
        "clientId": "xShGENLW"
      },
      "additionalInfo": {
        "CONNECTION_TIME": "1559972350825"
      }
    }
  ],

```

```

        {
            "resourceType": "CLIENT_ID",
            "resourceIdentifier": {
                "clientId": "xShGENLW"
            },
            "additionalInfo": {
                "CONNECTION_TIME": "1559255062002",
                "DISCONNECTION_TIME": "1559972350616"
            }
        }
    ],
    "reasonForNonCompliance": "Certificate shared by one or more devices.",
    "reasonForNonComplianceCode": "CERTIFICATE_SHARED_BY_MULTIPLE_DEVICES"
},
{
    "taskId": "c23f6233ba2d35879c4bb2810fb5ffd6",
    "checkName": "DEVICE_CERTIFICATE_SHARED_CHECK",
    "taskStartTime": 1559901817.31,
    "findingTime": 1559901817.767,
    "severity": "CRITICAL",
    "nonCompliantResource": {
        "resourceType": "DEVICE_CERTIFICATE",
        "resourceIdentifier": {
            "deviceCertificateId":
"b193ab7162c0fadca83246d24fa090300a1236fe58137e121b011804d8ac1d6b"
        }
    },
    "relatedResources": [
        {
            "resourceType": "CLIENT_ID",
            "resourceIdentifier": {
                "clientId": "TvnQoEoU"
            },
            "additionalInfo": {
                "CONNECTION_TIME": "1559826729768"
            }
        },
        {
            "resourceType": "CLIENT_ID",
            "resourceIdentifier": {
                "clientId": "TvnQoEoU"
            },
            "additionalInfo": {
                "CONNECTION_TIME": "1559345920964",

```

```

        "DISCONNECTION_TIME": "1559826728402"
      }
    }
  ],
  "reasonForNonCompliance": "Certificate shared by one or more devices.",
  "reasonForNonComplianceCode": "CERTIFICATE_SHARED_BY_MULTIPLE_DEVICES"
}
]
}

```

For more information, see [Audit Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListAuditFindings](#) in *AWS CLI Command Reference*.

list-audit-mitigation-actions-executions

The following code example shows how to use `list-audit-mitigation-actions-executions`.

AWS CLI

To list the details of an audit mitigation action execution

An audit mitigation action task applies a mitigation action to one or more findings from an AWS IoT Device Defender audit. The following `list-audit-mitigation-actions-executions` example lists the details for the mitigation action task with the specified `taskId` and for the specified finding.

```

aws iot list-audit-mitigation-actions-executions \
  --task-id myActionsTaskId \
  --finding-id 0edbaaec-2fe1-4cf5-abc9-d4c3e51f7464

```

Output:

```

{
  "actionsExecutions": [
    {
      "taskId": "myActionsTaskId",
      "findingId": "0edbaaec-2fe1-4cf5-abc9-d4c3e51f7464",
      "actionName": "ResetPolicyVersionAction",
      "actionId": "1ea0b415-bef1-4a01-bd13-72fb63c59afb",
    }
  ]
}

```

```

        "status": "COMPLETED",
        "startTime": "2019-12-10T15:19:13.279000-08:00",
        "endTime": "2019-12-10T15:19:13.337000-08:00"
    }
]
}

```

For more information, see [ListAuditMitigationActionsExecutions \(Mitigation Action Commands\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListAuditMitigationActionsExecutions](#) in *AWS CLI Command Reference*.

list-audit-mitigation-actions-tasks

The following code example shows how to use `list-audit-mitigation-actions-tasks`.

AWS CLI

To list audit mitigation action tasks

The following `list-audit-mitigation-actions-tasks` example lists the mitigation actions that were applied to findings within the specified time period.

```

aws iot list-audit-mitigation-actions-tasks \
  --start-time 1594157400 \
  --end-time 1594157430

```

Output:

```

{
  "tasks": [
    {
      "taskId": "0062f2d6-3999-488f-88c7-bef005414103",
      "startTime": "2020-07-07T14:30:15.172000-07:00",
      "taskStatus": "COMPLETED"
    }
  ]
}

```

For more information, see [ListAuditMitigationActionsTasks \(Mitigation Action Commands\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListAuditMitigationActionsTasks](#) in *AWS CLI Command Reference*.

list-audit-suppressions

The following code example shows how to use `list-audit-suppressions`.

AWS CLI

To list all audit finding suppressions

The following `list-audit-suppressions` example lists all active audit finding suppressions.

```
aws iot list-audit-suppressions
```

Output:

```
{
  "suppressions": [
    {
      "checkName": "DEVICE_CERTIFICATE_EXPIRING_CHECK",
      "resourceIdentifier": {
        "deviceCertificateId": "c7691e<shortened>"
      },
      "expirationDate": 1597881600.0,
      "suppressIndefinitely": false
    }
  ]
}
```

For more information, see [Audit finding suppressions](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListAuditSuppressions](#) in *AWS CLI Command Reference*.

list-audit-tasks

The following code example shows how to use `list-audit-tasks`.

AWS CLI

To list all findings from an audit

The following `list-audit-tasks` example lists the audit tasks that ran between June 5, 2019 and June 12, 2019.

```
aws iot list-audit-tasks \  
  --start-time 1559747125 \  
  --end-time 1560357228
```

Output:

```
{  
  "tasks": [  
    {  
      "taskId": "a3aea009955e501a31b764abe1bebd3d",  
      "taskStatus": "COMPLETED",  
      "taskType": "ON_DEMAND_AUDIT_TASK"  
    },  
    {  
      "taskId": "f76b4b5102b632cd9ae38a279c266da1",  
      "taskStatus": "COMPLETED",  
      "taskType": "SCHEDULED_AUDIT_TASK"  
    },  
    {  
      "taskId": "51d9967d9f9ff4d26529505f6d2c444a",  
      "taskStatus": "COMPLETED",  
      "taskType": "SCHEDULED_AUDIT_TASK"  
    },  
    {  
      "taskId": "eef61068b0eb03c456d746c5a26ee04",  
      "taskStatus": "COMPLETED",  
      "taskType": "SCHEDULED_AUDIT_TASK"  
    },  
    {  
      "taskId": "041c49557b7c7b04c079a49514b55589",  
      "taskStatus": "COMPLETED",  
      "taskType": "SCHEDULED_AUDIT_TASK"  
    },  
    {  
      "taskId": "82c7f2afac1562d18a4560be73998acc",  
      "taskStatus": "COMPLETED",  
      "taskType": "SCHEDULED_AUDIT_TASK"  
    },  
    {  
      "taskId": "bade6b5efd2e1b1569822f6021b39cf5",
```

```
    "taskStatus": "COMPLETED",
    "taskType": "SCHEDULED_AUDIT_TASK"
  },
  {
    "taskId": "c23f6233ba2d35879c4bb2810fb5ffd6",
    "taskStatus": "COMPLETED",
    "taskType": "SCHEDULED_AUDIT_TASK"
  },
  {
    "taskId": "ac9086b7222a2f5e2e17bb6fd30b3aeb",
    "taskStatus": "COMPLETED",
    "taskType": "SCHEDULED_AUDIT_TASK"
  }
]
}
```

For more information, see [Audit Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListAuditTasks](#) in *AWS CLI Command Reference*.

list-authorizers

The following code example shows how to use `list-authorizers`.

AWS CLI

To list your custom authorizer

The following `list-authorizers` example lists the custom authorizers in your AWS account.

```
aws iot list-authorizers
```

Output:

```
{
  "authorizers": [
    {
      "authorizerName": "CustomAuthorizer",
      "authorizerArn": "arn:aws:iot:us-west-2:123456789012:authorizer/
CustomAuthorizer"
    },
    {
```

```
        "authorizerName": "CustomAuthorizer2",
        "authorizerArn": "arn:aws:iot:us-west-2:123456789012:authorizer/
CustomAuthorizer2"
    },
    {
        "authorizerName": "CustomAuthorizer3",
        "authorizerArn": "arn:aws:iot:us-west-2:123456789012:authorizer/
CustomAuthorizer3"
    }
]
}
```

For more information, see [ListAuthorizers](#) in the *AWS IoT API Reference*.

- For API details, see [ListAuthorizers](#) in *AWS CLI Command Reference*.

list-billing-groups

The following code example shows how to use `list-billing-groups`.

AWS CLI

To list the billing groups for your AWS account and region

The following `list-billing-groups` example lists all billing groups that are defined for your AWS account and AWS Region.

```
aws iot list-billing-groups
```

Output:

```
{
  "billingGroups": [
    {
      "groupName": "GroupOne",
      "groupArn": "arn:aws:iot:us-west-2:123456789012:billinggroup/GroupOne"
    }
  ]
}
```

For more information, see [Billing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListBillingGroups](#) in *AWS CLI Command Reference*.

list-ca-certificates

The following code example shows how to use `list-ca-certificates`.

AWS CLI

To list the CA certificates registered in your AWS account

The following `list-ca-certificates` example lists the CA certificates registered in your AWS account.

```
aws iot list-ca-certificates
```

Output:

```
{
  "certificates": [
    {
      "certificateArn": "arn:aws:iot:us-west-2:123456789012:cacert/
f4efed62c0142f16af278166f61962501165c4f0536295207426460058cd1467",
      "certificateId":
"f4efed62c0142f16af278166f61962501165c4f0536295207426460058cd1467",
      "status": "INACTIVE",
      "creationDate": 1569365372.053
    }
  ]
}
```

For more information, see [Use Your Own Certificate](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListCaCertificates](#) in *AWS CLI Command Reference*.

list-certificates-by-ca

The following code example shows how to use `list-certificates-by-ca`.

AWS CLI

To list all device certificates signed with a CA certificate

The following `list-certificates-by-ca` example lists all device certificates in your AWS account that are signed with the specified CA certificate.

```
aws iot list-certificates-by-ca \
  --ca-certificate-id
  f4efed62c0142f16af278166f61962501165c4f0536295207426460058cd1467
```

Output:

```
{
  "certificates": [
    {
      "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142",
      "certificateId":
"488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142",
      "status": "ACTIVE",
      "creationDate": 1569363250.557
    }
  ]
}
```

For more information, see [ListCertificatesByCA](#) in the *AWS IoT API Reference*.

- For API details, see [ListCertificatesByCa](#) in *AWS CLI Command Reference*.

list-certificates

The following code example shows how to use `list-certificates`.

AWS CLI

Example 1: To list the certificates registered in your AWS account

The following `list-certificates` example lists all certificates registered in your account. If you have more than the default paging limit of 25, you can use the `nextMarker` response value from this command and supply it to the next command to get the next batch of results. Repeat until `nextMarker` returns without a value.

```
aws iot list-certificates
```

Output:

```
{
  "certificates": [
    {
      "certificateArn": "arn:aws:iot:us-west-2:123456789012:cert/604c48437a57b7d5fc5d137c5be75011c6ee67c9a6943683a1acb4b1626bac36",
      "certificateId": "604c48437a57b7d5fc5d137c5be75011c6ee67c9a6943683a1acb4b1626bac36",
      "status": "ACTIVE",
      "creationDate": 1556810537.617
    },
    {
      "certificateArn": "arn:aws:iot:us-west-2:123456789012:cert/262a1ac8a7d8aa72f6e96e365480f7313aa9db74b8339ec65d34dc3074e1c31e",
      "certificateId": "262a1ac8a7d8aa72f6e96e365480f7313aa9db74b8339ec65d34dc3074e1c31e",
      "status": "ACTIVE",
      "creationDate": 1546447050.885
    },
    {
      "certificateArn": "arn:aws:iot:us-west-2:123456789012:cert/b193ab7162c0fadca83246d24fa090300a1236fe58137e121b011804d8ac1d6b",
      "certificateId": "b193ab7162c0fadca83246d24fa090300a1236fe58137e121b011804d8ac1d6b",
      "status": "ACTIVE",
      "creationDate": 1546292258.322
    },
    {
      "certificateArn": "arn:aws:iot:us-west-2:123456789012:cert/7aebeea3845d14a44ec80b06b8b78a89f3f8a706974b8b34d18f5adf0741db42",
      "certificateId": "7aebeea3845d14a44ec80b06b8b78a89f3f8a706974b8b34d18f5adf0741db42",
      "status": "ACTIVE",
      "creationDate": 1541457693.453
    },
    {
      "certificateArn": "arn:aws:iot:us-west-2:123456789012:cert/54458aa39ebb3eb39c91ffbbdcc3a6ca1c7c094d1644b889f735a6fc2cd9a7e3",
      "certificateId": "54458aa39ebb3eb39c91ffbbdcc3a6ca1c7c094d1644b889f735a6fc2cd9a7e3",
      "status": "ACTIVE",
      "creationDate": 1541113568.611
    },
  ],
}
```

```
{
  "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/4f0ba725787aa94d67d2fca420eca022242532e8b3c58e7465c7778b443fd65e",
  "certificateId":
"4f0ba725787aa94d67d2fca420eca022242532e8b3c58e7465c7778b443fd65e",
  "status": "ACTIVE",
  "creationDate": 1541022751.983
}
]
```

- For API details, see [ListCertificates](#) in *AWS CLI Command Reference*.

list-custom-metrics

The following code example shows how to use `list-custom-metrics`.

AWS CLI

To list your custom metrics

The following `list-custom-metrics` example lists all of your custom metrics.

```
aws iot list-custom-metrics \
  --region us-east-1
```

Output:

```
{
  "metricNames": [
    "batteryPercentage"
  ]
}
```

For more information, see [Custom metrics](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [ListCustomMetrics](#) in *AWS CLI Command Reference*.

list-dimensions

The following code example shows how to use `list-dimensions`.

AWS CLI

To list the dimensions for your AWS account

The following `list-dimensions` example lists all AWS IoT Device Defender dimensions that are defined in your AWS account.

```
aws iot list-dimensions
```

Output:

```
{
  "dimensionNames": [
    "TopicFilterForAuthMessages",
    "TopicFilterForActivityMessages"
  ]
}
```

For more information, see [Detect Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListDimensions](#) in *AWS CLI Command Reference*.

list-domain-configurations

The following code example shows how to use `list-domain-configurations`.

AWS CLI

To list domain configurations

The following `list-domain-configurations` example lists the domain configurations in your AWS account that have the specified service type.

```
aws iot list-domain-configurations \
  --service-type "DATA"
```

Output:

```
{
  "domainConfigurations":
```

```
[
  {
    "domainConfigurationName": "additionalDataDomain",
    "domainConfigurationArn": "arn:aws:iot:us-
west-2:123456789012:domainconfiguration/additionalDataDomain/dikMh",
    "serviceType": "DATA"
  },
  {
    "domainConfigurationName": "iot:Jobs",
    "domainConfigurationArn": "arn:aws:iot:us-
west-2:123456789012:domainconfiguration/iot:Jobs",
    "serviceType": "JOBS"
  },
  {
    "domainConfigurationName": "iot:Data-ATS",
    "domainConfigurationArn": "arn:aws:iot:us-
west-2:123456789012:domainconfiguration/iot:Data-ATS",
    "serviceType": "DATA"
  },
  {
    "domainConfigurationName": "iot:CredentialProvider",
    "domainConfigurationArn": "arn:aws:iot:us-
west-2:123456789012:domainconfiguration/iot:CredentialProvider",
    "serviceType": "CREDENTIAL_PROVIDER"
  }
]
```

For more information, see [Configurable Endpoints](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListDomainConfigurations](#) in *AWS CLI Command Reference*.

list-indices

The following code example shows how to use `list-indices`.

AWS CLI

To list the configured search indices

The following `list-indices` example lists all configured search indices in your AWS account. If you have not enabled thing indexing, you might not have any indices.

```
aws iot list-indices
```

Output:

```
{
  "indexNames": [
    "AWS_Things"
  ]
}
```

For more information, see [Managing Thing Indexing](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListIndices](#) in *AWS CLI Command Reference*.

list-job-executions-for-job

The following code example shows how to use `list-job-executions-for-job`.

AWS CLI

To list the jobs in your AWS account

The following `list-job-executions-for-job` example lists all job executions for a job in your AWS account, specified by the `jobId`.

```
aws iot list-job-executions-for-job \
  --job-id my-ota-job
```

Output:

```
{
  "executionSummaries": [
    {
      "thingArn": "arn:aws:iot:us-east-1:123456789012:thing/my_thing",
      "jobExecutionSummary": {
        "status": "QUEUED",
        "queuedAt": "2022-03-07T15:58:42.195000-08:00",
        "lastUpdatedAt": "2022-03-07T15:58:42.195000-08:00",
        "executionNumber": 1,
        "retryAttempt": 0
      }
    }
  ]
}
```

```
    }
  }
]
```

For more information, see [Creating and Managing Jobs \(CLI\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListJobExecutionsForJob](#) in *AWS CLI Command Reference*.

list-job-executions-for-thing

The following code example shows how to use `list-job-executions-for-thing`.

AWS CLI

To list the jobs that were executed for a thing

The following `list-job-executions-for-thing` example lists all jobs that were executed for the thing named `MyRaspberryPi`.

```
aws iot list-job-executions-for-thing \
  --thing-name "MyRaspberryPi"
```

Output:

```
{
  "executionSummaries": [
    {
      "jobId": "example-job-01",
      "jobExecutionSummary": {
        "status": "QUEUED",
        "queuedAt": 1560787023.636,
        "lastUpdatedAt": 1560787023.636,
        "executionNumber": 1
      }
    }
  ]
}
```

For more information, see [Creating and Managing Jobs \(CLI\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListJobExecutionsForThing](#) in *AWS CLI Command Reference*.

list-jobs

The following code example shows how to use `list-jobs`.

AWS CLI

To list the jobs in your AWS account

The following `list-jobs` example lists all jobs in your AWS account, sorted by the job status.

```
aws iot list-jobs
```

Output:

```
{
  "jobs": [
    {
      "jobArn": "arn:aws:iot:us-west-2:123456789012:job/example-job-01",
      "jobId": "example-job-01",
      "targetSelection": "SNAPSHOT",
      "status": "IN_PROGRESS",
      "createdAt": 1560787022.733,
      "lastUpdatedAt": 1560787026.294
    }
  ]
}
```

For more information, see [Creating and Managing Jobs \(CLI\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListJobs](#) in *AWS CLI Command Reference*.

list-mitigation-actions

The following code example shows how to use `list-mitigation-actions`.

AWS CLI

To list all defined mitigation actions

The following `list-mitigation-actions` example lists all defined mitigation actions for your AWS account and Region. For each action, the name, ARN, and creation date are listed.

```
aws iot list-mitigation-actions
```

Output:

```
{
  "actionIdentifiers": [
    {
      "actionName": "DeactivateCACertAction",
      "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/DeactivateCACertAction",
      "creationDate": "2019-12-10T11:12:47.574000-08:00"
    },
    {
      "actionName": "ResetPolicyVersionAction",
      "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/ResetPolicyVersionAction",
      "creationDate": "2019-12-10T11:11:48.920000-08:00"
    },
    {
      "actionName": "PublishFindingToSNSAction",
      "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/PublishFindingToSNSAction",
      "creationDate": "2019-12-10T11:10:49.546000-08:00"
    },
    {
      "actionName": "AddThingsToQuarantineGroupAction",
      "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/AddThingsToQuarantineGroupAction",
      "creationDate": "2019-12-10T11:09:35.999000-08:00"
    },
    {
      "actionName": "UpdateDeviceCertAction",
      "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/UpdateDeviceCertAction",
      "creationDate": "2019-12-10T11:08:44.263000-08:00"
    },
    {
      "actionName": "SampleMitigationAction",
      "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/SampleMitigationAction",
      "creationDate": "2019-12-10T11:03:41.840000-08:00"
    }
  ]
}
```

```
}
```

For more information, see [ListMitigationActions \(Mitigation Action Commands\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListMitigationActions](#) in *AWS CLI Command Reference*.

list-mitigations-actions

The following code example shows how to use `list-mitigations-actions`.

AWS CLI

To list all defined mitigation actions

The following `list-mitigations-actions` example lists all defined mitigation actions for your AWS account and Region. For each action, the name, ARN, and creation date are listed.

```
aws iot list-mitigation-actions
```

Output:

```
{
  "actionIdentifiers": [
    {
      "actionName": "DeactivateCACertAction",
      "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/DeactivateCACertAction",
      "creationDate": "2019-12-10T11:12:47.574000-08:00"
    },
    {
      "actionName": "ResetPolicyVersionAction",
      "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/ResetPolicyVersionAction",
      "creationDate": "2019-12-10T11:11:48.920000-08:00"
    },
    {
      "actionName": "PublishFindingToSNSAction",
      "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/PublishFindingToSNSAction",
      "creationDate": "2019-12-10T11:10:49.546000-08:00"
    },
    {
```

```

        "actionName": "AddThingsToQuarantineGroupAction",
        "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
AddThingsToQuarantineGroupAction",
        "creationDate": "2019-12-10T11:09:35.999000-08:00"
    },
    {
        "actionName": "UpdateDeviceCertAction",
        "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
UpdateDeviceCertAction",
        "creationDate": "2019-12-10T11:08:44.263000-08:00"
    },
    {
        "actionName": "SampleMitigationAction",
        "actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
SampleMitigationAction",
        "creationDate": "2019-12-10T11:03:41.840000-08:00"
    }
]
}

```

For more information, see [ListMitigationActions \(Mitigation Action Commands\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListMitigationsActions](#) in *AWS CLI Command Reference*.

list-ota-updates

The following code example shows how to use `list-ota-updates`.

AWS CLI

To list OTA Updates for the account

The following `list-ota-updates` example lists the available OTA updates.

```
aws iot list-ota-updates
```

Output:

```

{
  "otaUpdates": [
    {
      "otaUpdateId": "itsaupdate",

```

```
        "otaUpdateArn": "arn:aws:iot:us-west-2:123456789012:otaupdate/itsaupdate",
        "creationDate": 1557863215.995
    }
]
}
```

For more information, see [ListOTAUpdates](#) in the *AWS IoT API Reference*.

- For API details, see [ListOtaUpdates](#) in *AWS CLI Command Reference*.

list-outgoing-certificates

The following code example shows how to use `list-outgoing-certificates`.

AWS CLI

To list certificates being transferred to a different AWS account

The following `list-outgoing-certificates` example lists all device certificates that are in the process of being transferred to a different AWS account using the `transfer-certificate` command.

```
aws iot list-outgoing-certificates
```

Output:

```
{
  "outgoingCertificates": [
    {
      "certificateArn": "arn:aws:iot:us-west-2:030714055129:cert/488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142",
      "certificateId": "488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142",
      "transferredTo": "030714055129",
      "transferDate": 1569427780.441,
      "creationDate": 1569363250.557
    }
  ]
}
```

For more information, see [ListOutgoingCertificates](#) in the *AWS IoT API Reference*.

- For API details, see [ListOutgoingCertificates](#) in *AWS CLI Command Reference*.

list-policies

The following code example shows how to use `list-policies`.

AWS CLI

To list the policies defined in your AWS account

The following `list-policies` example lists all policies defined in your AWS account.

```
aws iot list-policies
```

Output:

```
{
  "policies": [
    {
      "policyName": "UpdateDeviceCertPolicy",
      "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/UpdateDeviceCertPolicy"
    },
    {
      "policyName": "PlantIoTPolicy",
      "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/PlantIoTPolicy"
    },
    {
      "policyName": "MyPiGroup_Core-policy",
      "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/MyPiGroup_Core-policy"
    }
  ]
}
```

For more information, see [AWS IoT Policies](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListPolicies](#) in *AWS CLI Command Reference*.

list-policy-versions

The following code example shows how to use `list-policy-versions`.

AWS CLI

Example 1: To see all versions of a policy

The following `list-policy-versions` example lists all versions of the specified policy and their creation dates.

```
aws iot list-policy-versions \  
  --policy-name LightBulbPolicy
```

Output:

```
{  
  "policyVersions": [  
    {  
      "versionId": "2",  
      "isDefaultVersion": true,  
      "createDate": 1559925941.924  
    },  
    {  
      "versionId": "1",  
      "isDefaultVersion": false,  
      "createDate": 1559925941.924  
    }  
  ]  
}
```

For more information, see [AWS IoT Policies](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListPolicyVersions](#) in *AWS CLI Command Reference*.

list-principal-things

The following code example shows how to use `list-principal-things`.

AWS CLI

To list the things attached with a principal

The following `list-principal-things` example lists the things attached to the principal specified by an ARN.

```
aws iot list-principal-things \  
  --principal arn:aws:iot:us-  
west-2:123456789012:cert/2e1eb273792174ec2b9bf4e9b37e6c6c692345499506002a35159767055278e8
```

Output:

```
{  
  "things": [  
    "DeskLamp",  
    "TableLamp"  
  ]  
}
```

For more information, see [ListPrincipalThings](#) in the *AWS IoT API Reference*.

- For API details, see [ListPrincipalThings](#) in *AWS CLI Command Reference*.

list-provisioning-template-versions

The following code example shows how to use `list-provisioning-template-versions`.

AWS CLI

To list provisioning template versions

The following `list-provisioning-template-versions` example lists the available versions of the specified provisioning template.

```
aws iot list-provisioning-template-versions \  
  --template-name "widget-template"
```

Output:

```
{  
  "versions": [  
    {  
      "versionId": 1,  
      "creationDate": 1574800471.339,  
      "isDefaultVersion": true  
    },  
    {
```



```
        "versionId": 2,  
        "creationDate": 1574801192.317,  
        "isDefaultVersion": false  
      }  
    ]  
  }  
}
```

For more information, see [AWS IoT Secure Tunneling](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [ListProvisioningTemplateVersions](#) in *AWS CLI Command Reference*.

list-provisioning-templates

The following code example shows how to use `list-provisioning-templates`.

AWS CLI

To list provisioning templates

The following `list-provisioning-templates` example lists all of the provisioning templates in your AWS account.

```
aws iot list-provisioning-templates
```

Output:

```
{  
  "templates": [  
    {  
      "templateArn": "arn:aws:iot:us-east-1:123456789012:provisioningtemplate/  
widget-template",  
      "templateName": "widget-template",  
      "description": "A provisioning template for widgets",  
      "creationDate": 1574800471.367,  
      "lastModifiedDate": 1574801192.324,  
      "enabled": false  
    }  
  ]  
}
```

For more information, see [AWS IoT Secure Tunneling](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [ListProvisioningTemplates](#) in *AWS CLI Command Reference*.

list-role-aliases

The following code example shows how to use `list-role-aliases`.

AWS CLI

To list the AWS IoT role aliases in your AWS account

The following `list-role-aliases` example lists the AWS IoT role aliases in your AWS account.

```
aws iot list-role-aliases
```

Output:

```
{
  "roleAliases": [
    "ResidentAlias",
    "ElectricianAlias"
  ]
}
```

For more information, see [ListRoleAliases](#) in the *AWS IoT API Reference*.

- For API details, see [ListRoleAliases](#) in *AWS CLI Command Reference*.

list-scheduled-audits

The following code example shows how to use `list-scheduled-audits`.

AWS CLI

To list the scheduled audits for your AWS account

The following `list-scheduled-audits` example lists any audits scheduled for your AWS account.

```
aws iot list-scheduled-audits
```

Output:

```
{
  "scheduledAudits": [
    {
      "scheduledAuditName": "AWSIoTDeviceDefenderDailyAudit",
      "scheduledAuditArn": "arn:aws:iot:us-west-2:123456789012:scheduledaudit/
AWSIoTDeviceDefenderDailyAudit",
      "frequency": "DAILY"
    },
    {
      "scheduledAuditName": "AWSDeviceDefenderWeeklyAudit",
      "scheduledAuditArn": "arn:aws:iot:us-west-2:123456789012:scheduledaudit/
AWSDeviceDefenderWeeklyAudit",
      "frequency": "WEEKLY",
      "dayOfWeek": "SUN"
    }
  ]
}
```

For more information, see [Audit Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListScheduledAudits](#) in *AWS CLI Command Reference*.

list-security-profiles-for-target

The following code example shows how to use `list-security-profiles-for-target`.

AWS CLI

To list the security profiles attached to a target

The following `list-security-profiles-for-target` example lists the AWS IoT Device Defender security profiles that are attached to unregistered devices.

```
aws iot list-security-profiles-for-target \
  --security-profile-target-arn "arn:aws:iot:us-west-2:123456789012:all/
unregistered-things"
```

Output:

```
{
  "securityProfileTargetMappings": [
    {
```

```
    "securityProfileIdentifier": {
      "name": "Testprofile",
      "arn": "arn:aws:iot:us-west-2:123456789012:securityprofile/
Testprofile"
    },
    "target": {
      "arn": "arn:aws:iot:us-west-2:123456789012:all/unregistered-things"
    }
  }
]
```

For more information, see [Detect Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListSecurityProfilesForTarget](#) in *AWS CLI Command Reference*.

list-security-profiles

The following code example shows how to use `list-security-profiles`.

AWS CLI

To list the security profiles for your AWS account

The following `list-security-profiles` example lists all AWS IoT Device Defender security profiles that are defined in your AWS account.

```
aws iot list-security-profiles
```

Output:

```
{
  "securityProfileIdentifiers": [
    {
      "name": "Testprofile",
      "arn": "arn:aws:iot:us-west-2:123456789012:securityprofile/Testprofile"
    }
  ]
}
```

For more information, see [Detect Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListSecurityProfiles](#) in *AWS CLI Command Reference*.

list-streams

The following code example shows how to use `list-streams`.

AWS CLI

To list the streams in the account

The following `list-streams` example lists all of the streams in your AWS account.

```
aws iot list-streams
```

Output:

```
{
  "streams": [
    {
      "streamId": "stream12345",
      "streamArn": "arn:aws:iot:us-west-2:123456789012:stream/stream12345",
      "streamVersion": 1,
      "description": "This stream is used for Amazon FreeRTOS OTA Update
12345."
    },
    {
      "streamId": "stream54321",
      "streamArn": "arn:aws:iot:us-west-2:123456789012:stream/stream54321",
      "streamVersion": 1,
      "description": "This stream is used for Amazon FreeRTOS OTA Update
54321."
    }
  ]
}
```

For more information, see [ListStreams](#) in the *AWS IoT API Reference*.

- For API details, see [ListStreams](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To display the tags and their values associated with a resource

The following `list-tags-for-resource` example displays the tags and values associated with the thing group `LightBulbs`.

```
aws iot list-tags-for-resource \  
  --resource-arn "arn:aws:iot:us-west-2:094249569039:thinggroup/LightBulbs"
```

Output:

```
{  
  "tags": [  
    {  
      "Key": "Assembly",  
      "Value": "Fact1NW"  
    },  
    {  
      "Key": "MyTag",  
      "Value": "777"  
    }  
  ]  
}
```

For more information, see [Tagging Your AWS IoT Resources](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

list-targets-for-policy

The following code example shows how to use `list-targets-for-policy`.

AWS CLI

To list the principals associated with an AWS IoT policy

The following `list-targets-for-policy` example lists the device certificates to which the specified policy is attached.

```
aws iot list-targets-for-policy \  
  --policy-name "PolicyName"
```

```
--policy-name UpdateDeviceCertPolicy
```

Output:

```
{
  "targets": [
    "arn:aws:iot:us-west-2:123456789012:cert/488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142",
    "arn:aws:iot:us-west-2:123456789012:cert/d1eb269fb55a628552143c8f96eb3c258fcd5331ea113e766ba0c82bf225f0be"
  ]
}
```

For more information, see [Thing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListTargetsForPolicy](#) in *AWS CLI Command Reference*.

list-targets-for-security-profile

The following code example shows how to use `list-targets-for-security-profile`.

AWS CLI

To list the targets to which a security profile is applied

The following `list-targets-for-security-profile` example lists the targets to which the AWS IoT Device Defender security profile named `PossibleIssue` is applied.

```
aws iot list-targets-for-security-profile \
  --security-profile-name Testprofile
```

Output:

```
{
  "securityProfileTargets": [
    {
      "arn": "arn:aws:iot:us-west-2:123456789012:all/unregistered-things"
    },
    {
      "arn": "arn:aws:iot:us-west-2:123456789012:all/registered-things"
    }
  ]
}
```

```
}
```

For more information, see [Detect Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListTargetsForSecurityProfile](#) in *AWS CLI Command Reference*.

list-thing-groups-for-thing

The following code example shows how to use `list-thing-groups-for-thing`.

AWS CLI

To list the groups that a thing belongs to

The following `list-thing-groups-for-thing` example lists the groups to which the specified thing belongs.

```
aws iot list-thing-groups-for-thing \  
  --thing-name MyLightBulb
```

Output:

```
{  
  "thingGroups": [  
    {  
      "groupName": "DeadBulbs",  
      "groupArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/DeadBulbs"  
    },  
    {  
      "groupName": "LightBulbs",  
      "groupArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/LightBulbs"  
    }  
  ]  
}
```

For more information, see [Thing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListThingGroupsForThing](#) in *AWS CLI Command Reference*.

list-thing-groups

The following code example shows how to use `list-thing-groups`.

AWS CLI

To list the thing groups defined in your AWS account

The following `describe-thing-group` example lists all thing groups defined in your AWS account.

```
aws iot list-thing-groups
```

Output:

```
{
  "thingGroups": [
    {
      "groupName": "HalogenBulbs",
      "groupArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/HalogenBulbs"
    },
    {
      "groupName": "LightBulbs",
      "groupArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/LightBulbs"
    }
  ]
}
```

For more information, see [Thing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListThingGroups](#) in *AWS CLI Command Reference*.

list-thing-principals

The following code example shows how to use `list-thing-principals`.

AWS CLI

To list the principals associated with a thing

The following `list-thing-principals` example lists the principals (X.509 certificates, IAM users, groups, roles, Amazon Cognito identities, or federated identities) associated with the specified thing.

```
aws iot list-thing-principals \
  --thing-name MyRaspberryPi
```

Output:

```
{
  "principals": [
    "arn:aws:iot:us-west-2:123456789012:cert/33475ac865079a5ffd5ecd44240640349293facc760642d7d8d5dbb6b4c86893"
  ]
}
```

For more information, see [ListThingPrincipals](#) in the *AWS IoT API Reference*.

- For API details, see [ListThingPrincipals](#) in *AWS CLI Command Reference*.

list-thing-types

The following code example shows how to use `list-thing-types`.

AWS CLI**To list the defined thing types**

The following `list-thing-types` example displays a list of thing types defined in your AWS account.

```
aws iot list-thing-types
```

Output:

```
{
  "thingTypes": [
    {
      "thingTypeName": "LightBulb",
      "thingTypeArn": "arn:aws:iot:us-west-2:123456789012:thingtype/LightBulb",
      "thingTypeProperties": {
        "thingTypeDescription": "light bulb type",
        "searchableAttributes": [
          "model",
          "wattage"
        ]
      },
      "thingTypeMetadata": {
```

```
        "deprecated": false,  
        "creationDate": 1559772562.498  
      }  
    ]  
  }  
}
```

For more information, see [Thing Types](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListThingTypes](#) in *AWS CLI Command Reference*.

list-things-in-billing-group

The following code example shows how to use `list-things-in-billing-group`.

AWS CLI

To list the things in a billing group

The following `list-things-in-billing-group` example lists the things that are in the specified billing group.

```
aws iot list-things-in-billing-group \  
  --billing-group-name GroupOne
```

Output:

```
{  
  "things": [  
    "MyOtherLightBulb",  
    "MyLightBulb"  
  ]  
}
```

For more information, see [Billing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListThingsInBillingGroup](#) in *AWS CLI Command Reference*.

list-things-in-thing-group

The following code example shows how to use `list-things-in-thing-group`.

AWS CLI

To list the things that belong to a group

The following `list-things-in-thing-group` example lists the things that belong to the specified thing group.

```
aws iot list-things-in-thing-group \  
  --thing-group-name LightBulbs
```

Output:

```
{  
  "things": [  
    "MyLightBulb"  
  ]  
}
```

For more information, see [Thing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListThingsInThingGroup](#) in *AWS CLI Command Reference*.

list-things

The following code example shows how to use `list-things`.

AWS CLI

Example 1: To list all things in the registry

The following `list-things` example lists the things (devices) that are defined in the AWS IoT registry for your AWS account.

```
aws iot list-things
```

Output:

```
{  
  "things": [  
    {  
      "thingName": "ThirdBulb",  
      "thingTypeName": "LightBulb",  
    }  
  ]  
}
```

```
    "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/ThirdBulb",
    "attributes": {
      "model": "123",
      "wattage": "75"
    },
    "version": 2
  },
  {
    "thingName": "MyOtherLightBulb",
    "thingTypeName": "LightBulb",
    "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MyOtherLightBulb",
    "attributes": {
      "model": "123",
      "wattage": "75"
    },
    "version": 3
  },
  {
    "thingName": "MyLightBulb",
    "thingTypeName": "LightBulb",
    "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MyLightBulb",
    "attributes": {
      "model": "123",
      "wattage": "75"
    },
    "version": 1
  },
  {
    "thingName": "SampleIoTThing",
    "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/SampleIoTThing",
    "attributes": {},
    "version": 1
  }
]
```

Example 2: To list the defined things that have a specific attribute

The following `list-things` example displays a list of things that have an attribute named `wattage`.

```
aws iot list-things \  
  --attribute-name wattage
```

Output:

```
{
  "things": [
    {
      "thingName": "MyLightBulb",
      "thingTypeName": "LightBulb",
      "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MyLightBulb",
      "attributes": {
        "model": "123",
        "wattage": "75"
      },
      "version": 1
    },
    {
      "thingName": "MyOtherLightBulb",
      "thingTypeName": "LightBulb",
      "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MyOtherLightBulb",
      "attributes": {
        "model": "123",
        "wattage": "75"
      },
      "version": 3
    }
  ]
}
```

For more information, see [How to Manage Things with the Registry](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListThings](#) in *AWS CLI Command Reference*.

list-topic-rule-destinations

The following code example shows how to use `list-topic-rule-destinations`.

AWS CLI**To list your topic rule destinations**

The following `list-topic-rule-destinations` example lists all topic rule destinations that you have defined in the current AWS Region.

```
aws iot list-topic-rule-destinations
```

Output:

```
{
  "destinationSummaries": [
    {
      "arn": "arn:aws:iot:us-west-2:123456789012:ruledestination/http/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
      "status": "ENABLED",
      "httpUrlSummary": {
        "confirmationUrl": "https://example.com"
      }
    }
  ]
}
```

For more information, see [Working with topic rule destinations](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListTopicRuleDestinations](#) in *AWS CLI Command Reference*.

list-topic-rules

The following code example shows how to use `list-topic-rules`.

AWS CLI

To list your rules

The following `list-topic-rules` example lists all rules that you have defined.

```
aws iot list-topic-rules
```

Output:

```
{
  "rules": [
    {
      "ruleArn": "arn:aws:iot:us-west-2:123456789012:rule/
MyRPiLowMoistureAlertRule",
      "ruleName": "MyRPiLowMoistureAlertRule",
      "topicPattern": "$aws/things/MyRPi/shadow/update/accepted",

```

```
        "createdAt": 1558624363.0,
        "ruleDisabled": false
    },
    {
        "ruleArn": "arn:aws:iot:us-west-2:123456789012:rule/
MyPlantPiMoistureAlertRule",
        "ruleName": "MyPlantPiMoistureAlertRule",
        "topicPattern": "$aws/things/MyPlantPi/shadow/update/accepted",
        "createdAt": 1541458459.0,
        "ruleDisabled": false
    }
]
}
```

For more information, see [Viewing Your Rules](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListTopicRules](#) in *AWS CLI Command Reference*.

list-v2-logging-levels

The following code example shows how to use `list-v2-logging-levels`.

AWS CLI

To list logging levels

The following `list-v2-logging-levels` example lists the configured logging levels. If logging levels were not set, a `NotConfiguredException` occurs when you run this command.

```
aws iot list-v2-logging-levels
```

Output:

```
{
  "logTargetConfigurations": [
    {
      "logTarget": {
        "targetType": "DEFAULT"
      },
      "logLevel": "ERROR"
    }
  ]
}
```



```
}
```

- For API details, see [ListV2LoggingLevels](#) in *AWS CLI Command Reference*.

list-violation-events

The following code example shows how to use `list-violation-events`.

AWS CLI

To list the security profile violations during a time period

The following `list-violation-events` example lists violations that occurred between June 5, 2019 and June 12, 2019 for all AWS IoT Device Defender security profiles for the current AWS account and AWS Region.

```
aws iot list-violation-events \  
  --start-time 1559747125 \  
  --end-time 1560351925
```

Output:

```
{  
  "violationEvents": [  
    {  
      "violationId": "174db59167fa474c80a652ad1583fd44",  
      "thingName": "iotconsole-1560269126751-1",  
      "securityProfileName": "Testprofile",  
      "behavior": {  
        "name": "Authorization",  
        "metric": "aws:num-authorization-failures",  
        "criteria": {  
          "comparisonOperator": "greater-than",  
          "value": {  
            "count": 10  
          },  
          "durationSeconds": 300,  
          "consecutiveDatapointsToAlarm": 1,  
          "consecutiveDatapointsToClear": 1  
        }  
      },  
      "metricValue": {
```

```
        "count": 0
    },
    "violationEventType": "in-alarm",
    "violationEventTime": 1560279000.0
},
{
    "violationId": "c8a9466a093d3b7b35cd44ca58bdbbeab",
    "thingName": "TvnQoEoU",
    "securityProfileName": "Testprofile",
    "behavior": {
        "name": "CellularBandwidth",
        "metric": "aws:message-byte-size",
        "criteria": {
            "comparisonOperator": "greater-than",
            "value": {
                "count": 128
            },
            "consecutiveDatapointsToAlarm": 1,
            "consecutiveDatapointsToClear": 1
        }
    },
    "metricValue": {
        "count": 110
    },
    "violationEventType": "in-alarm",
    "violationEventTime": 1560276600.0
},
{
    "violationId": "74aa393adea02e6648f3ac362beed55e",
    "thingName": "iotconsole-1560269232412-2",
    "securityProfileName": "Testprofile",
    "behavior": {
        "name": "Authorization",
        "metric": "aws:num-authorization-failures",
        "criteria": {
            "comparisonOperator": "greater-than",
            "value": {
                "count": 10
            },
            "durationSeconds": 300,
            "consecutiveDatapointsToAlarm": 1,
            "consecutiveDatapointsToClear": 1
        }
    }
},
```

```

        "metricValue": {
            "count": 0
        },
        "violationEventType": "in-alarm",
        "violationEventTime": 1560276600.0
    },
    {
        "violationId": "1e6ab5f7cf39a1466fcd154e1377e406",
        "thingName": "TvnQoEoU",
        "securityProfileName": "Testprofile",
        "behavior": {
            "name": "Authorization",
            "metric": "aws:num-authorization-failures",
            "criteria": {
                "comparisonOperator": "greater-than",
                "value": {
                    "count": 10
                },
                "durationSeconds": 300,
                "consecutiveDatapointsToAlarm": 1,
                "consecutiveDatapointsToClear": 1
            }
        },
        "metricValue": {
            "count": 0
        },
        "violationEventType": "in-alarm",
        "violationEventTime": 1560276600.0
    }
]
}

```

For more information, see [Detect Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [ListViolationEvents](#) in *AWS CLI Command Reference*.

register-ca-certificate

The following code example shows how to use register-ca-certificate.

AWS CLI

To register a certificate authority (CA) certificate

The following `register-ca-certificate` example registers a CA certificate. The command supplies the CA certificate and a key verification certificate that proves you own the private key associated with the CA certificate.

```
aws iot register-ca-certificate \  
  --ca-certificate file://rootCA.pem \  
  --verification-cert file://verificationCert.pem
```

Output:

```
{  
  "certificateArn": "arn:aws:iot:us-west-2:123456789012:cacert/  
f4efed62c0142f16af278166f61962501165c4f0536295207426460058cd1467",  
  "certificateId":  
  "f4efed62c0142f16af278166f61962501165c4f0536295207426460058cd1467"  
}
```

For more information, see [RegisterCACertificate](#) in the *AWS IoT API Reference*.

- For API details, see [RegisterCaCertificate](#) in *AWS CLI Command Reference*.

register-certificate

The following code example shows how to use `register-certificate`.

AWS CLI

To register a self signed device certificate

The following `register-certificate` example registers the `deviceCert.pem` device certificate signed by the `rootCA.pem` CA certificate. The CA certificate must be registered before you use it to register a self-signed device certificate. The self-signed certificate must be signed by the same CA certificate you pass to this command.

```
aws iot register-certificate \  
  --certificate-pem file://deviceCert.pem \  
  --ca-certificate-pem file://rootCA.pem
```

Output:

```
{
```

```

    "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142",
    "certificateId":
    "488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142"
}

```

For more information, see [RegisterCertificate](#) in the *AWS IoT API Reference*.

- For API details, see [RegisterCertificate](#) in *AWS CLI Command Reference*.

register-thing

The following code example shows how to use register-thing.

AWS CLI

To register a thing

The following register-thing example registers a thing using a provisioning template.

```

aws iot register-thing \
  --template-body '{"Parameters":{"ThingName":
{"Type":"String"},"AWS::IoT::Certificate::Id":{"Type":"String"},"Resources":
{"certificate":{"Properties":{"CertificateId":
{"Ref":"AWS::IoT::Certificate::Id"},"Status":"Active"},"Type":"AWS::IoT::Certificate"},"poli
{"Properties":{"PolicyName":"MyIotPolicy"},"Type":"AWS::IoT::Policy"},"thing":
{"OverrideSettings":
{"AttributePayload":"MERGE","ThingGroups":"DO_NOTHING","ThingTypeName":"REPLACE"},"Propertie
{"AttributePayload":{},"ThingGroups":[],"ThingName":
{"Ref":"ThingName"},"ThingTypeName":"VirtualThings"},"Type":"AWS::IoT::Thing"}}}' \
  --parameters '{"ThingName":"Register-thing-
trial-1","AWS::IoT::Certificate::Id":"799a9ea048a1e6aea42b55EXAMPLEf8697b4bafcd77a318a3068e3

```

Output:

```

{
  "certificatePem": "-----BEGIN CERTIFICATE-----
\nMIIDWTCCAkGgAwIBAgIUYLk81I35cIppobpw
Hi0J2jNjboIwDQYJKoZIhvcNAQEL
\nBQAwTTFLEkGGA1UECwxCQW1hem9uIFd1YiBTZXJ2aWNlcyBPPUftYXpvbi
5jb20g\nSW5jLiBMPVNlYXR0bGUgU1Q9V2FzaGluZ3RvbiBDPVVTMB4XDTIwMDcyMzE2NDUw
\n0VoXDTQ5MTIzMT

```

```

IzNTk10VowHjEcMBoGA1UEAwTQVdTIElVVCBDZXJ0aWZpY2F0\nZTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC
AQoCggEBA071uADhdBajqTmgrMV5\nmCFfBZQRMo1MdtVoZr2X+M4MzL
+RARrtUzH9a2SMAckeX8Keb1IOTKzORI
RDXnye
\n61V0wjgAsd0ku22rFex4eG2ikha7pYYkvuToqA7L3TxItRvfKrxRI4ZfJoFPip4\nKqiuBJVNOGKTcQ
Hd1RN0rddwwu6kFJLeKDmEXAMPLEdUF0N+qfR9yKnZQkm
+g6Q2\nGXu7u0W3hn6n1RN8qVoka0uW12p53xM7oHVz
Gf+cxKBx1b0hGkp6yCfTskUBm3Sp\n9zLw35kiHXVm4EVpwgNlnk6XcIGIkW8a/iy4pzmVUgAANY1/uU/
zgCjymw
ZT5S30\nBV0CAwEAAANgMF4wHwYDVR0jBBgwFoAUGx0tCcU3q2n1WXAuUCv6hugXjKswHQYD
\nVR00BBYEF0VtvZ
9Aj2RYFnkX7Iu01XTRUdxgMAwGA1UdEwEB/wQCMAAwDgYDVR0P\nAQH/
BAQDAgeAMA0GCSqGSIb3DQEBwUAA4IB
AQCXCQcp0tubS5ft0sDMTcP/jNX
\nDHyaRxmjpSc2aCdm7WX591TKWyAdxGAvqaDVWqTo0oXI7tZ8w7aIN1Gi5
pXnifx\n3SBebMUoBbTktrC97yUaeL025mCFv8emDnTR/fe7PTsBKjW0g/rfPwBxZLXDFwN
\nnqkQjy3EDfifj2
6j0xYIqqWMPogyn4sr0CKynS5wMJUQZ1HQ0nabVwnwK4Y0Mflp
\np9+4susFUR9aT3BT1AcIwqSpzh1Khh4Iz7ND
kRn4amsUT210jg/z0010w+BTHcVQ\nJly8XDuoCWSu04q6SnaBzHmlySIajxuRTP/AdfRouP10Xe
+q1bPOBcvVvF
8o\n-----END CERTIFICATE-----\n",
  "resourceArns": {
    "certificate": "arn:aws:iot:us-
west-2:571032923833:cert/799a9ea048a1e6aea42b55EXAMPLEf8697b4bafcd77a318a3068e30404b9233c",
    "thing": "arn:aws:iot:us-west-2:571032923833:thing/Register-thing-trial-1"
  }
}

```

For more information, see [Provisioning by trusted user](#) in the *AWS IoT Core Developers Guide*.

- For API details, see [RegisterThing](#) in *AWS CLI Command Reference*.

reject-certificate-transfer

The following code example shows how to use `reject-certificate-transfer`.

AWS CLI

To reject a certificate transfer

The following `reject-certificate-transfer` example rejects the transfer of the specified device certificate from another AWS account.

```
aws iot reject-certificate-transfer \  
  --certificate-id  
  f0f33678c7c9a046e5cc87b2b1a58dfa0beec26db78add5e605d630e05c7fc8
```

This command produces no output.

For more information, see [Transfer a certificate to another account](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [RejectCertificateTransfer](#) in *AWS CLI Command Reference*.

remove-thing-from-billing-group

The following code example shows how to use `remove-thing-from-billing-group`.

AWS CLI

To remove a thing from a billing group

The following `remove-thing-from-billing-group` example removes the specified thing from a billing group.

```
aws iot remove-thing-from-billing-group \  
  --billing-group-name GroupOne \  
  --thing-name MyOtherLightBulb
```

This command produces no output.

For more information, see [Billing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [RemoveThingFromBillingGroup](#) in *AWS CLI Command Reference*.

remove-thing-from-thing-group

The following code example shows how to use `remove-thing-from-thing-group`.

AWS CLI

To remove a thing from a thing group

The following `remove-thing-from-thing-group` example removes the specified thing from a thing group.

```
aws iot remove-thing-from-thing-group \  
  --thing-name bulb7 \  
  --thing-group-name DeadBulbs
```

This command produces no output.

For more information, see Thing Groups <<https://docs.aws.amazon.com/iot/latest/developerguide/thing-groups.html>> in the *AWS IoT Developer Guide*.

- For API details, see [RemoveThingFromThingGroup](#) in *AWS CLI Command Reference*.

replace-topic-rule

The following code example shows how to use `replace-topic-rule`.

AWS CLI

To update a topic's rule definition

The following `replace-topic-rule` example updates the specified rule to send an SNS alert when soil moisture level readings are too low.

```
aws iot replace-topic-rule \  
  --rule-name MyRPiLowMoistureAlertRule \  
  --topic-rule-payload "{\"sql\": \"SELECT * FROM '$aws/things/MyRPi/shadow/  
update/accepted' WHERE state.reported.moisture = 'low'\", \"description\": \"Sends  
an alert when soil moisture level readings are too low.\", \"actions\": [{\"sns  
\": {\"targetArn\": \"arn:aws:sns:us-west-2:123456789012:MyRPiLowMoistureTopic\",  
\"roleArn\": \"arn:aws:iam::123456789012:role/service-role/MyRPiLowMoistureTopicRole  
\", \"messageFormat\": \"RAW\"}}], \"ruleDisabled\": false, \"awsIotSqlVersion\":  
\"2016-03-23\"}"
```

This command produces no output.

For more information, see [Creating an AWS IoT Rule](#) in the *AWS IoT Developer Guide*.

- For API details, see [ReplaceTopicRule](#) in *AWS CLI Command Reference*.

search-index

The following code example shows how to use `search-index`.

AWS CLI

To query the thing index

The following search-index example queries the AWS_Things index for things that have a type of LightBulb.

```
aws iot search-index \  
  --index-name "AWS_Things" \  
  --query-string "thingTypeName:LightBulb"
```

Output:

```
{  
  "things": [  
    {  
      "thingName": "MyLightBulb",  
      "thingId": "40da2e73-c6af-406e-b415-15acae538797",  
      "thingTypeName": "LightBulb",  
      "thingGroupNames": [  
        "LightBulbs",  
        "DeadBulbs"  
      ],  
      "attributes": {  
        "model": "123",  
        "wattage": "75"  
      },  
      "connectivity": {  
        "connected": false  
      }  
    },  
    {  
      "thingName": "ThirdBulb",  
      "thingId": "615c8455-33d5-40e8-95fd-3ee8b24490af",  
      "thingTypeName": "LightBulb",  
      "attributes": {  
        "model": "123",  
        "wattage": "75"  
      },  
      "connectivity": {  
        "connected": false  
      }  
    }  
  ],  
}
```

```
{
  "thingName": "MyOtherLightBulb",
  "thingId": "6dae0d3f-40c1-476a-80c4-1ed24ba6aa11",
  "thingTypeName": "LightBulb",
  "attributes": {
    "model": "123",
    "wattage": "75"
  },
  "connectivity": {
    "connected": false
  }
}
]
```

For more information, see [Managing Thing Indexing](#) in the *AWS IoT Developer Guide*.

- For API details, see [SearchIndex](#) in *AWS CLI Command Reference*.

set-default-authorizer

The following code example shows how to use `set-default-authorizer`.

AWS CLI

To set a default authorizer

The following `set-default-authorizer` example sets the custom authorizer named `CustomAuthorizer` as the default authorizer.

```
aws iot set-default-authorizer \
  --authorizer-name CustomAuthorizer
```

Output:

```
{
  "authorizerName": "CustomAuthorizer",
  "authorizerArn": "arn:aws:iot:us-west-2:123456789012:authorizer/
CustomAuthorizer"
}
```

For more information, see [CreateDefaultAuthorizer](#) in the *AWS IoT API Reference*.

- For API details, see [SetDefaultAuthorizer](#) in *AWS CLI Command Reference*.

set-default-policy-version

The following code example shows how to use `set-default-policy-version`.

AWS CLI

To set the default version for a policy

The following `set-default-policy-version` example sets the default version to 2 for the policy named `UpdateDeviceCertPolicy`.

```
aws iot set-default-policy-version \  
  --policy-name UpdateDeviceCertPolicy \  
  --policy-version-id 2
```

This command produces no output.

- For API details, see [SetDefaultPolicyVersion](#) in *AWS CLI Command Reference*.

set-v2-logging-level

The following code example shows how to use `set-v2-logging-level`.

AWS CLI

To set the logging level for a thing group

The following `set-v2-logging-level` example sets the logging level to log warnings for the specified thing group.

```
aws iot set-v2-logging-level \  
  --log-target "{\"targetType\":\"THING_GROUP\",\"targetName\":\"LightBulbs\"}" \  
  --log-level WARN
```

This command produces no output.

- For API details, see [SetV2LoggingLevel](#) in *AWS CLI Command Reference*.

set-v2-logging-options

The following code example shows how to use `set-v2-logging-options`.

AWS CLI

To set the logging options

The following `set-v2-logging-options` example sets the default logging verbosity level to `ERROR` and specifies the ARN to use for logging.

```
aws iot set-v2-logging-options \  
  --default-log-level ERROR \  
  --role-arn "arn:aws:iam::094249569039:role/service-role/iotLoggingRole"
```

This command produces no output.

- For API details, see [SetV2LoggingOptions](#) in *AWS CLI Command Reference*.

start-audit-mitigation-actions-task

The following code example shows how to use `start-audit-mitigation-actions-task`.

AWS CLI

To apply a mitigation action to the findings from an audit

The following `start-audit-mitigation-actions-task` example applies the `ResetPolicyVersionAction` action (which clears the policy) to the specified single finding.

```
aws iot start-audit-mitigation-actions-task \  
  --task-id "myActionsTaskId" \  
  --target "findingIds=[\"0edbaaec-2fe1-4cf5-abc9-d4c3e51f7464\"]" \  
  --audit-check-to-actions-mapping  
  "IOT_POLICY_OVERLY_PERMISSIVE_CHECK=[\"ResetPolicyVersionAction\"]" \  
  --client-request-token "adhadhahda"
```

Output:

```
{  
  "taskId": "myActionsTaskId"
```

```
}
```

For more information, see [StartAuditMitigationActionsTask \(Mitigation Action Commands\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [StartAuditMitigationActionsTask](#) in *AWS CLI Command Reference*.

start-on-demand-audit-task

The following code example shows how to use `start-on-demand-audit-task`.

AWS CLI

To start an audit right away

The following `start-on-demand-audit-task` example starts an AWS IoT Device Defender audit and performs three certificate checks.

```
aws iot start-on-demand-audit-task \  
  --target-check-names CA_CERTIFICATE_EXPIRING_CHECK \  
  DEVICE_CERTIFICATE_EXPIRING_CHECK REVOKED_CA_CERTIFICATE_STILL_ACTIVE_CHECK
```

Output:

```
{  
  "taskId": "a3aea009955e501a31b764abe1bebd3d"  
}
```

For more information, see [Audit Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [StartOnDemandAuditTask](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To specify a tag key and value for a resource

The following `tag-resource` example applies the tag with a key `Assembly` and the value `Fact1NW` to the thing group `LightBulbs`.

```
aws iot tag-resource \  
  --tags Key=Assembly,Value="Fact1NW" \  
  --resource-arn "arn:aws:iot:us-west-2:094249569039:thinggroup/LightBulbs"
```

This command produces no output.

For more information, see [Tagging Your AWS IoT Resources](#) in the *AWS IoT Developer Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

test-authorization

The following code example shows how to use test-authorization.

AWS CLI

To test your AWS IoT policies

The following test-authorization example tests the AWS IoT policies associated with the specified principal.

```
aws iot test-authorization \  
  --auth-infos actionType=CONNECT,resources=arn:aws:iot:us-  
east-1:123456789012:client/client1 \  
  --principal arn:aws:iot:us-west-2:123456789012:cert/  
aab1068f7f43ac3e3cae4b3a8aa3f308d2a750e6350507962e32c1eb465d9775
```

Output:

```
{  
  "authResults": [  
    {  
      "authInfo": {  
        "actionType": "CONNECT",  
        "resources": [  
          "arn:aws:iot:us-east-1:123456789012:client/client1"  
        ]  
      },  
      "allowed": {  
        "policies": [  
          {  
            "policyName": "TestPolicyAllowed",
```

```

        "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/
TestPolicyAllowed"
    }
  ]
},
"denied": {
  "implicitDeny": {
    "policies": [
      {
        "policyName": "TestPolicyDenied",
        "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/
TestPolicyDenied"
      }
    ]
  },
  "explicitDeny": {
    "policies": [
      {
        "policyName": "TestPolicyExplicitDenied",
        "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/
TestPolicyExplicitDenied"
      }
    ]
  }
},
"authDecision": "IMPLICIT_DENY",
"missingContextValues": []
}
]
}

```

For more information, see [TestAuthorization](#) in the *AWS IoT API Reference*.

- For API details, see [TestAuthorization](#) in *AWS CLI Command Reference*.

test-invoke-authorizer

The following code example shows how to use `test-invoke-authorizer`.

AWS CLI

To test your custom authorizer

The following `test-invoke-authorizer` example tests your custom authorizer.

```
aws iot test-invoke-authorizer \
  --authorizer-name IoTAuthorizer \
  --token allow \
  --token-signature "mE0GvaHqy9nER/
FdgtJX5lXYEJ3b3vE7t1gEszc0TKGgLKWXTnPk2AbKn0AZ8lGyoN5dVtWDWVmr25m7+
+zjbYIMk2TBvyGXh0mvKFBPkdgyA43KL6SiZy0cTq1PMcQDsP7VX2rXr7CTowCxSNKphGXdQe0/
I5dQ+J06KUaHwCmupt0/MejKtaNwiia064j6wpr0AUwG5S1IYFuRd0X
+wfo8pb0DubAIX1Ua705kuhRUcTx4SxUSHEYKmN4IDEvLB6FsIr0B2wvB7y4iPmcajxzG102ExvyCUNctCV9dY1RRGJj
```

Output:

```
{
  "isAuthenticated": true,
  "principalId": "principalId",
  "policyDocuments": [
    {"Version": "2012-10-17", "Statement":
  [{"Action": "iot:Publish", "Effect": "Allow", "Resource": "arn:aws:iot:us-
west-2:123456789012:topic/customauthtesting"}]}]
  },
  "refreshAfterInSeconds": 600,
  "disconnectAfterInSeconds": 3600
}
```

For more information, see [TestInvokeAuthorizer](#) in the *AWS IoT API Reference*.

- For API details, see [TestInvokeAuthorizer](#) in *AWS CLI Command Reference*.

transfer-certificate

The following code example shows how to use `transfer-certificate`.

AWS CLI

To transfer a device certificate to a different AWS account

The following `transfer-certificate` example transfers a device certificate to another AWS account. The certificate and AWS account are identified by ID.

```
aws iot transfer-certificate \
  --certificate-id
488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142 \
  --target-aws-account 030714055129
```


Output:

```
{
  "transferredCertificateArn": "arn:aws:iot:us-
west-2:030714055129:cert/488b6a7f2acdeb00a77384e63c4e40b18b1b3caaae57b7272ba44c45e3448142"
}
```

For more information, see [Transfer a certificate to another account](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [TransferCertificate](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI**To remove a tag key from a resource**

The following `untag-resource` example removes the tag `MyTag` and its value from the thing group `LightBulbs`.

```
command
```

This command produces no output.

For more information, see [Tagging Your AWS IoT Resources](#) in the *AWS IoT Developer Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-account-audit-configuration

The following code example shows how to use `update-account-audit-configuration`.

AWS CLI**Example 1: To enable Amazon SNS notifications for audit notifications**

The following `update-account-audit-configuration` example enables Amazon SNS notifications for AWS IoT Device Defender audit notifications, specifying a target and the role used to write to that target.

```
aws iot update-account-audit-configuration \  
  --audit-notification-target-configurations "SNS={targetArn=\"arn:aws:sns:us-  
west-2:123456789012:ddauids\"},roleArn=\"arn:aws:iam::123456789012:role/service-  
role/AWSIoTDeviceDefenderAudit\"},enabled=true}"
```

This command produces no output.

Example 2: To enable an audit check

The following `update-account-audit-configuration` example enables the AWS IoT Device Defender audit check named `AUTHENTICATED_COGNITO_ROLE_OVERLY_PERMISSIVE_CHECK`. You cannot disable an audit check if it is part of the `targetCheckNames` for one or more scheduled audits for the AWS account.

```
aws iot update-account-audit-configuration \  
  --audit-check-configurations  
  "{\"AUTHENTICATED_COGNITO_ROLE_OVERLY_PERMISSIVE_CHECK\":{\"enabled\":true}}"
```

This command produces no output.

For more information, see [Audit Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [UpdateAccountAuditConfiguration](#) in *AWS CLI Command Reference*.

update-audit-suppression

The following code example shows how to use `update-audit-suppression`.

AWS CLI

To update an audit finding suppression

The following `update-audit-suppression` example updates an audit finding suppression's expiration date to `2020-09-21`.

```
aws iot update-audit-suppression \  
  --check-name DEVICE_CERTIFICATE_EXPIRING_CHECK \  
  --resource-identifier deviceCertificateId=c7691e<shortened> \  
  --no-suppress-indefinitely \  
  --expiration-date 2020-09-21
```

This command produces no output.

For more information, see [Audit finding suppressions](#) in the *AWS IoT Developers Guide*.

- For API details, see [UpdateAuditSuppression](#) in *AWS CLI Command Reference*.

update-authorizer

The following code example shows how to use `update-authorizer`.

AWS CLI

To update a custom authorizer

The following `update-authorizer` example sets the state of `CustomAuthorizer2` to `INACTIVE`.

```
aws iot update-authorizer \  
  --authorizer-name CustomAuthorizer2 \  
  --status INACTIVE
```

Output:

```
{  
  "authorizerName": "CustomAuthorizer2",  
  "authorizerArn": "arn:aws:iot:us-west-2:123456789012:authorizer/  
CustomAuthorizer2"  
}
```

For more information, see [UpdateAuthorizer](#) in the *AWS IoT API Reference*.

- For API details, see [UpdateAuthorizer](#) in *AWS CLI Command Reference*.

update-billing-group

The following code example shows how to use `update-billing-group`.

AWS CLI

To update information about a billing group

The following `update-billing-group` example updates the description for the specified billing group.

```
aws iot update-billing-group \  
  --billing-group-name GroupOne \  
  --billing-group-properties "billingGroupDescription=\"Primary bulb billing group  
\""
```

Output:

```
{  
  "version": 2  
}
```

For more information, see [Billing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [UpdateBillingGroup](#) in *AWS CLI Command Reference*.

update-ca-certificate

The following code example shows how to use `update-ca-certificate`.

AWS CLI

To update a certificate authority (CA) certificate

The following `update-ca-certificate` example sets the specified CA certificate to ACTIVE status.

```
aws iot update-ca-certificate \  
  --certificate-id  
  f4efed62c0142f16af278166f61962501165c4f0536295207426460058cd1467 \  
  --new-status ACTIVE
```

This command produces no output.

For more information, see [UpdateCACertificate](#) in the *AWS IoT API Reference*.

- For API details, see [UpdateCaCertificate](#) in *AWS CLI Command Reference*.

update-certificate

The following code example shows how to use `update-certificate`.

AWS CLI

To update a device certificate

The following `update-certificate` example sets the specified device certificate to INACTIVE status.

```
aws iot update-certificate \  
  --certificate-id  
  d1eb269fb55a628552143c8f96eb3c258fcd5331ea113e766ba0c82bf225f0be \  
  --new-status INACTIVE
```

This command produces no output.

For more information, see [UpdateCertificate](#) in the *AWS IoT API Reference*.

- For API details, see [UpdateCertificate](#) in *AWS CLI Command Reference*.

update-custom-metric

The following code example shows how to use `update-custom-metric`.

AWS CLI

To update a custom metric

The following `update-custom-metric` example updates a custom metric to have a new `display-name`.

```
aws iot update-custom-metric \  
  --metric-name batteryPercentage \  
  --display-name 'remaining battery percentage on device' \  
  --region us-east-1
```

Output:

```
{  
  "metricName": "batteryPercentage",  
  "metricArn": "arn:aws:iot:us-east-1:1234564789012:custommetric/  
batteryPercentage",  
  "metricType": "number",
```

```
"displayName": "remaining battery percentage on device",
"creationDate": "2020-11-17T23:01:35.110000-08:00",
"lastModifiedDate": "2020-11-17T23:02:12.879000-08:00"
}
```

For more information, see [Custom metrics](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [UpdateCustomMetric](#) in *AWS CLI Command Reference*.

update-dimension

The following code example shows how to use update-dimension.

AWS CLI

To update a dimension

The following update-dimension example updates a dimension.

```
aws iot update-dimension \
  --name TopicFilterForAuthMessages \
  --string-values device/${iot:ClientId}/auth
```

Output:

```
{
  "name": "TopicFilterForAuthMessages",
  "lastModifiedDate": 1585866222.317,
  "stringValues": [
    "device/${iot:ClientId}/auth"
  ],
  "creationDate": 1585854500.474,
  "type": "TOPIC_FILTER",
  "arn": "arn:aws:iot:us-west-2:1234564789012:dimension/TopicFilterForAuthMessages"
}
```

For more information, see [Scoping metrics in security profiles using dimensions](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [UpdateDimension](#) in *AWS CLI Command Reference*.

update-domain-configuration

The following code example shows how to use `update-domain-configuration`.

AWS CLI

To update a domain configuration

The following `update-domain-configuration` example disables the specified domain configuration.

```
aws iot update-domain-configuration \  
  --domain-configuration-name "additionalDataDomain" \  
  --domain-configuration-status "DISABLED"
```

Output:

```
{  
  "domainConfigurationName": "additionalDataDomain",  
  "domainConfigurationArn": "arn:aws:iot:us-  
west-2:123456789012:domainconfiguration/additionalDataDomain/dikMh"  
}
```

For more information, see [Configurable Endpoints](#) in the *AWS IoT Developer Guide*.

- For API details, see [UpdateDomainConfiguration](#) in *AWS CLI Command Reference*.

update-dynamic-thing-group

The following code example shows how to use `update-dynamic-thing-group`.

AWS CLI

To update a dynamic thing group

The following `update-dynamic-thing-group` example updates the specified dynamic thing group. It provides a description and updates the query string to change the group membership criteria.

```
aws iot update-dynamic-thing-group \  
  --thing-group-name "RoomTooWarm"
```

```
--thing-group-properties "thingGroupDescription=\"This thing group contains  
rooms warmer than 65F.\" \" \" \  
--query-string "attributes.temperature>65"
```

Output:

```
{  
  "version": 2  
}
```

For more information, see [Dynamic Thing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [UpdateDynamicThingGroup](#) in *AWS CLI Command Reference*.

update-event-configurations

The following code example shows how to use `update-event-configurations`.

AWS CLI

To show which event types are published

The following `update-event-configurations` example updates the configuration to enable messages when the CA certificate is added, updated, or deleted.

```
aws iot update-event-configurations \  
  --event-configurations "{\"CA_CERTIFICATE\":{\"Enabled\":true}}"
```

This command produces no output.

For more information, see [Event Messages](#) in the *AWS IoT Developer Guide*.

- For API details, see [UpdateEventConfigurations](#) in *AWS CLI Command Reference*.

update-indexing-configuration

The following code example shows how to use `update-indexing-configuration`.

AWS CLI

To enable thing indexing

The following `update-indexing-configuration` example enables thing indexing to support searching registry data, shadow data, and thing connectivity status using the `AWS_Things` index.

```
aws iot update-indexing-configuration
  --thing-indexing-configuration
  thingIndexingMode=REGISTRY_AND_SHADOW,thingConnectivityIndexingMode=STATUS
```

This command produces no output.

For more information, see [Managing Thing Indexing](#) in the *AWS IoT Developers Guide*.

- For API details, see [UpdateIndexingConfiguration](#) in *AWS CLI Command Reference*.

update-job

The following code example shows how to use `update-job`.

AWS CLI

To get detailed status for a job

The following `update-job` example gets detailed status for the job whose ID is `example-job-01`.

```
aws iot describe-job \
  --job-id "example-job-01"
```

Output:

```
{
  "job": {
    "jobArn": "arn:aws:iot:us-west-2:123456789012:job/example-job-01",
    "jobId": "example-job-01",
    "targetSelection": "SNAPSHOT",
    "status": "IN_PROGRESS",
    "targets": [
      "arn:aws:iot:us-west-2:123456789012:thing/MyRaspberryPi"
    ],
    "description": "example job test",
    "presignedUrlConfig": {},
  }
}
```

```

    "jobExecutionsRolloutConfig": {},
    "createdAt": 1560787022.733,
    "lastUpdatedAt": 1560787026.294,
    "jobProcessDetails": {
      "numberOfCanceledThings": 0,
      "numberOfSucceededThings": 0,
      "numberOfFailedThings": 0,
      "numberOfRejectedThings": 0,
      "numberOfQueuedThings": 1,
      "numberOfInProgressThings": 0,
      "numberOfRemovedThings": 0,
      "numberOfTimedOutThings": 0
    },
    "timeoutConfig": {}
  }
}

```

For more information, see [Creating and Managing Jobs \(CLI\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [UpdateJob](#) in *AWS CLI Command Reference*.

update-mitigation-action

The following code example shows how to use update-mitigation-action.

AWS CLI

To update a mitigation action

The following update-mitigation-action example updates the specified mitigation action named AddThingsToQuarantineGroupAction, changes the thing group name, and sets overrideDynamicGroups to false. You can verify your changes by using the describe-mitigation-action command.

```

aws iot update-mitigation-action \
  --cli-input-json "{ \"actionName\": \"AddThingsToQuarantineGroupAction\",
  \"actionParams\": { \"addThingsToThingGroupParams\": {\"thingGroupNames\":
  [\"QuarantineGroup2\"],\"overrideDynamicGroups\": false}}}"

```

Output:

```
{
```

```
"actionArn": "arn:aws:iot:us-west-2:123456789012:mitigationaction/
AddThingsToQuarantineGroupAction",
  "actionId": "2fd2726d-98e1-4abf-b10f-09465ccd6bfa"
}
```

For more information, see [UpdateMitigationAction \(Mitigation Action Commands\)](#) in the *AWS IoT Developer Guide*.

- For API details, see [UpdateMitigationAction](#) in *AWS CLI Command Reference*.

update-provisioning-template

The following code example shows how to use `update-provisioning-template`.

AWS CLI

To update a provisioning template

The following `update-provisioning-template` example modifies the description and role arn for the specified provisioning template and enables the template.

```
aws iot update-provisioning-template \
  --template-name widget-template \
  --enabled \
  --description "An updated provisioning template for widgets" \
  --provisioning-role-arn arn:aws:iam::504350838278:role/Provision_role
```

This command produces no output.

For more information, see [AWS IoT Secure Tunneling](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [UpdateProvisioningTemplate](#) in *AWS CLI Command Reference*.

update-role-alias

The following code example shows how to use `update-role-alias`.

AWS CLI

To update a role alias

The following `update-role-alias` example updates the `LightBulbRole` role alias.

```
aws iot update-role-alias \  
  --role-alias LightBulbRole \  
  --role-arn arn:aws:iam::123456789012:role/lightbulbrole-001
```

Output:

```
{  
  "roleAlias": "LightBulbRole",  
  "roleAliasArn": "arn:aws:iot:us-west-2:123456789012:rolealias/LightBulbRole"  
}
```

For more information, see [UpdateRoleAlias](#) in the *AWS IoT API Reference*.

- For API details, see [UpdateRoleAlias](#) in *AWS CLI Command Reference*.

update-scheduled-audit

The following code example shows how to use `update-scheduled-audit`.

AWS CLI

To update a scheduled audit definition

The following `update-scheduled-audit` example changes the target check names for an AWS IoT Device Defender scheduled audit.

```
aws iot update-scheduled-audit \  
  --scheduled-audit-name WednesdayCertCheck \  
  --target-check-names CA_CERTIFICATE_EXPIRING_CHECK  
  DEVICE_CERTIFICATE_EXPIRING_CHECK REVOKED_CA_CERTIFICATE_STILL_ACTIVE_CHECK
```

Output:

```
{  
  "scheduledAuditArn": "arn:aws:iot:us-west-2:123456789012:scheduledaudit/  
  WednesdayCertCheck"  
}
```

For more information, see [Audit Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [UpdateScheduledAudit](#) in *AWS CLI Command Reference*.

update-security-profile

The following code example shows how to use update-security-profile.

AWS CLI

To change a security profile

The following update-security-profile example updates both the description and the behaviors for an AWS IoT Device Defender security profile.

```
aws iot update-security-profile \
  --security-profile-name PossibleIssue \
  --security-profile-description "Check to see if authorization fails 12 times in
  5 minutes or if cellular bandwidth exceeds 128" \
  --behaviors "[{\\"name\\":\\"CellularBandwidth\\",\\"metric\\":\\"aws:message-byte-size
  \",\\"criteria\\":{\\"comparisonOperator\\":\\"greater-than\\",\\"value\\":{\\"count\\":128},
  \\"consecutiveDatapointsToAlarm\\":1,\\"consecutiveDatapointsToClear\\":1}},{\\"name
  \":\\"Authorization\\",\\"metric\\":\\"aws:num-authorization-failures\\",\\"criteria\\":
  {\\"comparisonOperator\\":\\"less-than\\",\\"value\\":{\\"count\\":12},\\"durationSeconds
  \":300,\\"consecutiveDatapointsToAlarm\\":1,\\"consecutiveDatapointsToClear\\":1}}]"
```

Output:

```
{
  "securityProfileName": "PossibleIssue",
  "securityProfileArn": "arn:aws:iot:us-west-2:123456789012:securityprofile/
  PossibleIssue",
  "securityProfileDescription": "check to see if authorization fails 12 times in 5
  minutes or if cellular bandwidth exceeds 128",
  "behaviors": [
    {
      "name": "CellularBandwidth",
      "metric": "aws:message-byte-size",
      "criteria": {
        "comparisonOperator": "greater-than",
        "value": {
          "count": 128
        },
        "consecutiveDatapointsToAlarm": 1,
        "consecutiveDatapointsToClear": 1
      }
    }
  ],
}
```

```
{
  "name": "Authorization",
  "metric": "aws:num-authorization-failures",
  "criteria": {
    "comparisonOperator": "less-than",
    "value": {
      "count": 12
    },
    "durationSeconds": 300,
    "consecutiveDatapointsToAlarm": 1,
    "consecutiveDatapointsToClear": 1
  }
},
"version": 2,
"creationDate": 1560278102.528,
"lastModifiedDate": 1560352711.207
}
```

For more information, see [Detect Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [UpdateSecurityProfile](#) in *AWS CLI Command Reference*.

update-stream

The following code example shows how to use `update-stream`.

AWS CLI

To update a stream

The following `update-stream` example updates an existing stream. The stream version is incremented by one.

```
aws iot update-stream \
  --cli-input-json file://update-stream.json
```

Contents of `update-stream.json`:

```
{
  "streamId": "stream12345",
  "description": "This stream is used for Amazon FreeRTOS OTA Update 12345.",
  "files": [
```

```

    {
      "fileId": 123,
      "s3Location": {
        "bucket": "codesign-ota-bucket",
        "key": "48c67f3c-63bb-4f92-a98a-4ee0fbc2bef6"
      }
    }
  ]
  "roleArn": "arn:aws:iam:us-west-2:123456789012:role/service-role/my_ota_stream_role"
}

```

Output:

```

{
  "streamId": "stream12345",
  "streamArn": "arn:aws:iot:us-west-2:123456789012:stream/stream12345",
  "description": "This stream is used for Amazon FreeRTOS OTA Update 12345.",
  "streamVersion": 2
}

```

For more information, see [UpdateStream](#) in the *AWS IoT API Reference*.

- For API details, see [UpdateStream](#) in *AWS CLI Command Reference*.

update-thing-group

The following code example shows how to use `update-thing-group`.

AWS CLI**To update the definition for a thing group**

The following `update-thing-group` example updates the definition for the specified thing group, changing the description and two attributes.

```

aws iot update-thing-group \
  --thing-group-name HalogenBulbs \
  --thing-group-properties "thingGroupDescription=\"Halogen bulb group\"",
  attributePayload={attributes={Manufacturer=AnyCompany,wattage=60}}"

```

Output:

```
{
  "version": 2
}
```

For more information, see [Thing Groups](#) in the *AWS IoT Developers Guide*.

- For API details, see [UpdateThingGroup](#) in *AWS CLI Command Reference*.

update-thing-groups-for-thing

The following code example shows how to use `update-thing-groups-for-thing`.

AWS CLI

To change the groups to which a thing belongs

The following `update-thing-groups-for-thing` example removes the thing named `MyLightBulb` from the group named `DeadBulbs` and adds it to the group named `replaceableItems` at the same time.

```
aws iot update-thing-groups-for-thing \
  --thing-name MyLightBulb \
  --thing-groups-to-add "replaceableItems" \
  --thing-groups-to-remove "DeadBulbs"
```

This command produces no output.

For more information, see [Thing Groups](#) in the *AWS IoT Developer Guide*.

- For API details, see [UpdateThingGroupsForThing](#) in *AWS CLI Command Reference*.

update-thing

The following code example shows how to use `update-thing`.

AWS CLI

To associate a thing with a thing type

The following `update-thing` example associates a thing in the AWS IoT registry with a thing type. When you make the association, you provide values for the attributes defined by the thing type.


```
aws iot update-thing \  
  --thing-name "MyOtherLightBulb" \  
  --thing-type-name "LightBulb" \  
  --attribute-payload '{"attributes": {"wattage": "75", "model": "123"}}'
```

This command does not produce output. Use the `describe-thing` command to see the result.

For more information, see [Thing Types](#) in the *AWS IoT Developers Guide*.

- For API details, see [UpdateThing](#) in *AWS CLI Command Reference*.

update-topic-rule-destination

The following code example shows how to use `update-topic-rule-destination`.

AWS CLI

Example 1: To enable a topic rule destination

The following `update-topic-rule-destination` example enables traffic to a topic rule destination.

```
aws iot update-topic-rule-destination \  
  --arn "arn:aws:iot:us-west-2:123456789012:ruledestination/http/  
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE" \  
  --status ENABLED
```

This command produces no output.

For more information, see [Enabling a topic rule destination](#) in the *AWS IoT Developer Guide*.

Example 2: To disable a topic rule destination

The following `update-topic-rule-destination` example disables traffic to a topic rule destination.

```
aws iot update-topic-rule-destination \  
  --arn "arn:aws:iot:us-west-2:123456789012:ruledestination/http/  
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE" \  
  --status DISABLED
```

This command produces no output.

For more information, see [Disabling a topic rule destination](#) in the *AWS IoT Developer Guide*.

Example 3: To send a new confirmation message

The following `update-topic-rule-destination` example sends a new confirmation message for a topic rule destination.

```
aws iot update-topic-rule-destination \  
  --arn "arn:aws:iot:us-west-2:123456789012:ruledestination/http/  
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE" \  
  --status IN_PROGRESS
```

This command produces no output.

For more information, see [Sending a new confirmation message](#) in the *AWS IoT Developer Guide*.

- For API details, see [UpdateTopicRuleDestination](#) in *AWS CLI Command Reference*.

validate-security-profile-behaviors

The following code example shows how to use `validate-security-profile-behaviors`.

AWS CLI

Example 1: To validate the behaviors parameters for a security profile

The following `validate-security-profile-behaviors` example validates a well-formed and correct set of behaviors for an AWS IoT Device Defender security profile.

```
aws iot validate-security-profile-behaviors \  
  --behaviors "[{"name":"CellularBandwidth","metric":"aws:message-byte-size",  
,"criteria":{"comparisonOperator":"greater-than","value":{"count":128},  
"consecutiveDatapointsToAlarm":1,"consecutiveDatapointsToClear":1}}, {"name  
":"Authorization","metric":"aws:num-authorization-failures","criteria":  
{"comparisonOperator":"greater-than","value":{"count":12},"durationSeconds  
":300,"consecutiveDatapointsToAlarm":1,"consecutiveDatapointsToClear":1}]]"
```

Output:

```
{  
  "valid": true,
```

```
"validationErrors": []
}
```

Example 2: To validate incorrect behaviors parameters for a security profile

The following `validate-security-profile-behaviors` example validates a set of behaviors that contains an error for an AWS IoT Device Defender security profile.

```
aws iot validate-security-profile-behaviors \
  --behaviors "[{\\"name\\":\\"CellularBandwidth\\",\\"metric\\":\\"aws:message-byte-size\\",\\"criteria\\":{\\"comparisonOperator\\":\\"greater-than\\",\\"value\\":{\\"count\\":128},\\"consecutiveDatapointsToAlarm\\":1,\\"consecutiveDatapointsToClear\\":1}},{\\"name\\":\\"Authorization\\",\\"metric\\":\\"aws:num-authorization-failures\\",\\"criteria\\":{\\"comparisonOperator\\":\\"greater-than\\",\\"value\\":{\\"count\\":12},\\"durationSeconds\\":300,\\"consecutiveDatapointsToAlarm\\":100000,\\"consecutiveDatapointsToClear\\":1}}]"
```

Output:

```
{
  "valid": false,
  "validationErrors": [
    {
      "errorMessage": "Behavior Authorization is malformed.
consecutiveDatapointsToAlarm 100000 should be in range[1,10]"
    }
  ]
}
```

For more information, see [Detect Commands](#) in the *AWS IoT Developer Guide*.

- For API details, see [ValidateSecurityProfileBehaviors](#) in *AWS CLI Command Reference*.

AWS IoT 1-Click Devices examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS IoT 1-Click Devices.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

claim-devices-by-claim-code

The following code example shows how to use `claim-devices-by-claim-code`.

AWS CLI

To claim one or more AWS IoT 1-Click devices using a claim code

The following `claim-devices-by-claim-code` example claims the specified AWS IoT 1-Click device using a claim code (instead of a device ID).

```
aws iot1click-devices claim-devices-by-claim-code \  
  --claim-code C-123EXAMPLE
```

Output:

```
{  
  "Total": 9  
  "ClaimCode": "C-123EXAMPLE"  
}
```

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [ClaimDevicesByClaimCode](#) in *AWS CLI Command Reference*.

describe-device

The following code example shows how to use `describe-device`.

AWS CLI

To describe a device

The following describe-device example describes the specified device.

```
aws iot1click-devices describe-device \  
  --device-id G030PM0123456789
```

Output:

```
{  
  "DeviceDescription": {  
    "Arn": "arn:aws:iot1click:us-west-2:012345678901:devices/G030PM0123456789",  
    "Attributes": {  
      "projectRegion": "us-west-2",  
      "projectName": "AnytownDumpsters",  
      "placementName": "customer217",  
      "deviceTemplateName": "empty-dumpster-request"  
    },  
    "DeviceId": "G030PM0123456789",  
    "Enabled": false,  
    "RemainingLife": 99.9,  
    "Type": "button",  
    "Tags": {}  
  }  
}
```

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [DescribeDevice](#) in *AWS CLI Command Reference*.

finalize-device-claim

The following code example shows how to use finalize-device-claim.

AWS CLI

To finalize a claim request for an AWS IoT 1-Click device using a device ID

The following finalize-device-claim example finalizes a claim request for the specified AWS IoT 1-Click device using a device ID (instead of a claim code).

```
aws iot1click-devices finalize-device-claim \  
  --device-id G030PM0123456789
```

Output:

```
{  
  "State": "CLAIMED"  
}
```

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [FinalizeDeviceClaim](#) in *AWS CLI Command Reference*.

get-device-methods

The following code example shows how to use `get-device-methods`.

AWS CLI

To list the available methods for a device

The following `get-device-methods` example lists the available methods for a device.

```
aws iot1click-devices get-device-methods \  
  --device-id G030PM0123456789
```

Output:

```
{  
  "DeviceMethods": [  
    {  
      "MethodName": "getDeviceHealthParameters"  
    },  
    {  
      "MethodName": "setDeviceHealthMonitorCallback"  
    },  
    {  
      "MethodName": "getDeviceHealthMonitorCallback"  
    },  
    {
```

```
        "MethodName": "setOnClickCallback"
      },
      {
        "MethodName": "getOnClickCallback"
      }
    ]
  }
}
```

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [GetDeviceMethods](#) in *AWS CLI Command Reference*.

initiate-device-claim

The following code example shows how to use `initiate-device-claim`.

AWS CLI

To initiate a claim request for an AWS IoT 1-Click device using a device ID

The following `initiate-device-claim` example initiates a claim request for the specified AWS IoT 1-Click device using a device ID (instead of a claim code).

```
aws iot1click-devices initiate-device-claim \
  --device-id G030PM0123456789
```

Output:

```
{
  "State": "CLAIM_INITIATED"
}
```

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [InitiateDeviceClaim](#) in *AWS CLI Command Reference*.

invoke-device-method

The following code example shows how to use `invoke-device-method`.

AWS CLI

To invoke a device method on a device

The following `invoke-device-method` example invokes the specified method on a device.

```
aws iot1click-devices invoke-device-method \  
  --cli-input-json file://invoke-device-method.json
```

Contents of `invoke-device-method.json`:

```
{  
  "DeviceId": "G030PM0123456789",  
  "DeviceMethod": {  
    "DeviceType": "device",  
    "MethodName": "getDeviceHealthParameters"  
  }  
}
```

Output:

```
{  
  "DeviceMethodResponse": "{\"remainingLife\": 99.8}"  
}
```

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [InvokeDeviceMethod](#) in *AWS CLI Command Reference*.

list-device-events

The following code example shows how to use `list-device-events`.

AWS CLI

To list a device's events for a specified time range

The following `list-device-events` example lists the specified device's events for the specified time range.


```
aws iot1click-devices list-device-events \  
  --device-id G030PM0123456789 \  
  --from-time-stamp 2019-07-17T15:45:12.880Z --to-time-stamp  
  2019-07-19T15:45:12.880Z
```

Output:

```
{  
  "Events": [  
    {  
      "Device": {  
        "Attributes": {},  
        "DeviceId": "G030PM0123456789",  
        "Type": "button"  
      },  
      "StdEvent": "{\\"clickType\\": \\"SINGLE\\",  
        \\"reportedTime\\": \\"2019-07-18T23:47:55.015Z\\", \\"certificateId\\":  
        \\"fe8798a6c97c62ef8756b80eeefdcf2280f3352f82faa8080c74cc4f4a4d1811\\",  
        \\"remainingLife\\": 99.850000000000001, \\"testMode\\": false}"  
    },  
    {  
      "Device": {  
        "Attributes": {},  
        "DeviceId": "G030PM0123456789",  
        "Type": "button"  
      },  
      "StdEvent": "{\\"clickType\\": \\"DOUBLE\\",  
        \\"reportedTime\\": \\"2019-07-19T00:14:41.353Z\\", \\"certificateId\\":  
        \\"fe8798a6c97c62ef8756b80eeefdcf2280f3352f82faa8080c74cc4f4a4d1811\\",  
        \\"remainingLife\\": 99.8, \\"testMode\\": false}"  
    }  
  ]  
}
```

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [ListDeviceEvents](#) in *AWS CLI Command Reference*.

list-devices

The following code example shows how to use `list-devices`.

AWS CLI

To list the devices of a specified type

The following `list-devices` example lists the devices of a specified type.

```
aws iot1click-devices list-devices \  
  --device-type button
```

This command produces no output.

Output:

```
{  
  "Devices": [  
    {  
      "remainingLife": 99.9,  
      "attributes": {  
        "arn": "arn:aws:iot1click:us-west-2:123456789012:devices/  
G030PM0123456789",  
        "type": "button",  
        "deviceId": "G030PM0123456789",  
        "enabled": false  
      }  
    }  
  ]  
}
```

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [ListDevices](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags for a device

The following `list-tags-for-resource` example list the tags for the specified device.

```
aws iot1click-devices list-tags-for-resource \  
  --resource-arn "arn:aws:iot1click:us-west-2:012345678901:devices/  
G030PM0123456789"
```

Output:

```
{  
  "Tags": {  
    "Driver Phone": "123-555-0199",  
    "Driver": "Jorge Souza"  
  }  
}
```

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To add tags to a device AWS resource

The following tag-resource example adds two tags to the specified resource.

```
aws iot1click-devices tag-resource \  
  --cli-input-json file://devices-tag-resource.json
```

Contents of devices-tag-resource.json:

```
{  
  "ResourceArn": "arn:aws:iot1click:us-west-2:123456789012:devices/  
G030PM0123456789",  
  "Tags": {  
    "Driver": "Jorge Souza",  
    "Driver Phone": "123-555-0199"  
  }  
}
```

```
}
```

This command produces no output.

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

unclaim-device

The following code example shows how to use unclaim-device.

AWS CLI

To unclaim (deregister) a device from your AWS account

The following unclaim-device example unclaims (deregisters) the specified device from your AWS account.

```
aws iot1click-devices unclaim-device \  
  --device-id G030PM0123456789
```

Output:

```
{  
  "State": "UNCLAIMED"  
}
```

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [UnclaimDevice](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use untag-resource.

AWS CLI

To remove tags from a device AWS resource

The following `untag-resource` example removes the tags with the names `Driver Phone` and `Driver` from the specified device resource.

```
aws iot1click-devices untag-resource \  
  --resource-arn "arn:aws:iot1click:us-west-2:123456789012:projects/  
AnytownDumpsters" \  
  --tag-keys "Driver Phone" "Driver"
```

This command produces no output.

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-device-state

The following code example shows how to use `update-device-state`.

AWS CLI

To update the `enabled` state for a device

The following `update-device-state` sets the state of the specified device to `enabled`.

```
aws iot1click-devices update-device-state \  
  --device-id G030PM0123456789 \  
  --enabled
```

This command produces no output.

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [UpdateDeviceState](#) in *AWS CLI Command Reference*.

AWS IoT 1-Click Projects examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS IoT 1-Click Projects.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-device-with-placement

The following code example shows how to use `associate-device-with-placement`.

AWS CLI

To associate an AWS IoT 1-Click device with an existing placement

The following `associate-device-with-placement` example associates the specified AWS IoT 1-Click device with an existing placement.

```
aws iot1click-projects associate-device-with-placement \  
  --project-name AnytownDumpsters \  
  --placement-name customer217 \  
  --device-template-name empty-dumpster-request \  
  --device-id G030PM0123456789
```

This command produces no output.

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [AssociateDeviceWithPlacement](#) in *AWS CLI Command Reference*.

create-placement

The following code example shows how to use `create-placement`.

AWS CLI

To create an AWS IoT 1-Click placement for a project

The following `create-placement` example create an AWS IoT 1-Click placement for the specified project.

```
aws iot1click-projects create-placement \  
  --project-name AnytownDumpsters \  
  --placement-name customer217 \  
  --attributes '{"location": "123 Any Street Anytown, USA 10001", "phone":  
  "123-456-7890"}'
```

This command produces no output.

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [CreatePlacement](#) in *AWS CLI Command Reference*.

create-project

The following code example shows how to use `create-project`.

AWS CLI

To create an AWS IoT 1-Click project for zero or more placements

The following `create-project` example creates an AWS IoT 1-Click project for a placement.

```
aws iot1click-projects create-project --cli-input-json file://create-project.json
```

Contents of `create-project.json`:

```
{  
  "projectName": "AnytownDumpsters",  
  "description": "All dumpsters in the Anytown region.",  
  "placementTemplate": {  
    "defaultAttributes": {  
      "City" : "Anytown"  
    },  
    "deviceTemplates": {  
      "empty-dumpster-request" : {
```

```
        "deviceType": "button"
      }
    }
  }
}
```

This command produces no output.

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [CreateProject](#) in *AWS CLI Command Reference*.

delete-placement

The following code example shows how to use delete-placement.

AWS CLI

To delete a placement from a project

The following delete-placement example deletes the specified placement from a project.

```
aws iot1click-projects delete-placement \
  --project-name AnytownDumpsters \
  --placement-name customer217
```

This command produces no output.

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [DeletePlacement](#) in *AWS CLI Command Reference*.

delete-project

The following code example shows how to use delete-project.

AWS CLI

To delete a project from your AWS account

The following delete-project example deletes the specified project from your AWS account.


```
aws iot1click-projects delete-project \  
  --project-name AnytownDumpsters
```

This command produces no output.

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [DeleteProject](#) in *AWS CLI Command Reference*.

describe-placement

The following code example shows how to use describe-placement.

AWS CLI

To describe a placement for a project

The following describe-placement example describes a placement for the specified project.

```
aws iot1click-projects describe-placement \  
  --project-name AnytownDumpsters \  
  --placement-name customer217
```

Output:

```
{  
  "placement": {  
    "projectName": "AnytownDumpsters",  
    "placementName": "customer217",  
    "attributes": {  
      "phone": "123-555-0110",  
      "location": "123 Any Street Anytown, USA 10001"  
    },  
    "createdDate": 1563488454,  
    "updatedAt": 1563488454  
  }  
}
```

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [DescribePlacement](#) in *AWS CLI Command Reference*.

describe-project

The following code example shows how to use `describe-project`.

AWS CLI

To describe an AWS IoT 1-Click project

The following `describe-project` example describes the specified AWS IoT 1-Click project.

```
aws iot1click-projects describe-project \  
  --project-name AnytownDumpsters
```

Output:

```
{  
  "project": {  
    "arn": "arn:aws:iot1click:us-west-2:012345678901:projects/AnytownDumpsters",  
    "projectName": "AnytownDumpsters",  
    "description": "All dumpsters in the Anytown region.",  
    "createdDate": 1563483100,  
    "updatedAt": 1563483100,  
    "placementTemplate": {  
      "defaultAttributes": {  
        "City": "Anytown"  
      },  
      "deviceTemplates": {  
        "empty-dumpster-request": {  
          "deviceType": "button",  
          "callbackOverrides": {}  
        }  
      }  
    },  
    "tags": {}  
  }  
}
```

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [DescribeProject](#) in *AWS CLI Command Reference*.

disassociate-device-from-placement

The following code example shows how to use `disassociate-device-from-placement`.

AWS CLI

To disassociate a device from a placement

The following `disassociate-device-from-placement` example disassociates the specified device from a placement.

```
aws iot1click-projects disassociate-device-from-placement \  
  --project-name AnytownDumpsters \  
  --placement-name customer217 \  
  --device-template-name empty-dumpster-request
```

This command produces no output.

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [DisassociateDeviceFromPlacement](#) in *AWS CLI Command Reference*.

get-devices-in-placement

The following code example shows how to use `get-devices-in-placement`.

AWS CLI

To list all devices in a placement contained in a project

The following `get-devices-in-placement` example lists all devices in a the specified placement contained in the specified project.

```
aws iot1click-projects get-devices-in-placement \  
  --project-name AnytownDumpsters \  
  --placement-name customer217
```

Output:

```
{
  "devices": {
    "empty-dumpster-request": "G030PM0123456789"
  }
}
```

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [GetDevicesInPlacement](#) in *AWS CLI Command Reference*.

list-placements

The following code example shows how to use `list-placements`.

AWS CLI

To list all AWS IoT 1-Click placements for a project

The following `list-placements` example lists all AWS IoT 1-Click placements for the specified project.

```
aws iot1click-projects list-placements \
  --project-name AnytownDumpsters
```

Output:

```
{
  "placements": [
    {
      "projectName": "AnytownDumpsters",
      "placementName": "customer217",
      "createdDate": 1563488454,
      "updatedAt": 1563488454
    }
  ]
}
```

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [ListPlacements](#) in *AWS CLI Command Reference*.

list-projects

The following code example shows how to use `list-projects`.

AWS CLI

To list all AWS IoT 1-Click projects

The following `list-projects` example list all AWS IoT 1-Click projects in your account.

```
aws iot1click-projects list-projects
```

Output:

```
{
  "projects": [
    {
      "arn": "arn:aws:iot1click:us-west-2:012345678901:projects/
AnytownDumpsters",
      "projectName": "AnytownDumpsters",
      "createdDate": 1563483100,
      "updatedAt": 1563483100,
      "tags": {}
    }
  ]
}
```

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [ListProjects](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags for a project resource

The following `list-tags-for-resource` example list the tags for the specified project resource.

```
aws iot1click-projects list-tags-for-resource \  
  --resource-arn "arn:aws:iot1click:us-west-2:123456789012:projects/  
AnytownDumpsters"
```

Output:

```
{  
  "tags": {  
    "Manager": "Li Juan",  
    "Account": "45215"  
  }  
}
```

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To add tags to a project resource

The following tag-resource example adds two tags to the specified project resource.

```
aws iot1click-projects tag-resource \  
  --cli-input-json file://devices-tag-resource.json
```

Contents of devices-tag-resource.json:

```
{  
  "resourceArn": "arn:aws:iot1click:us-west-2:123456789012:projects/  
AnytownDumpsters",  
  "tags": {  
    "Account": "45215",  
    "Manager": "Li Juan"  
  }  
}
```

This command produces no output.

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use untag-resource.

AWS CLI

To remove tags from a project resource

The following untag-resource example removes the tag with the key name Manager from the specified project.

```
aws iot1click-projects untag-resource \  
  --resource-arn "arn:aws:iot1click:us-west-2:123456789012:projects/  
AnytownDumpsters" \  
  --tag-keys "Manager"
```

This command produces no output.

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-placement

The following code example shows how to use update-placement.

AWS CLI

To update the "attributes" key-value pairs of a placement

The following update-placement example update the "attributes" key-value pairs of a placement.

```
aws iot1click-projects update-placement \  

```

```
--cli-input-json file://update-placement.json
```

Contents of `update-placement.json`:

```
{
  "projectName": "AnytownDumpsters",
  "placementName": "customer217",
  "attributes": {
    "phone": "123-456-7890",
    "location": "123 Any Street Anytown, USA 10001"
  }
}
```

This command produces no output.

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [UpdatePlacement](#) in *AWS CLI Command Reference*.

update-project

The following code example shows how to use `update-project`.

AWS CLI

To update settings for a project

The following `update-project` example updates the description for a project.

```
aws iot1click-projects update-project \
  --project-name AnytownDumpsters \
  --description "All dumpsters (yard waste, recycling, garbage) in the Anytown
region."
```

This command produces no output.

For more information, see [Using AWS IoT 1-Click with the AWS CLI](#) in the *AWS IoT 1-Click Developer Guide*.

- For API details, see [UpdateProject](#) in *AWS CLI Command Reference*.

AWS IoT Analytics examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS IoT Analytics.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

batch-put-message

The following code example shows how to use `batch-put-message`.

AWS CLI

To send a message to a channel

The following `batch-put-message` example sends a message to the specified channel.

```
aws iotanalytics batch-put-message \  
  --cli-binary-format raw-in-base64-out \  
  --cli-input-json file://batch-put-message.json
```

Contents of `batch-put-message.json`:

```
{  
  "channelName": "mychannel",  
  "messages": [  
    {  
      "messageId": "0001",  
      "payload": "eyJhdGVtcGVyYXR1cmUiOiAyMCMCB9"
```

```
    }  
  ]  
}
```

Output:

```
{  
  "batchPutMessageErrorEntries": []  
}
```

For more information, see [BatchPutMessage](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [BatchPutMessage](#) in *AWS CLI Command Reference*.

cancel-pipeline-reprocessing

The following code example shows how to use `cancel-pipeline-reprocessing`.

AWS CLI

To cancel the reprocessing of data through a pipeline

The following `cancel-pipeline-reprocessing` example cancels the reprocessing of data through the specified pipeline.

```
aws iotanalytics cancel-pipeline-reprocessing \  
  --pipeline-name mypipeline \  
  --reprocessing-id "6ad2764f-fb13-4de3-b101-4e74af03b043"
```

This command produces no output.

For more information, see [CancelPipelineReprocessing](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [CancelPipelineReprocessing](#) in *AWS CLI Command Reference*.

create-channel

The following code example shows how to use `create-channel`.

AWS CLI

To create a channel

The following `create-channel` example creates a channel with the specified configuration. A channel collects data from an MQTT topic and archives the raw, unprocessed messages before publishing the data to a pipeline.

```
aws iotanalytics create-channel \  
  --cli-input-json file://create-channel.json
```

Contents of `create-channel.json`:

```
{  
  "channelName": "mychannel",  
  "retentionPeriod": {  
    "unlimited": true  
  },  
  "tags": [  
    {  
      "key": "Environment",  
      "value": "Production"  
    }  
  ]  
}
```

Output:

```
{  
  "channelArn": "arn:aws:iotanalytics:us-west-2:123456789012:channel/mychannel",  
  "channelName": "mychannel",  
  "retentionPeriod": {  
    "unlimited": true  
  }  
}
```

For more information, see [CreateChannel](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [CreateChannel](#) in *AWS CLI Command Reference*.

create-dataset-content

The following code example shows how to use `create-dataset-content`.

AWS CLI

To create the content of a dataset

The following `create-dataset-content` example creates the content of the specified dataset by applying a `queryAction` (an SQL query) or a `containerAction` (executing a containerized application).

```
aws iotanalytics create-dataset-content \  
  --dataset-name mydataset
```

Output:

```
{  
  "versionId": "d494b416-9850-4670-b885-ca22f1e89d62"  
}
```

For more information, see [CreateDatasetContent](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [CreateDatasetContent](#) in *AWS CLI Command Reference*.

create-dataset

The following code example shows how to use `create-dataset`.

AWS CLI

To create a dataset

The following `create-dataset` example creates a dataset. A dataset stores data retrieved from a data store by applying a `queryAction` (a SQL query) or a `containerAction` (executing a containerized application). This operation creates the skeleton of a dataset. You can populate the dataset manually by calling `CreateDatasetContent` or automatically according to a `trigger` you specify.

```
aws iotanalytics create-dataset \  
  --cli-input-json file://create-dataset.json
```

Contents of `create-dataset.json`:

```
{
  "datasetName": "mydataset",
  "actions": [
    {
      "actionName": "myDatasetAction",
      "queryAction": {
        "sqlQuery": "SELECT * FROM mydatastore"
      }
    }
  ],
  "retentionPeriod": {
    "unlimited": true
  },
  "tags": [
    {
      "key": "Environment",
      "value": "Production"
    }
  ]
}
```

Output:

```
{
  "datasetName": "mydataset",
  "retentionPeriod": {
    "unlimited": true
  },
  "datasetArn": "arn:aws:iotanalytics:us-west-2:123456789012:dataset/mydataset"
}
```

For more information, see [CreateDataset](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [CreateDataset](#) in *AWS CLI Command Reference*.

create-datastore

The following code example shows how to use create-datastore.

AWS CLI

To create a data store

The following `create-datastore` example creates a data store, which is a repository for messages.

```
aws iotanalytics create-datastore \  
  --cli-input-json file://create-datastore.json
```

Contents of `create-datastore.json`:

```
{  
  "datastoreName": "mydatastore",  
  "retentionPeriod": {  
    "numberOfDays": 90  
  },  
  "tags": [  
    {  
      "key": "Environment",  
      "value": "Production"  
    }  
  ]  
}
```

Output:

```
{  
  "datastoreName": "mydatastore",  
  "datastoreArn": "arn:aws:iotanalytics:us-west-2:123456789012:datastore/  
mydatastore",  
  "retentionPeriod": {  
    "numberOfDays": 90,  
    "unlimited": false  
  }  
}
```

For more information, see [CreateDatastore](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [CreateDatastore](#) in *AWS CLI Command Reference*.

create-pipeline

The following code example shows how to use `create-pipeline`.

AWS CLI

Create an IoT Analytics Pipeline

The following `create-pipeline` example creates a pipeline. A pipeline consumes messages from a channel and allows you to process the messages before storing them in a data store. You must specify both a channel and a data store activity and, optionally, as many as 23 additional activities in the `pipelineActivities` array.

```
aws iotanalytics create-pipeline \  
  --cli-input-json file://create-pipeline.json
```

Contents of `create-pipeline.json`:

```
{  
  "pipelineName": "mypipeline",  
  "pipelineActivities": [  
    {  
      "channel": {  
        "name": "myChannelActivity",  
        "channelName": "mychannel",  
        "next": "myMathActivity"  
      }  
    },  
    {  
      "datastore": {  
        "name": "myDatastoreActivity",  
        "datastoreName": "mydatastore"  
      }  
    },  
    {  
      "math": {  
        "name": "myMathActivity",  
        "math": "((temp - 32) * 5.0) / 9.0",  
        "attribute": "tempC",  
        "next": "myDatastoreActivity"  
      }  
    }  
  ],  
  "tags": [  
    {  
      "key": "Environment",  
      "value": "Beta"  
    }  
  ]  
}
```

```
    }  
  ]  
}
```

Output:

```
{  
  "pipelineArn": "arn:aws:iotanalytics:us-west-2:123456789012:pipeline/  
mypipeline",  
  "pipelineName": "mypipeline"  
}
```

For more information, see [CreatePipeline](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [CreatePipeline](#) in *AWS CLI Command Reference*.

delete-channel

The following code example shows how to use `delete-channel`.

AWS CLI

Delete an IoT Analytics Channel

The following `delete-channel` example deletes the specified channel.

```
aws iotanalytics delete-channel \  
  --channel-name mychannel
```

This command produces no output.

For more information, see [DeleteChannel](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [DeleteChannel](#) in *AWS CLI Command Reference*.

delete-dataset-content

The following code example shows how to use `delete-dataset-content`.

AWS CLI

To delete dataset content

The following `delete-dataset-content` example deletes the content of the specified dataset.

```
aws iotanalytics delete-dataset-content \  
  --dataset-name mydataset
```

This command produces no output.

For more information, see [DeleteDatasetContent](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [DeleteDatasetContent](#) in *AWS CLI Command Reference*.

delete-dataset

The following code example shows how to use `delete-dataset`.

AWS CLI

To delete a dataset

The following `delete-dataset` example deletes the specified dataset. You don't have to delete the content of the dataset before you perform this operation.

```
aws iotanalytics delete-dataset \  
  --dataset-name mydataset
```

This command produces no output.

For more information, see [DeleteDataset](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [DeleteDataset](#) in *AWS CLI Command Reference*.

delete-datastore

The following code example shows how to use `delete-datastore`.

AWS CLI

To delete a data store

The following `delete-datastore` example deletes the specified data store.

```
aws iotanalytics delete-datastore \  
  --datastore-name mydatastore
```

This command produces no output.

For more information, see [DeleteDatastore](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [DeleteDatastore](#) in *AWS CLI Command Reference*.

delete-pipeline

The following code example shows how to use `delete-pipeline`.

AWS CLI

To delete a pipeline

The following `delete-pipeline` example deletes the specified pipeline.

```
aws iotanalytics delete-pipeline \  
  --pipeline-name mypipeline
```

This command produces no output.

For more information, see [DeletePipeline](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [DeletePipeline](#) in *AWS CLI Command Reference*.

describe-channel

The following code example shows how to use `describe-channel`.

AWS CLI

To retrieve information about a channel

The following `describe-channel` example displays details, including statistics, for the specified channel.

```
aws iotanalytics describe-channel \  
  --channel-name mychannel \  
  --include-statistics
```

Output:

```
{
  "statistics": {
    "size": {
      "estimatedSizeInBytes": 402.0,
      "estimatedOn": 1561504380.0
    }
  },
  "channel": {
    "status": "ACTIVE",
    "name": "mychannel",
    "lastUpdateTime": 1557860351.001,
    "creationTime": 1557860351.001,
    "retentionPeriod": {
      "unlimited": true
    },
    "arn": "arn:aws:iotanalytics:us-west-2:123456789012:channel/mychannel"
  }
}
```

For more information, see [DescribeChannel](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [DescribeChannel](#) in *AWS CLI Command Reference*.

describe-dataset

The following code example shows how to use `describe-dataset`.

AWS CLI**To retrieve information about a dataset**

The following `describe-dataset` example displays details for the specified dataset.

```
aws iotanalytics describe-dataset \
  --dataset-name mydataset
```

Output:

```
{
  "dataset": {
```

```
"status": "ACTIVE",
"contentDeliveryRules": [],
"name": "mydataset",
"lastUpdateTime": 1557859240.658,
"triggers": [],
"creationTime": 1557859240.658,
"actions": [
  {
    "actionName": "query_32",
    "queryAction": {
      "sqlQuery": "SELECT * FROM mydatastore",
      "filters": []
    }
  }
],
"retentionPeriod": {
  "numberOfDays": 90,
  "unlimited": false
},
"arn": "arn:aws:iotanalytics:us-west-2:123456789012:dataset/mydataset"
}
```

For more information, see [DescribeDataset](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [DescribeDataset](#) in *AWS CLI Command Reference*.

describe-datastore

The following code example shows how to use describe-datastore.

AWS CLI

To retrieve information about a data store

The following describe-datastore example displays details, including statistics, for the specified data store.

```
aws iotanalytics describe-datastore \
  --datastore-name mydatastore \
  --include-statistics
```

Output:

```
{
  "datastore": {
    "status": "ACTIVE",
    "name": "mydatastore",
    "lastUpdateTime": 1557858971.02,
    "creationTime": 1557858971.02,
    "retentionPeriod": {
      "unlimited": true
    },
    "arn": "arn:aws:iotanalytics:us-west-2:123456789012:datastore/mydatastore"
  },
  "statistics": {
    "size": {
      "estimatedSizeInBytes": 397.0,
      "estimatedOn": 1561592040.0
    }
  }
}
```

For more information, see [DescribeDatastore](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [DescribeDatastore](#) in *AWS CLI Command Reference*.

describe-logging-options

The following code example shows how to use `describe-logging-options`.

AWS CLI

To retrieve the current logging options

The following `describe-logging-options` example displays the current AWS IoT Analytics logging options.

```
aws iotanalytics describe-logging-options
```

This command produces no output. Output:

```
{
  "loggingOptions": {
    "roleArn": "arn:aws:iam::123456789012:role/service-role/myIoTAnalyticsRole",
    "enabled": true,
  }
}
```

```
    "level": "ERROR"
  }
}
```

For more information, see [DescribeLoggingOptions](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [DescribeLoggingOptions](#) in *AWS CLI Command Reference*.

describe-pipeline

The following code example shows how to use `describe-pipeline`.

AWS CLI

To retrieve information about a pipeline

The following `describe-pipeline` example displays details for the specified pipeline.

```
aws iotanalytics describe-pipeline \  
  --pipeline-name mypipeline
```

Output:

```
{  
  "pipeline": {  
    "activities": [  
      {  
        "channel": {  
          "channelName": "mychannel",  
          "name": "mychannel_28",  
          "next": "mydatastore_29"  
        }  
      },  
      {  
        "datastore": {  
          "datastoreName": "mydatastore",  
          "name": "mydatastore_29"  
        }  
      }  
    ],  
    "name": "mypipeline",  
    "lastUpdateTime": 1561676362.515,  
    "creationTime": 1557859124.432,  
  }  
}
```

```
    "reprocessingSummaries": [
      {
        "status": "SUCCEEDED",
        "creationTime": 1561676362.189,
        "id": "6ad2764f-fb13-4de3-b101-4e74af03b043"
      }
    ],
    "arn": "arn:aws:iotanalytics:us-west-2:123456789012:pipeline/mypipeline"
  }
}
```

For more information, see [DescribePipeline](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [DescribePipeline](#) in *AWS CLI Command Reference*.

get-dataset-content

The following code example shows how to use `get-dataset-content`.

AWS CLI

To retrieve the contents of a dataset

The following `get-dataset-content` example retrieves the contents of a dataset as presigned URIs.

```
aws iotanalytics get-dataset-content --dataset-name mydataset
```

Output:

```
{
  "status": {
    "state": "SUCCEEDED"
  },
  "timestamp": 1557863215.995,
  "entries": [
    {
      "dataURI": "https://aws-radiant-
dataset-12345678-1234-1234-1234-123456789012.s3.us-west-2.amazonaws.com/
results/12345678-e8b3-46ba-b2dd-efe8d86cf385.csv?X-Amz-Security-Token=...-Amz-
Algorithm=AWS4-HMAC-SHA256&X-Amz-Date=20190628T173437Z&X-Amz-SignedHeaders=host&X-
Amz-Expires=7200&X-Amz-Credential=...F20190628%2Fus-west-2%2Fs3%2Faws4_request&X-
Amz-Signature=..."
    }
  ]
}
```

```
    }  
  ]  
}
```

For more information, see [GetDatasetContent](#) in the *guide*.

- For API details, see [GetDatasetContent](#) in *AWS CLI Command Reference*.

list-channels

The following code example shows how to use `list-channels`.

AWS CLI

To retrieve a list of channels

The following `list-channels` example displays summary information for the available channels.

```
aws iotanalytics list-channels
```

Output:

```
{  
  "channelSummaries": [  
    {  
      "status": "ACTIVE",  
      "channelName": "mychannel",  
      "creationTime": 1557860351.001,  
      "lastUpdateTime": 1557860351.001  
    }  
  ]  
}
```

For more information, see [ListChannels](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [ListChannels](#) in *AWS CLI Command Reference*.

list-dataset-contents

The following code example shows how to use `list-dataset-contents`.

AWS CLI

To list information about dataset contents

The following `list-dataset-contents` example lists information about dataset contents that have been created.

```
aws iotanalytics list-dataset-contents \  
  --dataset-name mydataset
```

Output:

```
{  
  "datasetContentSummaries": [  
    {  
      "status": {  
        "state": "SUCCEEDED"  
      },  
      "scheduleTime": 1557863215.995,  
      "version": "b10ea2a9-66c1-4d99-8d1f-518113b738d0",  
      "creationTime": 1557863215.995  
    }  
  ]  
}
```

For more information, see [ListDatasetContents](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [ListDatasetContents](#) in *AWS CLI Command Reference*.

list-datasets

The following code example shows how to use `list-datasets`.

AWS CLI

To retrieve information about datasets

The following `list-datasets` example lists summary information about available datasets.

```
aws iotanalytics list-datasets
```

Output:

```
{
  "datasetSummaries": [
    {
      "status": "ACTIVE",
      "datasetName": "mydataset",
      "lastUpdateTime": 1557859240.658,
      "triggers": [],
      "creationTime": 1557859240.658,
      "actions": [
        {
          "actionName": "query_32",
          "actionType": "QUERY"
        }
      ]
    }
  ]
}
```

For more information, see [ListDatasets](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [ListDatasets](#) in *AWS CLI Command Reference*.

list-datastores

The following code example shows how to use `list-datastores`.

AWS CLI

To retrieve a list of data stores

The following `list-datastores` example displays summary information about the available data stores.

```
aws iotanalytics list-datastores
```

Output:

```
{
  "datastoreSummaries": [
    {
      "status": "ACTIVE",
      "datastoreName": "mydatastore",
```

```
        "creationTime": 1557858971.02,  
        "lastUpdateTime": 1557858971.02  
      }  
    ]  
  }
```

For more information, see [ListDatastores](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [ListDatastores](#) in *AWS CLI Command Reference*.

list-pipelines

The following code example shows how to use `list-pipelines`.

AWS CLI

To retrieve a list of pipelines

The following `list-pipelines` example displays a list of available pipelines.

```
aws iotanalytics list-pipelines
```

Output:

```
{  
  "pipelineSummaries": [  
    {  
      "pipelineName": "mypipeline",  
      "creationTime": 1557859124.432,  
      "lastUpdateTime": 1557859124.432,  
      "reprocessingSummaries": []  
    }  
  ]  
}
```

For more information, see [ListPipelines](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [ListPipelines](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags for a resource

The following `list-tags-for-resource` example Lists the tags that you have attached to the specified resource.

```
aws iotanalytics list-tags-for-resource \  
  --resource-arn "arn:aws:iotanalytics:us-west-2:123456789012:channel/mychannel"
```

Output:

```
{  
  "tags": [  
    {  
      "value": "bar",  
      "key": "foo"  
    }  
  ]  
}
```

For more information, see [ListTagsForResource](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

put-logging-options

The following code example shows how to use `put-logging-options`.

AWS CLI

To set or update logging options

The following `put-logging-options` example sets or updates the AWS IoT Analytics logging options. If you update the value of any `loggingOptions` field, it can take up to one minute for the change to take effect. Also, if you change the policy attached to the role you specified in the `roleArn` field (for example, to correct an invalid policy) it can take up to five minutes for that change to take effect.

```
aws iotanalytics put-logging-options \  
  \
```

```
--cli-input-json file://put-logging-options.json
```

Contents of `put-logging-options.json`:

```
{
  "loggingOptions": {
    "roleArn": "arn:aws:iam::123456789012:role/service-role/myIoTAnalyticsRole",
    "level": "ERROR",
    "enabled": true
  }
}
```

This command produces no output.

For more information, see [PutLoggingOptions](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [PutLoggingOptions](#) in *AWS CLI Command Reference*.

run-pipeline-activity

The following code example shows how to use `run-pipeline-activity`.

AWS CLI

To simulate a pipeline activity

The following `run-pipeline-activity` example simulates the results of running a pipeline activity on a message payload.

```
aws iotanalytics run-pipeline-activity \
  --pipeline-activity file://maths.json \
  --payloads file://payloads.json
```

Contents of `maths.json`:

```
{
  "math": {
    "name": "MyMathActivity",
    "math": "((temp - 32) * 5.0) / 9.0",
    "attribute": "tempC"
  }
}
```

```
}
```

Contents of `payloads.json`:

```
[
  "{\"humidity\": 52, \"temp\": 68 }",
  "{\"humidity\": 52, \"temp\": 32 }"
]
```

Output:

```
{
  "logResult": "",
  "payloads": [
    "eyJodW1pZG10eSI6NTIsInRlbXAiOjY4LCJ0ZW1wQyI6MjB9",
    "eyJodW1pZG10eSI6NTIsInRlbXAiOjMyLCJ0ZW1wQyI6MH0="
  ]
}
```

For more information, see [RunPipelineActivity](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [RunPipelineActivity](#) in *AWS CLI Command Reference*.

sample-channel-data

The following code example shows how to use `sample-channel-data`.

AWS CLI

To retrieve sample messages from a channel

The following `sample-channel-data` example retrieves a sample of messages from the specified channel ingested during the specified timeframe. You can retrieve up to 10 messages.

```
aws iotanalytics sample-channel-data \
  --channel-name mychannel
```

Output:

```
{
```

```
"payloads": [  
  "eyJhdGVtcGVyYXR1cmUiOiAyMCM9",  
  "eyJhZm9vIjogImJhcnIgfQ=="  
]
```

For more information, see [SampleChannelData](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [SampleChannelData](#) in *AWS CLI Command Reference*.

start-pipeline-reprocessing

The following code example shows how to use `start-pipeline-reprocessing`.

AWS CLI

To start pipeline reprocessing

The following `start-pipeline-reprocessing` example starts the reprocessing of raw message data through the specified pipeline.

```
aws iotanalytics start-pipeline-reprocessing \  
  --pipeline-name mypipeline
```

Output:

```
{  
  "reprocessingId": "6ad2764f-fb13-4de3-b101-4e74af03b043"  
}
```

For more information, see [StartPipelineReprocessing](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [StartPipelineReprocessing](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add or modify tags for a resource

The following `tag-resource` example adds to or modifies the tags attached to the specified resource.

```
aws iotanalytics tag-resource \  
  --resource-arn "arn:aws:iotanalytics:us-west-2:123456789012:channel/mychannel" \  
  --tags "[{"key": "Environment", "value": "Production"}]"
```

This command produces no output.

For more information, see [TagResource](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from a resource

The following `untag-resource` example removes the tags with the specified key names from the specified resource.

```
aws iotanalytics untag-resource \  
  --resource-arn "arn:aws:iotanalytics:us-west-2:123456789012:channel/mychannel" \  
  --tag-keys ["Environment"]
```

This command produces no output.

For more information, see `UntagResource` <https://docs.aws.amazon.com/iotanalytics/latest/APIReference/API_UntagResource.html> in the *AWS IoT Analytics API Reference*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-channel

The following code example shows how to use `update-channel`.

AWS CLI

To modify a channel

The following `update-channel` example modifies the settings for the specified channel.

```
aws iotanalytics update-channel \  
  --cli-input-json file://update-channel.json
```

Contents of `update-channel.json`:

```
{  
  "channelName": "mychannel",  
  "retentionPeriod": {  
    "numberOfDays": 92  
  }  
}
```

This command produces no output.

For more information, see [UpdateChannel](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [UpdateChannel](#) in *AWS CLI Command Reference*.

update-dataset

The following code example shows how to use `update-dataset`.

AWS CLI

To update a dataset

The following `update-dataset` example modifies the settings of the specified dataset.

```
aws iotanalytics update-dataset \  
  --cli-input-json file://update-dataset.json
```

Contents of `update-dataset.json`:

```
{  
  "datasetName": "mydataset",  
  "actions": [  
    {  
      "actionName": "myDatasetUpdateAction",  
      "queryAction": {  
        "sqlQuery": "SELECT * FROM mydatastore"  
      }  
    }  
  ]  
}
```

```
    }
  }
],
"retentionPeriod": {
  "numberOfDays": 92
}
}
```

This command produces no output.

For more information, see `UpdateDataset` <https://docs.aws.amazon.com/iotanalytics/latest/APIReference/API_UpdateDataset.html> in the *AWS IoT Analytics API Reference*.

- For API details, see [UpdateDataset](#) in *AWS CLI Command Reference*.

update-datastore

The following code example shows how to use `update-datastore`.

AWS CLI

To update a data store

The following `update-datastore` example modifies the settings of the specified data store.

```
aws iotanalytics update-datastore \
  --cli-input-json file://update-datastore.json
```

Contents of `update-datastore.json`:

```
{
  "datastoreName": "mydatastore",
  "retentionPeriod": {
    "numberOfDays": 93
  }
}
```

This command produces no output.

For more information, see [UpdateDatastore](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [UpdateDatastore](#) in *AWS CLI Command Reference*.

update-pipeline

The following code example shows how to use update-pipeline.

AWS CLI

To update a pipeline

The following update-pipeline example modifies the settings of the specified pipeline. You must specify both a channel and a data store activity and, optionally, as many as 23 additional activities, in the pipelineActivities array.

```
aws iotanalytics update-pipeline \  
  --cli-input-json file://update-pipeline.json
```

Contents of update-pipeline.json:

```
{  
  "pipelineName": "mypipeline",  
  "pipelineActivities": [  
    {  
      "channel": {  
        "name": "myChannelActivity",  
        "channelName": "mychannel",  
        "next": "myMathActivity"  
      }  
    },  
    {  
      "datastore": {  
        "name": "myDatastoreActivity",  
        "datastoreName": "mydatastore"  
      }  
    },  
    {  
      "math": {  
        "name": "myMathActivity",  
        "math": "(((temp - 32) * 5.0) / 9.0) + 273.15",  
        "attribute": "tempK",  
        "next": "myDatastoreActivity"  
      }  
    }  
  ]  
}
```

```
}
```

This command produces no output.

For more information, see [UpdatePipeline](#) in the *AWS IoT Analytics API Reference*.

- For API details, see [UpdatePipeline](#) in *AWS CLI Command Reference*.

Device Advisor examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Device Advisor.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-suite-definition

The following code example shows how to use `create-suite-definition`.

AWS CLI

Example 1: To create an IoT Device Advisor test suite

The following `create-suite-definition` example creates a device advisor test suite in the AWS IoT with the specified suite definition configuration.

```
aws iotdeviceadvisor create-suite-definition \  
  --suite-definition-configuration '{ \  
    "name": "Test Suite",
```

```

    "suiteDefinitionName": "TestSuiteName", \
    "devices": [{"thingArn": "arn:aws:iot:us-east-1:123456789012:thing/
MyIotThing"}], \
    "intendedForQualification": false, \
    "rootGroup": "{ \"configuration\": {}, \"tests\": [{ \"name\": \"MQTT Connect\",
\"configuration\": { \"EXECUTION_TIMEOUT\": 120 }, \"tests\": [{ \"name\": \"MQTT_Connect\",
\"configuration\": {}, \"test\": { \"id\": \"MQTT_Connect\", \"testCase\": null, \"version
\": \"0.0.0\" } } ] } ] }", \
    "devicePermissionRoleArn": "arn:aws:iam::123456789012:role/Myrole"'

```

Output:

```

{
  "suiteDefinitionId": "0jtsigio7yenu",
  "suiteDefinitionArn": "arn:aws:iotdeviceadvisor:us-
east-1:123456789012:suitedefinition/0jtsigio7yenu",
  "suiteDefinitionName": "TestSuiteName",
  "createdAt": "2022-12-02T11:38:13.263000-05:00"
}

```

For more information, see [Create a test suite definition](#) in the *AWS IoT Core Developer Guide*.

Example 2: To create an IoT Device Advisor Latest Qualification test suite

The following `create-suite-definition` example creates a device advisor qualification test suite with the latest version in the AWS IoT with the specified suite definition configuration.

```

aws iotdeviceadvisor create-suite-definition \
  --suite-definition-configuration '{ \
    "suiteDefinitionName": "TestSuiteName", \
    "devices": [{"thingArn": "arn:aws:iot:us-east-1:123456789012:thing/
MyIotThing"}], \
    "intendedForQualification": true, \
    "rootGroup": "", \
    "devicePermissionRoleArn": "arn:aws:iam::123456789012:role/Myrole"'

```

Output:

```

{
  "suiteDefinitionId": "txgsuolk2myj",
  "suiteDefinitionArn": "arn:aws:iotdeviceadvisor:us-
east-1:123456789012:suitedefinition/txgsuolk2myj",

```

```
"suiteDefinitionName": "TestSuiteName",  
"createdAt": "2022-12-02T11:38:13.263000-05:00"  
}
```

For more information, see [Create a test suite definition](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [CreateSuiteDefinition](#) in *AWS CLI Command Reference*.

delete-suite-definition

The following code example shows how to use `delete-suite-definition`.

AWS CLI

To delete the IoT Device Advisor test suite

The following `delete-suite-definition` example deletes the device advisor test suite with the specified suite definition ID.

```
aws iotdeviceadvisor delete-suite-definition \  
  --suite-definition-id 0jtsgio7yenu
```

This command produces no output.

For more information, see [DeleteSuiteDefinition](#) in the *AWS IoT API Reference*.

- For API details, see [DeleteSuiteDefinition](#) in *AWS CLI Command Reference*.

get-endpoint

The following code example shows how to use `get-endpoint`.

AWS CLI

Example 1: To get the information about an IoT Device Advisor Account-level endpoint

The following `get-endpoint` example gets the information about a device advisor Account-level test endpoint.

```
aws iotdeviceadvisor get-endpoint
```

Output:

```
{
  "endpoint": "t6y4c143x9sfo.deviceadvisor.iot.us-east-1.amazonaws.com"
}
```

Example 2: To get the information about an IoT Device Advisor Device-level endpoint

The following `get-endpoint` example gets the information about a device advisor device-level test endpoint with the specified `thing-arn` or `certificate-arn`.

```
aws iotdeviceadvisor get-endpoint \
  --thing-arn arn:aws:iot:us-east-1:123456789012:thing/MyIotThing
```

Output:

```
{
  "endpoint": "tdb7719be5t6y4c143x9sfo.deviceadvisor.iot.us-east-1.amazonaws.com"
}
```

For more information, see [Get a test endpoint](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [GetEndpoint](#) in *AWS CLI Command Reference*.

get-suite-definition

The following code example shows how to use `get-suite-definition`.

AWS CLI

To get the information about an IoT Device Advisor test suite

The following `get-suite-definition` example get the information about a aevice advisor test suite with the specified suite definition ID.

```
aws iotdeviceadvisor get-suite-definition \
  --suite-definition-id qqcsmtyyjabl
```

Output:

```
{
  "suiteDefinitionId": "qqcsmtyyjabl",
```

```

    "suiteDefinitionArn": "arn:aws:iotdeviceadvisor:us-
east-1:123456789012:suitedefinition/qqcsmtyyjabl",
    "suiteDefinitionVersion": "v1",
    "latestVersion": "v1",
    "suiteDefinitionConfiguration": {
      "suiteDefinitionName": "MQTT connection",
      "devices": [],
      "intendedForQualification": false,
      "isLongDurationTest": false,
      "rootGroup": "{\\"configuration\\":{\\},\\"tests\\":[{\\"id\\":\\"uta5d9j1kvwc\\",
\\"name\\":\\"Test group 1\\",\\"configuration\\":{\\},\\"tests\\":[{\\"id\\":\\"awr8pq5vc9yp\\",
\\"name\\":\\"MQTT Connect\\",\\"configuration\\":{\\},\\"test\\":{\\"id\\":\\"MQTT_Connect\\",
\\"testCase\\":null,\\"version\\":\\"0.0.0\\"}]}]}]",
      "devicePermissionRoleArn": "arn:aws:iam::123456789012:role/Myrole",
      "protocol": "MqttV3_1_1"
    },
    "createdAt": "2022-11-11T22:28:52.389000-05:00",
    "lastModifiedAt": "2022-11-11T22:28:52.389000-05:00",
    "tags": {}
  }
}

```

For more information, see [Get a test suite definition](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [GetSuiteDefinition](#) in *AWS CLI Command Reference*.

get-suite-run-report

The following code example shows how to use `get-suite-run-report`.

AWS CLI

To get the information about an IoT Device Advisor qualifying test suite run report

The following `get-suite-run-report` example gets the report download link for a successful device advisor qualifying test suite run with the specified suite definition ID and suite run ID.

```

aws iotdeviceadvisor get-suite-run-report \
  --suite-definition-id ztvb5aek4w4x \
  --suite-run-id p6awv83nre6v

```

Output:

```
{
```



```
"qualificationReportDownloadUrl": "https://senate-apn-reports-us-east-1-
prod.s3.amazonaws.com/report.downloadlink"
}
```

For more information, see [Get a qualification report for a successful qualification test suite run](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [GetSuiteRunReport](#) in *AWS CLI Command Reference*.

get-suite-run

The following code example shows how to use `get-suite-run`.

AWS CLI

To get the information about an IoT Device Advisor test suite run status

The following `get-suite-run` example gets the information about a device advisor test suite run status with the specified suite definition ID and suite run ID.

```
aws iotdeviceadvisor get-suite-run \
  --suite-definition-id qqcsmtyyjabl \
  --suite-run-id nzlfyhaa18oa
```

Output:

```
{
  "suiteDefinitionId": "qqcsmtyyjabl",
  "suiteDefinitionVersion": "v1",
  "suiteRunId": "nzlfyhaa18oa",
  "suiteRunArn": "arn:aws:iotdeviceadvisor:us-east-1:123456789012:suiterun/
qqcsmtyyjabl/nzlfyhaa18oa",
  "suiteRunConfiguration": {
    "primaryDevice": {
      "thingArn": "arn:aws:iot:us-east-1:123456789012:thing/MyIotThing",
      "certificateArn": "arn:aws:iot:us-east-1:123456789012:cert/certFile"
    },
    "parallelRun": false
  },
  "testResult": {
    "groups": [
      {
        "groupId": "uta5d9j1kvwc",
```

```

        "groupName": "Test group 1",
        "tests": [
            {
                "testCaseRunId": "2ve2twrqyr0s",
                "testCaseDefinitionId": "awr8pq5vc9yp",
                "testCaseDefinitionName": "MQTT Connect",
                "status": "PASS",
                "startTime": "2022-11-12T00:01:53.693000-05:00",
                "endTime": "2022-11-12T00:02:15.443000-05:00",
                "logUrl": "https://console.aws.amazon.com/
cloudwatch/home?region=us-east-1#logEventViewer:group=/aws/iot/deviceadvisor/
qqcsmtyyjabl;stream=nzlfyhaa18oa_2ve2twrqyr0s",
                "warnings": "null",
                "failure": "null"
            }
        ]
    },
    ],
    },
    "startTime": "2022-11-12T00:01:52.673000-05:00",
    "endTime": "2022-11-12T00:02:16.496000-05:00",
    "status": "PASS",
    "tags": {}
}

```

For more information, see [Get a test suite run](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [GetSuiteRun](#) in *AWS CLI Command Reference*.

list-suite-definitions

The following code example shows how to use `list-suite-definitions`.

AWS CLI

Example 1: To list the IoT Device Advisor test suites you created

The following `list-suite-definitions` example lists up to 25 device advisor test suites you created in AWS IoT. If you have more than 25 test suites, the "nextToken" will be shown in the output. You can use this "nextToken" to show the rest of the test suites you created.

```
aws iotdeviceadvisor list-suite-definitions
```

Output:

```
{
  "suiteDefinitionInformationList": [
    {
      "suiteDefinitionId": "3hsn88h4p2g5",
      "suiteDefinitionName": "TestSuite1",
      "defaultDevices": [
        {
          "thingArn": "arn:aws:iot:us-east-1:123456789012:thing/
MyIotThing"
        }
      ],
      "intendedForQualification": false,
      "isLongDurationTest": false,
      "protocol": "MqttV3_1_1",
      "createdAt": "2022-11-17T14:15:56.830000-05:00"
    },
    {
      .....
    }
  ],
  "nextToken": "nextTokenValue"
}
```

Example 2: To list the IoT Device Advisor test suites you created with the specified settings

The following `list-suite-definitions` example lists device advisor test suites you created in AWS IoT with the specified `max-result` number. If you have more test suites than the `max` number, the `nextToken` will be shown in the output. If you have `nextToken`, you can use `nextToken` to show the test suites you created that weren't shown before.

```
aws iotdeviceadvisor list-suite-definitions \
  --max-result 1 \
  --next-token "nextTokenValue"
```

Output:

```
{
  "suiteDefinitionInformationList": [
    {
      "suiteDefinitionId": "ztvb5aew4w4x",
```

```

        "suiteDefinitionName": "TestSuite2",
        "defaultDevices": [],
        "intendedForQualification": true,
        "isLongDurationTest": false,
        "protocol": "MqttV3_1_1",
        "createdAt": "2022-11-17T14:15:56.830000-05:00"
    }
],
"nextToken": "nextTokenValue"
}

```

For more information, see [ListSuiteDefinitions](#) in the *AWS IoT API Reference*.

- For API details, see [ListSuiteDefinitions](#) in *AWS CLI Command Reference*.

list-suite-runs

The following code example shows how to use `list-suite-runs`.

AWS CLI

Example 1: To list all information about the specified IoT Device Advisor test suite runs status

The following `list-suite-runs` example lists all information about a device advisor test suite runs status with the specified suite definition ID. If you have more than 25 test suite runs, the "nextToken" will be shown in the output. You can use this "nextToken" to show the rest of the test suite runs.

```

aws iotdeviceadvisor list-suite-runs \
  --suite-definition-id ztvb5aew4w4x

```

Output:

```

{
  "suiteRunsList": [
    {
      "suiteDefinitionId": "ztvb5aew4w4x",
      "suiteDefinitionVersion": "v1",
      "suiteDefinitionName": "TestSuite",
      "suiteRunId": "p6awv89nre6v",

```

```

        "createdAt": "2022-12-01T16:33:14.212000-05:00",
        "startedAt": "2022-12-01T16:33:15.710000-05:00",
        "endAt": "2022-12-01T16:42:03.323000-05:00",
        "status": "PASS",
        "passed": 6,
        "failed": 0
    }
]
}

```

Example 2: To list information about the specified IoT Device Advisor test suite runs status with the specified settings

The following `list-suite-runs` example lists information about a device advisor test suite runs status with the specified suite definition ID and the specified max-result number. If you have more test suite runs than the max number, the "nextToken" will be shown in the output. If you have "nextToken", you can use "nextToken" to show the test suite runs that weren't shown before.

```

aws iotdeviceadvisor list-suite-runs \
  --suite-definition-id qqcsmtyyjam1 \
  --max-result 1 \
  --next-token "nextTokenValue"

```

Output:

```

{
  "suiteRunsList": [
    {
      "suiteDefinitionId": "qqcsmtyyjam1",
      "suiteDefinitionVersion": "v1",
      "suiteDefinitionName": "MQTT connection",
      "suiteRunId": "gz9vm2s6d2jy",
      "createdAt": "2022-12-01T20:10:27.079000-05:00",
      "startedAt": "2022-12-01T20:10:28.003000-05:00",
      "endAt": "2022-12-01T20:10:45.084000-05:00",
      "status": "STOPPED",
      "passed": 0,
      "failed": 0
    }
  ],
  "nextToken": "nextTokenValue"
}

```

```
}
```

For more information, see [ListSuiteRuns](#) in the *AWS IoT API Reference*.

- For API details, see [ListSuiteRuns](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags attached to an IoT Device Advisor resource

The following `list-tags-for-resource` example lists the tags attached to a device advisor resource. The device advisor resource can be a `Suitedefinition-Arn` or a `Suiterun-Arn`.

```
aws iotdeviceadvisor list-tags-for-resource \
  --resource-arn arn:aws:iotdeviceadvisor:us-east-1:123456789012:suitedefinition/
ba0uyjpg38ny
```

Output:

```
{
  "tags": {
    "TestTagKey": "TestTagValue"
  }
}
```

For more information, see [ListTagsForResource](#) in the *AWS IoT API Reference* and [Resource types defined by AWS IoT Core Device Advisor](#) in the *Service Authorization Reference*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

start-suite-run

The following code example shows how to use `start-suite-run`.

AWS CLI

To start an IoT Device Advisor test suite run

The following `start-suite-run` example lists the available widgets in your AWS account.

```
aws iotdeviceadvisor start-suite-run \  
  --suite-definition-id qqcsmtyyjabl \  
  --suite-definition-version v1 \  
  --suite-run-configuration '{"primaryDevice":{"thingArn": "arn:aws:iot:us-  
east-1:123456789012:thing/MyIoTThing", "certificateArn": "arn:aws:iot:us-  
east-1:123456789012:cert/certFile"}}'
```

Output:

```
{  
  "suiteRunId": "pwmucgw71t9s",  
  "suiteRunArn": "arn:aws:iotdeviceadvisor:us-east-1:123456789012:suiterun/  
qqcsmtyyjabl/pwmucgw71k9s",  
  "createdAt": "2022-12-02T15:43:05.581000-05:00"  
}
```

For more information, see [Start a test suite run](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [StartSuiteRun](#) in *AWS CLI Command Reference*.

stop-suite-run

The following code example shows how to use `stop-suite-run`.

AWS CLI

To stop an IoT Device Advisor test suite that is currently running

The following `stop-suite-run` example stops a device advisor test suite that is currently running with the specified suite definition ID and suite run ID.

```
aws iotdeviceadvisor stop-suite-run \  
  --suite-definition-id qqcsmtyyjabl \  
  --suite-run-id nzlfyhaa18oa
```

This command produces no output.

For more information, see [Stop a test suite run](#) in the *AWS IoT Core Developer Guide*.

- For API details, see [StopSuiteRun](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add to and modify the existing tags of an IoT Device Advisor resource

The following `tag-resource` example adds to and modifies the existing tags of a device advisor resource with the specified resource arn and tags. The device advisor resource can be a `Suitedefinition-Arn` or a `Suiterun-Arn`.

```
aws iotdeviceadvisor tag-resource \  
  --resource-arn arn:aws:iotdeviceadvisor:us-east-1:123456789012:suitedefinition/  
ba0uyjpg38ny \  
  --tags '{"TagKey": "TagValue"}
```

This command produces no output.

For more information, see [TagResource](#) in the *AWS IoT API Reference* and [Resource types defined by AWS IoT Core Device Advisor](#) in the *Service Authorization Reference*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove the existing tags from an IoT Device Advisor resource

The following `untag-resource` example removes the existing tags from a device advisor resource with the specified resource arn and tag key. The device advisor resource can be a `Suitedefinition-Arn` or a `Suiterun-Arn`.

```
aws iotdeviceadvisor untag-resource \  
  --resource-arn arn:aws:iotdeviceadvisor:us-east-1:123456789012:suitedefinition/  
ba0uyjpg38ny \  
  --tag-keys "TagKey"
```


This command produces no output.

For more information, see [UntagResource](#) in the *AWS IoT API Reference* and [Resource types defined by AWS IoT Core Device Advisor](#) in the *Service Authorization Reference*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-suite-definition

The following code example shows how to use update-suite-definition.

AWS CLI

Example 1: To update an IoT Device Advisor test suite

The following update-suite-definition example updates a device advisor test suite in the AWS IoT with the specified suite definition ID and suite definition configuration.

```
aws iotdeviceadvisor update-suite-definition \
  --suite-definition-id 3hsn88h4p2g5 \
  --suite-definition-configuration '{ \
    "suiteDefinitionName": "TestSuiteName", \
    "devices": [{"thingArn": "arn:aws:iot:us-east-1:123456789012:thing/MyIotThing"}], \
    "intendedForQualification": false, \
    "rootGroup": "{\ "configuration\": {}, \ "tests\": [{\ "name\": \ "MQTT Connect\ ", \ "configuration\": {\ "EXECUTION_TIMEOUT\ ": 120}, \ "tests\": [{\ "name\": \ "MQTT_Connect\ ", \ "configuration\": {}, \ "test\": {\ "id\": \ "MQTT_Connect\ ", \ "testCase\": null, \ "version\ ": \ "0.0.0\ "}}]}]}", \
    "devicePermissionRoleArn": "arn:aws:iam::123456789012:role/Myrole" }
```

Output:

```
{
  "suiteDefinitionId": "3hsn88h4p2g5",
  "suiteDefinitionName": "TestSuiteName",
  "suiteDefinitionVersion": "v3",
  "createdAt": "2022-11-17T14:15:56.830000-05:00",
  "lastUpdatedAt": "2022-12-02T16:02:45.857000-05:00"
}
```

Example 2: To update an IoT Device Advisor Qualification test suite

The following `update-suite-definition` example updates a device advisor qualification test suite in the AWS IoT with the specified suite definition ID and suite definition configuration.

```
aws iotdeviceadvisor update-suite-definition \  
  --suite-definition-id txgsuolk2myj \  
  --suite-definition-configuration '{  
    "suiteDefinitionName": "TestSuiteName", \  
    "devices": [{"thingArn": "arn:aws:iot:us-east-1:123456789012:thing/  
MyIoTThing"}], \  
    "intendedForQualification": true, \  
    "rootGroup": "", \  
    "devicePermissionRoleArn": "arn:aws:iam::123456789012:role/Myrole"}'
```

Output:

```
{  
  "suiteDefinitionId": "txgsuolk2myj",  
  "suiteDefinitionName": "TestSuiteName",  
  "suiteDefinitionVersion": "v3",  
  "createdAt": "2022-11-17T14:15:56.830000-05:00",  
  "lastUpdatedAt": "2022-12-02T16:02:45.857000-05:00"  
}
```

For more information, see [UpdateSuiteDefinition](#) in the *AWS IoT API Reference*.

- For API details, see [UpdateSuiteDefinition](#) in *AWS CLI Command Reference*.

AWS IoT data examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS IoT data.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

delete-thing-shadow

The following code example shows how to use `delete-thing-shadow`.

AWS CLI

To delete a device's shadow document

The following `delete-thing-shadow` example deletes the entire shadow document for the device named `MyRPi`.

```
aws iot-data delete-thing-shadow \  
  --thing-name MyRPi \  
  "output.txt"
```

The command produces no output on the display, but `output.txt` contains information that confirms the version and timestamp of the shadow document that you deleted.

```
{"version":2,"timestamp":1560270384}
```

For more information, see [Using Shadows](#) in the *AWS IoT Developers Guide*.

- For API details, see [DeleteThingShadow](#) in *AWS CLI Command Reference*.

get-thing-shadow

The following code example shows how to use `get-thing-shadow`.

AWS CLI

To get a thing shadow document

The following `get-thing-shadow` example gets the thing shadow document for the specified IoT thing.

```
aws iot-data get-thing-shadow \  
  --thing-name MyRPi
```

```
--thing-name MyRPi \  
output.txt
```

The command produces no output on the display, but the following shows the contents of `output.txt`:

```
{  
  "state":{  
    "reported":{  
      "moisture":"low"  
    }  
  },  
  "metadata":{  
    "reported":{  
      "moisture":{  
        "timestamp":1560269319  
      }  
    }  
  },  
  "version":1,"timestamp":1560269405  
}
```

For more information, see [Device Shadow Service Data Flow](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetThingShadow](#) in *AWS CLI Command Reference*.

update-thing-shadow

The following code example shows how to use `update-thing-shadow`.

AWS CLI

To update a thing shadow

The following `update-thing-shadow` example modifies the current state of the device shadow for the specified thing and saves it to the file `output.txt`.

```
aws iot-data update-thing-shadow \  
  --thing-name MyRPi \  
  --payload '{"state":{"reported":{"moisture":"okay"}}}' \  
  "output.txt"
```

The command produces no output on the display, but the following shows the contents of `output.txt`:

```
{
  "state": {
    "reported": {
      "moisture": "okay"
    }
  },
  "metadata": {
    "reported": {
      "moisture": {
        "timestamp": 1560270036
      }
    }
  },
  "version": 2,
  "timestamp": 1560270036
}
```

For more information, see [Device Shadow Service Data Flow](#) in the *AWS IoT Developers Guide*.

- For API details, see [UpdateThingShadow](#) in *AWS CLI Command Reference*.

AWS IoT Events examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS IoT Events.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

batch-put-message

The following code example shows how to use `batch-put-message`.

AWS CLI

To send messages (inputs) to AWS IoT Events

The following `batch-put-message` example sends a set of messages to the AWS IoT Events system. Each message payload is transformed into the input you specify (`inputName`) and ingested into any detectors that monitor that input. If multiple messages are sent, the order in which the messages are processed isn't guaranteed. To guarantee ordering, you must send messages one at a time and wait for a successful response.

```
aws iotevents-data batch-put-message \  
  --cli-input-json file://highPressureMessage.json
```

Contents of `highPressureMessage.json`:

```
{  
  "messages": [  
    {  
      "messageId": "00001",  
      "inputName": "PressureInput",  
      "payload": "{\"motorid\": \"Fulton-A32\", \"sensorData\": {\"pressure\":  
80, \"temperature\": 39} }"  
    }  
  ]  
}
```

Output:

```
{  
  "BatchPutMessageErrorEntries": []  
}
```

For more information, see [BatchPutMessage](#) in the *AWS IoT Events API Reference*.

- For API details, see [BatchPutMessage](#) in *AWS CLI Command Reference*.

batch-update-detector

The following code example shows how to use batch-update-detector.

AWS CLI

To update a detector (instance)

The following batch-update-detector example updates the state, variable values, and timer settings of one or more detectors (instances) of a specified detector model.

```
aws iotevents-data batch-update-detector \  
  --cli-input-json file://budFulton-A32.json
```

Contents of budFulton-A32.json:

```
{  
  "detectors": [  
    {  
      "messageId": "00001",  
      "detectorModelName": "motorDetectorModel",  
      "keyValue": "Fulton-A32",  
      "state": {  
        "stateName": "Normal",  
        "variables": [  
          {  
            "name": "pressureThresholdBreach",  
            "value": "0"  
          }  
        ],  
        "timers": [  
        ]  
      }  
    }  
  ]  
}
```

Output:

```
{  
  "batchUpdateDetectorErrorEntries": []  
}
```

For more information, see [BatchUpdateDetector](#) in the *AWS IoT Events API Reference*.

- For API details, see [BatchUpdateDetector](#) in *AWS CLI Command Reference*.

create-detector-model

The following code example shows how to use `create-detector-model`.

AWS CLI

To create a detector model

The following `create-detector-model` example creates a detector model with its configuration specified by a parameter file.

```
aws iotevents create-detector-model \
  --cli-input-json file://motorDetectorModel.json
```

Contents of `motorDetectorModel.json`:

```
{
  "detectorModelName": "motorDetectorModel",
  "detectorModelDefinition": {
    "states": [
      {
        "stateName": "Normal",
        "onEnter": {
          "events": [
            {
              "eventName": "init",
              "condition": "true",
              "actions": [
                {
                  "setVariable": {
                    "variableName": "pressureThresholdBreached",
                    "value": "0"
                  }
                }
              ]
            }
          ]
        }
      ]
    ],
    "onInput": {
```



```

        "transitionEvents": [
            {
                "eventName": "Overpressurized",
                "condition": "$input.PressureInput.sensorData.pressure
> 70",
                "actions": [
                    {
                        "setVariable": {
                            "variableName": "pressureThresholdBreach",
                            "value":
"$variable.pressureThresholdBreach + 3"
                        }
                    }
                ],
                "nextState": "Dangerous"
            }
        ],
    },
    {
        "stateName": "Dangerous",
        "onEnter": {
            "events": [
                {
                    "eventName": "Pressure Threshold Breached",
                    "condition": "$variable.pressureThresholdBreach >
1",
                    "actions": [
                        {
                            "sns": {
                                "targetArn": "arn:aws:sns:us-
east-1:123456789012:underPressureAction"
                            }
                        }
                    ]
                }
            ]
        },
        "onInput": {
            "events": [
                {
                    "eventName": "Overpressurized",
                    "condition": "$input.PressureInput.sensorData.pressure
> 70",

```

```

        "actions": [
            {
                "setVariable": {
                    "variableName": "pressureThresholdBreached",
                    "value": "3"
                }
            }
        ],
    },
    {
        "eventName": "Pressure Okay",
        "condition": "$input.PressureInput.sensorData.pressure
<= 70",
        "actions": [
            {
                "setVariable": {
                    "variableName": "pressureThresholdBreached",
                    "value":
"$variable.pressureThresholdBreached - 1"
                }
            }
        ]
    }
],
"transitionEvents": [
    {
        "eventName": "BackToNormal",
        "condition": "$input.PressureInput.sensorData.pressure
<= 70 && $variable.pressureThresholdBreached <= 1",
        "nextState": "Normal"
    }
],
"onExit": {
    "events": [
        {
            "eventName": "Normal Pressure Restored",
            "condition": "true",
            "actions": [
                {
                    "sns": {
                        "targetArn": "arn:aws:sns:us-
east-1:123456789012:pressureClearedAction"
                    }
                }
            ]
        }
    ]
}

```

```

    ],
    "initialStateName": "Normal"
  },
  "key": "motorid",
  "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole"
}

```

Output:

```

{
  "detectorModelConfiguration": {
    "status": "ACTIVATING",
    "lastUpdateTime": 1560796816.077,
    "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",
    "creationTime": 1560796816.077,
    "detectorModelArn": "arn:aws:iotevents:us-west-2:123456789012:detectorModel/
motorDetectorModel",
    "key": "motorid",
    "detectorModelName": "motorDetectorModel",
    "detectorModelVersion": "1"
  }
}

```

For more information, see [CreateDetectorModel](#) in the *AWS IoT Events API Reference*.

- For API details, see [CreateDetectorModel](#) in *AWS CLI Command Reference*.

create-input

The following code example shows how to use create-input.

AWS CLI

To create an input

The following create-input example creates an input.

```
aws iotevents create-input \  
  --cli-input-json file://pressureInput.json
```

Contents of `pressureInput.json`:

```
{  
  "inputName": "PressureInput",  
  "inputDescription": "Pressure readings from a motor",  
  "inputDefinition": {  
    "attributes": [  
      { "jsonPath": "sensorData.pressure" },  
      { "jsonPath": "motorid" }  
    ]  
  }  
}
```

Output:

```
{  
  "inputConfiguration": {  
    "status": "ACTIVE",  
    "inputArn": "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput",  
    "lastUpdateTime": 1560795312.542,  
    "creationTime": 1560795312.542,  
    "inputName": "PressureInput",  
    "inputDescription": "Pressure readings from a motor"  
  }  
}
```

For more information, see [CreateInput](#) in the *AWS IoT Events API Reference*.

- For API details, see [CreateInput](#) in *AWS CLI Command Reference*.

delete-detector-model

The following code example shows how to use `delete-detector-model`.

AWS CLI

To delete a detector model

The following `delete-detector-model` example deletes the specified detector model. Any active instances of the detector model are also deleted.

```
aws iotevents delete-detector-model \  
  --detector-model-name motorDetectorModel
```

This command produces no output.

For more information, see [DeleteDetectorModel](#) in the *AWS IoT Events API Reference*.

- For API details, see [DeleteDetectorModel](#) in *AWS CLI Command Reference*.

delete-input

The following code example shows how to use `delete-input`.

AWS CLI

To delete an input

The following `delete-input` example deletes the specified input.

```
aws iotevents delete-input \  
  --input-name PressureInput
```

This command produces no output.

For more information, see [DeleteInput](#) in the *AWS IoT Events API Reference*.

- For API details, see [DeleteInput](#) in *AWS CLI Command Reference*.

describe-detector-model

The following code example shows how to use `describe-detector-model`.

AWS CLI

To get information about a detector model

The following `describe-detector-model` example displays details for the specified detector model. Because the `version` parameter is not specified, information about the latest version is returned.

```
aws iotevents describe-detector-model \
  --detector-model-name motorDetectorModel
```

Output:

```
{
  "detectorModel": {
    "detectorModelConfiguration": {
      "status": "ACTIVE",
      "lastUpdateTime": 1560796816.077,
      "roleArn": "arn:aws:iam:123456789012:role/IoTEventsRole",
      "creationTime": 1560796816.077,
      "detectorModelArn": "arn:aws:iotevents:us-
west-2:123456789012:detectorModel/motorDetectorModel",
      "key": "motorid",
      "detectorModelName": "motorDetectorModel",
      "detectorModelVersion": "1"
    },
    "detectorModelDefinition": {
      "states": [
        {
          "onInput": {
            "transitionEvents": [
              {
                "eventName": "Overpressurized",
                "actions": [
                  {
                    "setVariable": {
                      "variableName":
"pressureThresholdBreached",
                      "value":
"$variable.pressureThresholdBreached + 3"
                    }
                  ]
                },
                "condition":
"$input.PressureInput.sensorData.pressure > 70",
                "nextState": "Dangerous"
              }
            ],
            "events": []
          },
          "stateName": "Normal",

```

```

        "onEnter": {
            "events": [
                {
                    "eventName": "init",
                    "actions": [
                        {
                            "setVariable": {
                                "variableName":
"pressureThresholdBreach",
                                "value": "0"
                            }
                        }
                    ],
                    "condition": "true"
                }
            ]
        },
        "onExit": {
            "events": []
        }
    },
    {
        "onInput": {
            "transitionEvents": [
                {
                    "eventName": "BackToNormal",
                    "actions": [],
                    "condition":
"$input.PressureInput.sensorData.pressure <= 70 &&
$variable.pressureThresholdBreach <= 1",
                    "nextState": "Normal"
                }
            ],
            "events": [
                {
                    "eventName": "Overpressurized",
                    "actions": [
                        {
                            "setVariable": {
                                "variableName":
"pressureThresholdBreach",
                                "value": "3"
                            }
                        }
                    ]
                }
            ]
        }
    }
}

```

```

        ],
        "condition":
"$input.PressureInput.sensorData.pressure > 70"
    },
    {
        "eventName": "Pressure Okay",
        "actions": [
            {
                "setVariable": {
                    "variableName":
"pressureThresholdBreached",
                    "value":
"$variable.pressureThresholdBreached - 1"
                }
            }
        ],
        "condition":
"$input.PressureInput.sensorData.pressure <= 70"
    }
]
},
"stateName": "Dangerous",
"onEnter": {
    "events": [
        {
            "eventName": "Pressure Threshold Breached",
            "actions": [
                {
                    "sns": {
                        "targetArn": "arn:aws:sns:us-
east-1:123456789012:underPressureAction"
                    }
                }
            ]
        },
        {
            "condition": "$variable.pressureThresholdBreached >
1"
        }
    ]
},
"onExit": {
    "events": [
        {
            "eventName": "Normal Pressure Restored",
            "actions": [

```



```

        {
            "sns": {
                "targetArn": "arn:aws:sns:us-
east-1:123456789012:pressureClearedAction"
            }
        },
        "condition": "true"
    ]
}
    ]
}
    ],
    "initialStateName": "Normal"
}
}
}
}

```

For more information, see [DescribeDetectorModel](#) in the *AWS IoT Events API Reference*.

- For API details, see [DescribeDetectorModel](#) in *AWS CLI Command Reference*.

describe-detector

The following code example shows how to use describe-detector.

AWS CLI

To get information about a detector (instance).

The following describe-detector example displays details for the specified detector (instance).

```

aws iotevents-data describe-detector \
  --detector-model-name motorDetectorModel \
  --key-value "Fulton-A32"

```

Output:

```

{
  "detector": {
    "lastUpdateTime": 1560797852.776,
    "creationTime": 1560797852.775,

```

```
    "state": {
      "variables": [
        {
          "name": "pressureThresholdBreached",
          "value": "3"
        }
      ],
      "stateName": "Dangerous",
      "timers": []
    },
    "keyValue": "Fulton-A32",
    "detectorModelName": "motorDetectorModel",
    "detectorModelVersion": "1"
  }
}
```

For more information, see [DescribeDetector](#) in the *AWS IoT Events API Reference*.

- For API details, see [DescribeDetector](#) in *AWS CLI Command Reference*.

describe-input

The following code example shows how to use `describe-input`.

AWS CLI

To get information about an input

The following `describe-input` example displays details for the specified input.

```
aws iotevents describe-input \
  --input-name PressureInput
```

Output:

```
{
  "input": {
    "inputConfiguration": {
      "status": "ACTIVE",
      "inputArn": "arn:aws:iotevents:us-west-2:123456789012:input/
PressureInput",
      "lastUpdateTime": 1560795312.542,
      "creationTime": 1560795312.542,

```

```
        "inputName": "PressureInput",
        "inputDescription": "Pressure readings from a motor"
    },
    "inputDefinition": {
        "attributes": [
            {
                "jsonPath": "sensorData.pressure"
            },
            {
                "jsonPath": "motorid"
            }
        ]
    }
}
```

For more information, see [DescribeInput](#) in the *AWS IoT Events API Reference*.

- For API details, see [DescribeInput](#) in *AWS CLI Command Reference*.

describe-logging-options

The following code example shows how to use `describe-logging-options`.

AWS CLI

To get information about logging settings

The following `describe-logging-options` example retrieves the current settings of the AWS IoT Events logging options.

```
aws iotevents describe-logging-options
```

Output:

```
{
  "loggingOptions": {
    "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",
    "enabled": false,
    "level": "ERROR"
  }
}
```

For more information, see [DescribeLoggingOptions](#) in the *AWS IoT Events API Reference*.

- For API details, see [DescribeLoggingOptions](#) in *AWS CLI Command Reference*.

list-detector-model-versions

The following code example shows how to use `list-detector-model-versions`.

AWS CLI

To get information about versions of a detector model

The following `list-detector-model-versions` example Lists all the versions of a detector model. Only the metadata associated with each detector model version is returned.

```
aws iotevents list-detector-model-versions \  
  --detector-model-name motorDetectorModel
```

Output:

```
{  
  "detectorModelVersionSummaries": [  
    {  
      "status": "ACTIVE",  
      "lastUpdateTime": 1560796816.077,  
      "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",  
      "creationTime": 1560796816.077,  
      "detectorModelArn": "arn:aws:iotevents:us-  
west-2:123456789012:detectorModel/motorDetectorModel",  
      "detectorModelName": "motorDetectorModel",  
      "detectorModelVersion": "1"  
    }  
  ]  
}
```

For more information, see [ListDetectorModelVersions](#) in the *AWS IoT Events API Reference*.

- For API details, see [ListDetectorModelVersions](#) in *AWS CLI Command Reference*.

list-detector-models

The following code example shows how to use `list-detector-models`.

AWS CLI

To get a list of your detector models

The following `list-detector-models` example Lists the detector models you have created. Only the metadata associated with each detector model is returned.

```
aws iotevents list-detector-models
```

Output:

```
{
  "detectorModelSummaries": [
    {
      "detectorModelName": "motorDetectorModel",
      "creationTime": 1552072424.212
      "detectorModelDescription": "Detect overpressure in a motor."
    }
  ]
}
```

For more information, see [ListDetectorModels](#) in the *AWS IoT Events API Reference*.

- For API details, see [ListDetectorModels](#) in *AWS CLI Command Reference*.

list-detectors

The following code example shows how to use `list-detectors`.

AWS CLI

To get a list of detectors for a detector model

The following `list-detectors` example lists the detectors (the instances of a detector model) in your account.

```
aws iotevents-data list-detectors \
  --detector-model-name motorDetectorModel
```

Output:

```
{
```

```
"detectorSummaries": [
  {
    "lastUpdateTime": 1558129925.2,
    "creationTime": 1552073155.527,
    "state": {
      "stateName": "Normal"
    },
    "keyValue": "Fulton-A32",
    "detectorModelName": "motorDetectorModel",
    "detectorModelVersion": "1"
  }
]
```

For more information, see [ListDetectors](#) in the *AWS IoT Events API Reference*.

- For API details, see [ListDetectors](#) in *AWS CLI Command Reference*.

list-inputs

The following code example shows how to use `list-inputs`.

AWS CLI

To list inputs

The following `list-inputs` example lists the inputs you have created in your account.

```
aws iotevents list-inputs
```

This command produces no output. Output:

```
{
  {
    "status": "ACTIVE",
    "inputArn": "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput",
    "lastUpdateTime": 1551742986.768,
    "creationTime": 1551742986.768,
    "inputName": "PressureInput",
    "inputDescription": "Pressure readings from a motor"
  }
}
```

For more information, see [ListInputs](#) in the *AWS IoT Events API Reference*.

- For API details, see [ListInputs](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags assigned to a resource.

The following `list-tags-for-resource` example lists the tag key names and values you have assigned to the resource.

```
aws iotevents list-tags-for-resource \  
  --resource-arn "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput"
```

Output:

```
{  
  "tags": [  
    {  
      "value": "motor",  
      "key": "deviceType"  
    }  
  ]  
}
```

For more information, see [ListTagsForResource](#) in the *AWS IoT Events API Reference*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

put-logging-options

The following code example shows how to use `put-logging-options`.

AWS CLI

To set logging options

The following `put-logging-options` example sets or updates the AWS IoT Events logging options. If you update the value of any `loggingOptions`` field, it can take up to

one minute for the change to take effect. Also, if you change the policy attached to the role you specified in the `roleArn` field (for example, to correct an invalid policy) it can take up to five minutes for that change to take effect.

```
aws iotevents put-logging-options \  
  --cli-input-json file://logging-options.json
```

Contents of `logging-options.json`:

```
{  
  "loggingOptions": {  
    "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",  
    "level": "DEBUG",  
    "enabled": true,  
    "detectorDebugOptions": [  
      {  
        "detectorModelName": "motorDetectorModel",  
        "keyValue": "Fulton-A32"  
      }  
    ]  
  }  
}
```

This command produces no output.

For more information, see [PutLoggingOptions](#) in the *AWS IoT Events API Reference*.

- For API details, see [PutLoggingOptions](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add tags to a resource

The following `tag-resource` example adds or modifies (if key `deviceType` already exists) the tag attached the specified resource.

```
aws iotevents tag-resource \  
  --cli-input-json file://pressureInput.tag.json
```


Contents of `pressureInput.tag.json`:

```
{
  "resourceArn": "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput",
  "tags": [
    {
      "key": "deviceType",
      "value": "motor"
    }
  ]
}
```

This command produces no output.

For more information, see [TagResource](#) in the *AWS IoT Events API Reference*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from a resource

The following `untag-resource` example removes the tag with the specified key name from the specified resource.

```
aws iotevents untag-resource \
  --resource-arn arn:aws:iotevents:us-west-2:123456789012:input/PressureInput \
  --tagkeys deviceType
```

This command produces no output.

For more information, see [UntagResource](#) in the *AWS IoT Events API Reference*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-detector-model

The following code example shows how to use `update-detector-model`.

AWS CLI

To update a detector model

The following `update-detector-model` example updates the specified detector model. Detectors (instances) spawned by the previous version are deleted and then re-created as new inputs arrive.

```
aws iotevents update-detector-model \  
  --cli-input-json file://motorDetectorModel.update.json
```

Contents of `motorDetectorModel.update.json`:

```
{  
  "detectorModelName": "motorDetectorModel",  
  "detectorModelDefinition": {  
    "states": [  
      {  
        "stateName": "Normal",  
        "onEnter": {  
          "events": [  
            {  
              "eventName": "init",  
              "condition": "true",  
              "actions": [  
                {  
                  "setVariable": {  
                    "variableName": "pressureThresholdBreach",  
                    "value": "0"  
                  }  
                }  
              ]  
            }  
          ]  
        },  
        "onInput": {  
          "transitionEvents": [  
            {  
              "eventName": "Overpressurized",  
              "condition": "$input.PressureInput.sensorData.pressure >  
70",  
              "actions": [  
                {
```

```

        "setVariable": {
            "variableName": "pressureThresholdBreached",
            "value":
"$variable.pressureThresholdBreached + 3"
        }
    ],
    "nextState": "Dangerous"
}
]
}
},
{
    "stateName": "Dangerous",
    "onEnter": {
        "events": [
            {
                "eventName": "Pressure Threshold Breached",
                "condition": "$variable.pressureThresholdBreached > 1",
                "actions": [
                    {
                        "sns": {
                            "targetArn": "arn:aws:sns:us-
east-1:123456789012:underPressureAction"
                        }
                    }
                ]
            }
        ]
    },
    "onInput": {
        "events": [
            {
                "eventName": "Overpressurized",
                "condition": "$input.PressureInput.sensorData.pressure >
70",
                "actions": [
                    {
                        "setVariable": {
                            "variableName": "pressureThresholdBreached",
                            "value": "3"
                        }
                    }
                ]
            }
        ]
    }
}
]

```

```

        },
        {
            "eventName": "Pressure Okay",
            "condition": "$input.PressureInput.sensorData.pressure
<= 70",
            "actions": [
                {
                    "setVariable": {
                        "variableName": "pressureThresholdBreached",
                        "value":
"$variable.pressureThresholdBreached - 1"
                    }
                }
            ]
        }
    ],
    "transitionEvents": [
        {
            "eventName": "BackToNormal",
            "condition": "$input.PressureInput.sensorData.pressure
<= 70 && $variable.pressureThresholdBreached <= 1",
            "nextState": "Normal"
        }
    ]
},
"onExit": {
    "events": [
        {
            "eventName": "Normal Pressure Restored",
            "condition": "true",
            "actions": [
                {
                    "sns": {
                        "targetArn": "arn:aws:sns:us-
east-1:123456789012:pressureClearedAction"
                    }
                }
            ]
        }
    ]
}
},
],
"initialStateName": "Normal"

```

```
  },
  "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole"
}
```

Output:

```
{
  "detectorModelConfiguration": {
    "status": "ACTIVATING",
    "lastUpdateTime": 1560799387.719,
    "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",
    "creationTime": 1560799387.719,
    "detectorModelArn": "arn:aws:iotevents:us-west-2:123456789012:detectorModel/
motorDetectorModel",
    "key": "motorid",
    "detectorModelName": "motorDetectorModel",
    "detectorModelVersion": "2"
  }
}
```

For more information, see [UpdateDetectorModel](#) in the *AWS IoT Events API Reference*.

- For API details, see [UpdateDetectorModel](#) in *AWS CLI Command Reference*.

update-input

The following code example shows how to use `update-input`.

AWS CLI

To update an input

The following `update-input` example updates the specified input with a new description and definition.

```
aws iotevents update-input \
  --cli-input-json file://pressureInput.json
```

Contents of `pressureInput.json`:

```
{
  "inputName": "PressureInput",
```

```
"inputDescription": "Pressure readings from a motor",
"inputDefinition": {
  "attributes": [
    { "jsonPath": "sensorData.pressure" },
    { "jsonPath": "motorid" }
  ]
}
```

Output:

```
{
  "inputConfiguration": {
    "status": "ACTIVE",
    "inputArn": "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput",
    "lastUpdateTime": 1560795976.458,
    "creationTime": 1560795312.542,
    "inputName": "PressureInput",
    "inputDescription": "Pressure readings from a motor"
  }
}
```

For more information, see [UpdateInput](#) in the *AWS IoT Events API Reference*.

- For API details, see [UpdateInput](#) in *AWS CLI Command Reference*.

AWS IoT Events-Data examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS IoT Events-Data.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

batch-put-message

The following code example shows how to use `batch-put-message`.

AWS CLI

To send messages (inputs) to AWS IoT Events

The following `batch-put-message` example sends a set of messages to the AWS IoT Events system. Each message payload is transformed into the input you specify (`inputName`) and ingested into any detectors that monitor that input. If multiple messages are sent, the order in which the messages are processed isn't guaranteed. To guarantee ordering, you must send messages one at a time and wait for a successful response.

```
aws iotevents-data batch-put-message \  
  --cli-binary-format raw-in-base64-out \  
  --cli-input-json file://highPressureMessage.json
```

Contents of `highPressureMessage.json`:

```
{  
  "messages": [  
    {  
      "messageId": "00001",  
      "inputName": "PressureInput",  
      "payload": "{\"motorid\": \"Fulton-A32\", \"sensorData\": {\"pressure\":  
80, \"temperature\": 39} }"  
    }  
  ]  
}
```

Output:

```
{  
  "BatchPutMessageErrorEntries": []  
}
```

For more information, see [BatchPutMessage](#) in the *AWS IoT Events Developer Guide**.

- For API details, see [BatchPutMessage](#) in *AWS CLI Command Reference*.

batch-update-detector

The following code example shows how to use batch-update-detector.

AWS CLI

To update a detector (instance)

The following batch-update-detector example updates the state, variable values, and timer settings of one or more detectors (instances) of a specified detector model.

```
aws iotevents-data batch-update-detector \  
  --cli-input-json file://budFulton-A32.json
```

Contents of budFulton-A32.json:

```
{  
  "detectors": [  
    {  
      "messageId": "00001",  
      "detectorModelName": "motorDetectorModel",  
      "keyValue": "Fulton-A32",  
      "state": {  
        "stateName": "Normal",  
        "variables": [  
          {  
            "name": "pressureThresholdBreach",  
            "value": "0"  
          }  
        ],  
        "timers": [  
        ]  
      }  
    }  
  ]  
}
```

Output:


```
{
  "batchUpdateDetectorErrorEntries": []
}
```

For more information, see [BatchUpdateDetector](#) in the *AWS IoT Events Developer Guide*.*

- For API details, see [BatchUpdateDetector](#) in *AWS CLI Command Reference*.

create-detector-model

The following code example shows how to use `create-detector-model`.

AWS CLI

To create a detector model

The following `create-detector-model` example creates a detector model.

```
aws iotevents create-detector-model \
  --cli-input-json file://motorDetectorModel.json
```

Contents of `motorDetectorModel.json`:

```
{
  "detectorModelName": "motorDetectorModel",
  "detectorModelDefinition": {
    "states": [
      {
        "stateName": "Normal",
        "onEnter": {
          "events": [
            {
              "eventName": "init",
              "condition": "true",
              "actions": [
                {
                  "setVariable": {
                    "variableName": "pressureThresholdBreach",
                    "value": "0"
                  }
                }
              ]
            }
          ]
        }
      }
    ]
  }
}
```

```

    ]
  },
  "onInput": {
    "transitionEvents": [
      {
        "eventName": "Overpressurized",
        "condition": "$input.PressureInput.sensorData.pressure
&gt; 70",
        "actions": [
          {
            "setVariable": {
              "variableName": "pressureThresholdBreach",
              "value":
"$variable.pressureThresholdBreach + 3"
            }
          }
        ],
        "nextState": "Dangerous"
      }
    ]
  }
},
{
  "stateName": "Dangerous",
  "onEnter": {
    "events": [
      {
        "eventName": "Pressure Threshold Breached",
        "condition": "$variable.pressureThresholdBreach &gt;
1",
        "actions": [
          {
            "sns": {
              "targetArn": "arn:aws:sns:us-
east-1:123456789012:underPressureAction"
            }
          }
        ]
      }
    ]
  },
  "onInput": {
    "events": [
      {

```

```

        "eventName": "Overpressurized",
        "condition": "$input.PressureInput.sensorData.pressure
&gt; 70",
        "actions": [
            {
                "setVariable": {
                    "variableName": "pressureThresholdBreached",
                    "value": "3"
                }
            }
        ]
    },
    {
        "eventName": "Pressure Okay",
        "condition": "$input.PressureInput.sensorData.pressure
&lt;= 70",
        "actions": [
            {
                "setVariable": {
                    "variableName": "pressureThresholdBreached",
                    "value":
"$variable.pressureThresholdBreached - 1"
                }
            }
        ]
    }
],
"transitionEvents": [
    {
        "eventName": "BackToNormal",
        "condition": "$input.PressureInput.sensorData.pressure
&lt;= 70 &amp; $variable.pressureThresholdBreached &lt;= 1",
        "nextState": "Normal"
    }
]
},
"onExit": {
    "events": [
        {
            "eventName": "Normal Pressure Restored",
            "condition": "true",
            "actions": [
                {
                    "sns": {

```

```

        "targetArn": "arn:aws:sns:us-
east-1:123456789012:pressureClearedAction"
    }
}
]
}
]
}
],
"initialStateName": "Normal"
},
"key": "motorid",
"roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole"
}

```

Output:

```

{
  "detectorModelConfiguration": {
    "status": "ACTIVATING",
    "lastUpdateTime": 1560796816.077,
    "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",
    "creationTime": 1560796816.077,
    "detectorModelArn": "arn:aws:iotevents:us-west-2:123456789012:detectorModel/
motorDetectorModel",
    "key": "motorid",
    "detectorModelName": "motorDetectorModel",
    "detectorModelVersion": "1"
  }
}

```

For more information, see [CreateDetectorModel](#) in the *AWS IoT Events Developer Guide**.

- For API details, see [CreateDetectorModel](#) in *AWS CLI Command Reference*.

create-input

The following code example shows how to use create-input.

AWS CLI

To create an input

The following `create-input` example creates an input.

```
aws iotevents create-input \  
  --cli-input-json file://pressureInput.json
```

Contents of `pressureInput.json`:

```
{  
  "inputName": "PressureInput",  
  "inputDescription": "Pressure readings from a motor",  
  "inputDefinition": {  
    "attributes": [  
      { "jsonPath": "sensorData.pressure" },  
      { "jsonPath": "motorid" }  
    ]  
  }  
}
```

Output:

```
{  
  "inputConfiguration": {  
    "status": "ACTIVE",  
    "inputArn": "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput",  
    "lastUpdateTime": 1560795312.542,  
    "creationTime": 1560795312.542,  
    "inputName": "PressureInput",  
    "inputDescription": "Pressure readings from a motor"  
  }  
}
```

For more information, see [CreateInput](#) in the *AWS IoT Events Developer Guide*.*.

- For API details, see [CreateInput](#) in *AWS CLI Command Reference*.

delete-detector-model

The following code example shows how to use `delete-detector-model`.

AWS CLI

To delete a detector model

The following `delete-detector-model` example deletes a detector model. Any active instances of the detector model are also deleted.

```
aws iotevents delete-detector-model \  
  --detector-model-name motorDetectorModel*
```

This command produces no output.

For more information, see [DeleteDetectorModel](#) in the *AWS IoT Events Developer Guide**.

- For API details, see [DeleteDetectorModel](#) in *AWS CLI Command Reference*.

delete-input

The following code example shows how to use `delete-input`.

AWS CLI

To delete an input

The following `delete-input` example deletes an input.

```
aws iotevents delete-input \  
  --input-name PressureInput
```

This command produces no output.

For more information, see [DeleteInput](#) in the *AWS IoT Events Developer Guide**.

- For API details, see [DeleteInput](#) in *AWS CLI Command Reference*.

describe-detector-model

The following code example shows how to use `describe-detector-model`.

AWS CLI

To get information about a detector model

The following `describe-detector-model` example describes a detector model. If the `version` parameter is not specified, the command returns information about the latest version.

```
aws iotevents describe-detector-model \
  --detector-model-name motorDetectorModel
```

Output:

```
{
  "detectorModel": {
    "detectorModelConfiguration": {
      "status": "ACTIVE",
      "lastUpdateTime": 1560796816.077,
      "roleArn": "arn:aws:iam:123456789012:role/IoTEventsRole",
      "creationTime": 1560796816.077,
      "detectorModelArn": "arn:aws:iotevents:us-
west-2:123456789012:detectorModel/motorDetectorModel",
      "key": "motorid",
      "detectorModelName": "motorDetectorModel",
      "detectorModelVersion": "1"
    },
    "detectorModelDefinition": {
      "states": [
        {
          "onInput": {
            "transitionEvents": [
              {
                "eventName": "Overpressurized",
                "actions": [
                  {
                    "setVariable": {
                      "variableName":
"pressureThresholdBreach",
                      "value":
"$variable.pressureThresholdBreach + 3"
                    }
                  ]
                },
                "condition":
"$input.PressureInput.sensorData.pressure > 70",
                "nextState": "Dangerous"
              }
            ],
            "events": []
          },
          "stateName": "Normal",

```

```

        "onEnter": {
            "events": [
                {
                    "eventName": "init",
                    "actions": [
                        {
                            "setVariable": {
                                "variableName":
"pressureThresholdBreached",
                                "value": "0"
                            }
                        }
                    ],
                    "condition": "true"
                }
            ]
        },
        "onExit": {
            "events": []
        }
    },
    {
        "onInput": {
            "transitionEvents": [
                {
                    "eventName": "BackToNormal",
                    "actions": [],
                    "condition":
"$input.PressureInput.sensorData.pressure <= 70 &&
$variable.pressureThresholdBreached <= 1",
                    "nextState": "Normal"
                }
            ],
            "events": [
                {
                    "eventName": "Overpressurized",
                    "actions": [
                        {
                            "setVariable": {
                                "variableName":
"pressureThresholdBreached",
                                "value": "3"
                            }
                        }
                    ]
                }
            ]
        }
    }
}

```



```

        ],
        "condition":
"$input.PressureInput.sensorData.pressure > 70"
    },
    {
        "eventName": "Pressure Okay",
        "actions": [
            {
                "setVariable": {
                    "variableName":
"pressureThresholdBreached",
                    "value":
"$variable.pressureThresholdBreached - 1"
                }
            }
        ],
        "condition":
"$input.PressureInput.sensorData.pressure <= 70"
    }
]
},
"stateName": "Dangerous",
"onEnter": {
    "events": [
        {
            "eventName": "Pressure Threshold Breached",
            "actions": [
                {
                    "sns": {
                        "targetArn": "arn:aws:sns:us-
east-1:123456789012:underPressureAction"
                    }
                }
            ]
        },
        {
            "condition": "$variable.pressureThresholdBreached >
1"
        }
    ]
},
"onExit": {
    "events": [
        {
            "eventName": "Normal Pressure Restored",
            "actions": [

```

```

        {
            "sns": {
                "targetArn": "arn:aws:sns:us-
east-1:123456789012:pressureClearedAction"
            }
        },
        "condition": "true"
    ]
}
    ]
}
    ],
    "initialStateName": "Normal"
}
}
}

```

For more information, see [DescribeDetectorModel](#) in the *AWS IoT Events Developer Guide**.

- For API details, see [DescribeDetectorModel](#) in *AWS CLI Command Reference*.

describe-detector

The following code example shows how to use describe-detector.

AWS CLI

To get information about a detector (instance)

The following describe-detector example returns information about the specified detector (instance).

```

aws iotevents-data describe-detector \
  --detector-model-name motorDetectorModel \
  --key-value "Fulton-A32"

```

Output:

```

{
  "detector": {
    "lastUpdateTime": 1560797852.776,
    "creationTime": 1560797852.775,

```

```
    "state": {
      "variables": [
        {
          "name": "pressureThresholdBreached",
          "value": "3"
        }
      ],
      "stateName": "Dangerous",
      "timers": []
    },
    "keyValue": "Fulton-A32",
    "detectorModelName": "motorDetectorModel",
    "detectorModelVersion": "1"
  }
}
```

For more information, see [DescribeDetector](#) in the *AWS IoT Events Developer Guide*.*

- For API details, see [DescribeDetector](#) in *AWS CLI Command Reference*.

describe-input

The following code example shows how to use `describe-input`.

AWS CLI

To get information about an input

The following `describe-input` example retrieves the details of an input.

```
aws iotevents describe-input \
  --input-name PressureInput
```

Output:

```
{
  "input": {
    "inputConfiguration": {
      "status": "ACTIVE",
      "inputArn": "arn:aws:iotevents:us-west-2:123456789012:input/
PressureInput",
      "lastUpdateTime": 1560795312.542,
      "creationTime": 1560795312.542,

```

```
        "inputName": "PressureInput",
        "inputDescription": "Pressure readings from a motor"
    },
    "inputDefinition": {
        "attributes": [
            {
                "jsonPath": "sensorData.pressure"
            },
            {
                "jsonPath": "motorid"
            }
        ]
    }
}
```

For more information, see [DescribeInput](#) in the *AWS IoT Events Developer Guide*.*.

- For API details, see [DescribeInput](#) in *AWS CLI Command Reference*.

describe-logging-options

The following code example shows how to use `describe-logging-options`.

AWS CLI

To get information about logging settings

The following `describe-logging-options` example retrieves the current AWS IoT Events logging options.

```
aws iotevents describe-logging-options
```

Output:

```
{
  "loggingOptions": {
    "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",
    "enabled": false,
    "level": "ERROR"
  }
}
```

For more information, see [DescribeLoggingOptions](#) in the *AWS IoT Events Developer Guide**.

- For API details, see [DescribeLoggingOptions](#) in *AWS CLI Command Reference*.

list-detector-model-versions

The following code example shows how to use `list-detector-model-versions`.

AWS CLI

To get information about versions of a detector model

The following `list-detector-model-versions` example lists all the versions of a detector model. Only the metadata associated with each detector model version is returned.

```
aws iotevents list-detector-model-versions \  
  --detector-model-name motorDetectorModel
```

Output:

```
{  
  "detectorModelVersionSummaries": [  
    {  
      "status": "ACTIVE",  
      "lastUpdateTime": 1560796816.077,  
      "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",  
      "creationTime": 1560796816.077,  
      "detectorModelArn": "arn:aws:iotevents:us-  
west-2:123456789012:detectorModel/motorDetectorModel",  
      "detectorModelName": "motorDetectorModel",  
      "detectorModelVersion": "1"  
    }  
  ]  
}
```

For more information, see [ListDetectorModelVersions](#) in the *AWS IoT Events Developer Guide**.

- For API details, see [ListDetectorModelVersions](#) in *AWS CLI Command Reference*.

list-detector-models

The following code example shows how to use `list-detector-models`.

AWS CLI

To get a list of your detector models

The following `list-detector-models` example lists the detector models you have created. Only the metadata associated with each detector model is returned.

```
aws iotevents list-detector-models
```

Output:

```
{
  "detectorModelSummaries": [
    {
      "detectorModelName": "motorDetectorModel",
      "creationTime": 1552072424.212
      "detectorModelDescription": "Detect overpressure in a motor."
    }
  ]
}
```

For more information, see [ListDetectorModels](#) in the *AWS IoT Events Developer Guide*.*

- For API details, see [ListDetectorModels](#) in *AWS CLI Command Reference*.

list-detectors

The following code example shows how to use `list-detectors`.

AWS CLI

To get a list of detectors for a detector model

The following `list-detectors` example lists detectors (the instances of a detector model).

```
aws iotevents-data list-detectors \
  --detector-model-name motorDetectorModel
```

Output:

```
{
  "detectorSummaries": [
```

```
    {
      "lastUpdateTime": 1558129925.2,
      "creationTime": 1552073155.527,
      "state": {
        "stateName": "Normal"
      },
      "keyValue": "Fulton-A32",
      "detectorModelName": "motorDetectorModel",
      "detectorModelVersion": "1"
    }
  ]
}
```

For more information, see [ListDetectors](#) in the *AWS IoT Events Developer Guide**.

- For API details, see [ListDetectors](#) in *AWS CLI Command Reference*.

list-inputs

The following code example shows how to use `list-inputs`.

AWS CLI

To list inputs

The following `list-inputs` example lists the inputs that you've created.

```
aws iotevents list-inputs
```

Output:

```
{
  "status": "ACTIVE",
  "inputArn": "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput",
  "lastUpdateTime": 1551742986.768,
  "creationTime": 1551742986.768,
  "inputName": "PressureInput",
  "inputDescription": "Pressure readings from a motor"
}
```

For more information, see [ListInputs](#) in the *AWS IoT Events Developer Guide**.

- For API details, see [ListInputs](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags assigned to a resource

The following `list-tags-for-resource` example lists the tags (metadata) you have assigned to the resource.

```
aws iotevents list-tags-for-resource \  
  --resource-arn "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput"
```

Output:

```
{  
  "tags": [  
    {  
      "value": "motor",  
      "key": "deviceType"  
    }  
  ]  
}
```

For more information, see [ListTagsForResource](#) in the *AWS IoT Events Developer Guide**.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

put-logging-options

The following code example shows how to use `put-logging-options`.

AWS CLI

To set logging options

The following `list-tags-for-resource` example sets or updates the AWS IoT Events logging options. If you update the value of any `loggingOptions` field, it takes up to one minute for the change to take effect. Also, if you change the policy attached to the role you specified in the `roleArn` field (for example, to correct an invalid policy) it takes up to five minutes for that change to take effect.


```
aws iotevents put-logging-options \  
  --cli-input-json file://logging-options.json
```

Contents of logging-options.json:

```
{  
  "loggingOptions": {  
    "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",  
    "level": "DEBUG",  
    "enabled": true,  
    "detectorDebugOptions": [  
      {  
        "detectorModelName": "motorDetectorModel",  
        "keyValue": "Fulton-A32"  
      }  
    ]  
  }  
}
```

This command produces no output.

For more information, see [PutLoggingOptions](#) in the *AWS IoT Events Developer Guide**

- For API details, see [PutLoggingOptions](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To add tags to a resource

The following tag-resource example adds to or modifies the tags of the given resource. Tags are metadata that can be used to manage a resource.

```
aws iotevents tag-resource \  
  --cli-input-json file://pressureInput.tag.json
```

Contents of pressureInput.tag.json:

```
{
```

```
"resourceArn": "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput",
"tags": [
  {
    "key": "deviceType",
    "value": "motor"
  }
]
}
```

This command produces no output.

For more information, see [TagResource](#) in the *AWS IoT Events Developer Guide*.*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from a resource

The following `untag-resource` example removes the specified tags from the resource.

```
aws iotevents untag-resource \
  --cli-input-json file://pressureInput.untag.json
```

Contents of `pressureInput.untag.json`:

```
{
  "resourceArn": "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput",
  "tagKeys": [
    "deviceType"
  ]
}
```

This command produces no output.

For more information, see [UntagResource](#) in the *AWS IoT Events Developer Guide*.*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-detector-model

The following code example shows how to use `update-detector-model`.

AWS CLI

To update a detector model

The following `update-detector-model` example updates a detector model. Detectors (instances) spawned by the previous version are deleted and then re-created as new inputs arrive.

```
aws iotevents update-detector-model \  
  --cli-input-json file://motorDetectorModel.update.json
```

Contents of `motorDetectorModel.update.json`:

```
{  
  "detectorModelName": "motorDetectorModel",  
  "detectorModelDefinition": {  
    "states": [  
      {  
        "stateName": "Normal",  
        "onEnter": {  
          "events": [  
            {  
              "eventName": "init",  
              "condition": "true",  
              "actions": [  
                {  
                  "setVariable": {  
                    "variableName": "pressureThresholdBreach",  
                    "value": "0"  
                  }  
                }  
              ]  
            }  
          ]  
        },  
        "onInput": {  
          "transitionEvents": [  
            {  
              "eventName": "Overpressurized",
```

```
    "condition": "$input.PressureInput.sensorData.pressure > 70",
    "actions": [
      {
        "setVariable": {
          "variableName": "pressureThresholdBreachd",
          "value": "$variable.pressureThresholdBreachd + 3"
        }
      }
    ],
    "nextState": "Dangerous"
  }
]
},
{
  "stateName": "Dangerous",
  "onEnter": {
    "events": [
      {
        "eventName": "Pressure Threshold Breached",
        "condition": "$variable.pressureThresholdBreachd > 1",
        "actions": [
          {
            "sns": {
              "targetArn": "arn:aws:sns:us-
east-1:123456789012:underPressureAction"
            }
          }
        ]
      }
    ]
  },
  "onInput": {
    "events": [
      {
        "eventName": "Overpressurized",
        "condition": "$input.PressureInput.sensorData.pressure > 70",
        "actions": [
          {
            "setVariable": {
              "variableName": "pressureThresholdBreachd",
              "value": "3"
            }
          }
        ]
      }
    ]
  }
}
```

```
    ]
  },
  {
    "eventName": "Pressure Okay",
    "condition": "$input.PressureInput.sensorData.pressure <= 70",
    "actions": [
      {
        "setVariable": {
          "variableName": "pressureThresholdBreach",
          "value": "$variable.pressureThresholdBreach - 1"
        }
      }
    ]
  }
],
"transitionEvents": [
  {
    "eventName": "BackToNormal",
    "condition": "$input.PressureInput.sensorData.pressure <= 70 &&
$variable.pressureThresholdBreach <= 1",
    "nextState": "Normal"
  }
]
},
"onExit": {
  "events": [
    {
      "eventName": "Normal Pressure Restored",
      "condition": "true",
      "actions": [
        {
          "sns": {
            "targetArn": "arn:aws:sns:us-
east-1:123456789012:pressureClearedAction"
          }
        }
      ]
    }
  ]
}
},
"initialStateName": "Normal"
},
```

```
"roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole"
}
```

Output:

```
{
  "detectorModelConfiguration": {
    "status": "ACTIVATING",
    "lastUpdateTime": 1560799387.719,
    "roleArn": "arn:aws:iam::123456789012:role/IoTEventsRole",
    "creationTime": 1560799387.719,
    "detectorModelArn": "arn:aws:iotevents:us-west-2:123456789012:detectorModel/
motorDetectorModel",
    "key": "motorid",
    "detectorModelName": "motorDetectorModel",
    "detectorModelVersion": "2"
  }
}
```

For more information, see [UpdateDetectorModel](#) in the *AWS IoT Events Developer Guide**.

- For API details, see [UpdateDetectorModel](#) in *AWS CLI Command Reference*.

update-input

The following code example shows how to use `update-input`.

AWS CLI

To update an input

The following `update-input` example updates an input.

```
aws iotevents update-input \
  --cli-input-json file://pressureInput.json
```

Contents of `pressureInput.json`:

```
{
  "inputName": "PressureInput",
  "inputDescription": "Pressure readings from a motor",
```

```
"inputDefinition": {
  "attributes": [
    { "jsonPath": "sensorData.pressure" },
    { "jsonPath": "motorid" }
  ]
}
```

Output:

```
{
  "inputConfiguration": {
    "status": "ACTIVE",
    "inputArn": "arn:aws:iotevents:us-west-2:123456789012:input/PressureInput",
    "lastUpdateTime": 1560795976.458,
    "creationTime": 1560795312.542,
    "inputName": "PressureInput",
    "inputDescription": "Pressure readings from a motor"
  }
}
```

For more information, see [UpdateInput](#) in the *AWS IoT Events Developer Guide**.

- For API details, see [UpdateInput](#) in *AWS CLI Command Reference*.

AWS IoT Greengrass examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS IoT Greengrass.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-role-to-group

The following code example shows how to use `associate-role-to-group`.

AWS CLI

To associate a role with a Greengrass group

The following `associate-role-to-group` example associates the specified IAM role with a Greengrass group. The group role is used by local Lambda functions and connectors to access AWS services. For example, your group role might grant permissions required for CloudWatch Logs integration.

```
aws greengrass associate-role-to-group \  
  --group-id 2494ee3f-7f8a-4e92-a78b-d205f808b84b \  
  --role-arn arn:aws:iam::123456789012:role/GG-Group-Role
```

Output:

```
{  
  "AssociatedAt": "2019-09-10T20:03:30Z"  
}
```

For more information, see [Configure the Group Role](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [AssociateRoleToGroup](#) in *AWS CLI Command Reference*.

associate-service-role-to-account

The following code example shows how to use `associate-service-role-to-account`.

AWS CLI

To associate a service role with your AWS account

The following `associate-service-role-to-account` example associates an IAM service role, specified by its ARN, with AWS IoT Greengrass in your AWS account. You must have

previously created the service role in IAM, and you must associate a policy document with it that allows AWS IoT Greengrass to assume this role.

```
aws greengrass associate-service-role-to-account \
  --role-arn "arn:aws:iam::123456789012:role/service-role/Greengrass_ServiceRole"
```

Output:

```
{
  "AssociatedAt": "2019-06-25T18:12:45Z"
}
```

For more information, see [Greengrass Service Role](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [AssociateServiceRoleToAccount](#) in *AWS CLI Command Reference*.

create-connector-definition-version

The following code example shows how to use `create-connector-definition-version`.

AWS CLI

To create a connector definition version

The following `create-connector-definition-version` example creates a connector definition version and associates it with the specified connector definition. All connectors in a version define values for their parameters.

```
aws greengrass create-connector-definition-version \
  --connector-definition-id "55d0052b-0d7d-44d6-b56f-21867215e118" \
  --connectors "[{\\"Id\\": \\"MyTwilioNotificationsConnector\\",
  \\"ConnectorArn\\": \\"arn:aws:greengrass:us-west-2::/connectors/
  TwilioNotifications/versions/2\\", \\"Parameters\\": {\\"TWILIO_ACCOUNT_SID
  \\": \\"AC1a8d4204890840d7fc482aab38090d57\\", \\"TwilioAuthTokenSecretArn\\":
  \\"arn:aws:secretsmanager:us-west-2:123456789012:secret:greengrass-TwilioAuthToken-
  ntSlp6\\", \\"TwilioAuthTokenSecretArn-ResourceId\\": \\"TwilioAuthToken\\",
  \\"DefaultFromPhoneNumber\\": \\"4254492999\\"}]]"
```

Output:

```
{
```

```

    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
connectors/55d0052b-0d7d-44d6-b56f-21867215e118/versions/33f709a0-c825-49cb-9eea-
dc8964fbd635",
    "CreationTimestamp": "2019-06-24T20:46:30.134Z",
    "Id": "55d0052b-0d7d-44d6-b56f-21867215e118",
    "Version": "33f709a0-c825-49cb-9eea-dc8964fbd635"
}

```

- For API details, see [CreateConnectorDefinitionVersion](#) in *AWS CLI Command Reference*.

create-connector-definition

The following code example shows how to use `create-connector-definition`.

AWS CLI

To create a connector definition

The following `create-connector-definition` example creates a connector definition and an initial connector definition version. The initial version contains one connector. All connectors in a version define values for their parameters.

```

aws greengrass create-connector-definition \
  --name MySNSConnector \
  --initial-version "{\"Connectors\": [{\"Id\": \"MySNSConnector\", \"ConnectorArn\": \"arn:aws:greengrass:us-west-2:/connectors/SNS/versions/1\", \"Parameters\": {\"DefaultSNSArn\": \"arn:aws:sns:us-west-2:123456789012:GGConnectorTopic\"}}]}"

```

Output:

```

{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
connectors/b5c4ebfd-f672-49a3-83cd-31c7216a7bb8",
  "CreationTimestamp": "2019-06-19T19:30:01.300Z",
  "Id": "b5c4ebfd-f672-49a3-83cd-31c7216a7bb8",
  "LastUpdatedTimestamp": "2019-06-19T19:30:01.300Z",
  "LatestVersion": "63c57963-c7c2-4a26-a7e2-7bf478ea2623",
  "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/connectors/b5c4ebfd-f672-49a3-83cd-31c7216a7bb8/versions/63c57963-
c7c2-4a26-a7e2-7bf478ea2623",
  "Name": "MySNSConnector"
}

```

```
}
```

For more information, see [Getting Started with Greengrass Connectors \(CLI\)](#) in the **AWS IoT Greengrass Developer Guide**.

- For API details, see [CreateConnectorDefinition](#) in *AWS CLI Command Reference*.

create-core-definition-version

The following code example shows how to use `create-core-definition-version`.

AWS CLI

To create a core definition version

The following `create-core-definition-version` example creates a core definition version and associates it with the specified core definition. The version can contain one core only. Before you can create a core, you must first create and provision the corresponding AWS IoT thing. This process includes the following `iot` commands, which return the `ThingArn` and `CertificateArn` required for the `create-core-definition-version` command.

Create the AWS IoT thing that corresponds to the core device:

```
aws iot create-thing \  
  --thing-name "MyCoreDevice"
```

Output:

```
{  
  "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MyCoreDevice",  
  "thingName": "MyCoreDevice",  
  "thingId": "cb419a19-9099-4515-9cec-e9b0e760608a"  
}
```

Create public and private keys and the core device certificate for the thing. This example uses the `create-keys-and-certificate` command and requires write permissions to the current directory. Alternatively, you can use the `create-certificate-from-csr` command.

```
aws iot create-keys-and-certificate \  
  --set-as-active \  
  --certificate-pem-outfile "myCore.cert.pem" \  
  \
```

```
--public-key-outfile "myCore.public.key" \
--private-key-outfile "myCore.private.key"
```

Output:

```
{
  "certificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz",
  "certificatePem": "-----BEGIN CERTIFICATE-----
\nMIIDWTCakGgAwIBATgIUCGq6EGqou6zFqWgIZRndgQEFW+gwDQYJKoZIhvc...KdGewQS\n-----END
CERTIFICATE-----\n",
  "keyPair": {
    "PublicKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBzrqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAqKpRgnn6yq26U3y...wIDAQAB\n-----END
PUBLIC KEY-----\n",
    "PrivateKey": "-----BEGIN RSA PRIVATE KEY-----
\nMIIIEowIABAKCAQEAqKpRgnn6yq26U3yt5YFZquyukfRjbmXDCnOK4rMCxDR...fvY4+te\n-----END
RSA PRIVATE KEY-----\n"
  },
  "certificateId":
  "123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz"
}
```

Create an AWS IoT policy that allows `iot` and `greengrass` actions. For simplicity, the following policy allows actions on all resources, but your policy should be more restrictive.

```
aws iot create-policy \
  --policy-name "Core_Devices" \
  --policy-document "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":"
  \"Allow\",\"Action\":[\"iot:Publish\",\"iot:Subscribe\",\"iot:Connect
  \",\"iot:Receive\"],\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":"
  [\"iot:GetThingShadow\",\"iot:UpdateThingShadow\",\"iot:DeleteThingShadow\"],
  \"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":[\"greengrass:*\"],\"Resource
  \":[\"*\"]}]}"
```

Output:

```
{
  "policyName": "Core_Devices",
  "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/Core_Devices",
  "policyDocument": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":"
  \"Allow\",\"Action\":[\"iot:Publish\",\"iot:Subscribe\",\"iot:Connect
```

```

\", \"iot:Receive\"], \"Resource\": [\"*\"]}, {\"Effect\": \"Allow\", \"Action\":
[\"iot:GetThingShadow\", \"iot:UpdateThingShadow\", \"iot>DeleteThingShadow\"],
\"Resource\": [\"*\"]}, {\"Effect\": \"Allow\", \"Action\": [\"greengrass:*\"], \"Resource
\": [\"*\"]}]}",
  "policyVersionId": "1"
}

```

Attach the policy to the certificate:

```

aws iot attach-policy \
  --policy-name "Core_Devices" \
  --target "arn:aws:iot:us-
west-2:123456789012:cert/123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz"

```

This command produces no output.

Attach the thing to the certificate:

```

aws iot attach-thing-principal \
  --thing-name "MyCoreDevice" \
  --principal "arn:aws:iot:us-
west-2:123456789012:cert/123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz"

```

This command produces no output.

Create the core definition version:

```

aws greengrass create-core-definition-version \
  --core-definition-id "582efe12-b05a-409e-9a24-a2ba1bcc4a12" \
  --cores "[{\"Id\": \"MyCoreDevice\", \"ThingArn\": \"arn:aws:iot:us-
west-2:123456789012:thing/MyCoreDevice\", \"CertificateArn\": \"arn:aws:iot:us-
west-2:123456789012:cert/123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz
\", \"SyncShadow\": true}]"

```

Output:

```

{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
cores/582efe12-b05a-409e-9a24-a2ba1bcc4a12/versions/3fdc1190-2ce5-44de-b98b-
eec8f9571014",
  "Version": "3fdc1190-2ce5-44de-b98b-eec8f9571014",
  "CreationTimestamp": "2019-09-18T00:15:09.838Z",

```

```
"Id": "582efe12-b05a-409e-9a24-a2ba1bcc4a12"
}
```

For more information, see [Configure the AWS IoT Greengrass Core](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [CreateCoreDefinitionVersion](#) in *AWS CLI Command Reference*.

create-core-definition

The following code example shows how to use `create-core-definition`.

AWS CLI

Example 1: To create an empty core definition

The following `create-core-definition` example creates an empty (no initial version) Greengrass core definition. Before the core is usable, you must use the `create-core-definition-version` command to provide the other parameters for the core.

```
aws greengrass create-core-definition \
  --name cliGroup_Core
```

Output:

```
{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/cores/
b5c08008-54cb-44bd-9eec-c121b04283b5",
  "CreationTimestamp": "2019-06-25T18:23:22.106Z",
  "Id": "b5c08008-54cb-44bd-9eec-c121b04283b5",
  "LastUpdatedTimestamp": "2019-06-25T18:23:22.106Z",
  "Name": "cliGroup_Core"
}
```

Example 2: To create a core definition with an initial version

The following `create-core-definition` example creates a core definition that contains an initial core definition version. The version can contain one core only. Before you can create a core, you must first create and provision the corresponding AWS IoT thing. This process includes the following `iot` commands, which return the `ThingArn` and `CertificateArn` required for the `create-core-definition` command.

Create the AWS IoT thing that corresponds to the core device:

```
aws iot create-thing \
  --thing-name "MyCoreDevice"
```

Output:

```
{
  "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MyCoreDevice",
  "thingName": "MyCoreDevice",
  "thingId": "cb419a19-9099-4515-9cec-e9b0e760608a"
}
```

Create public and private keys and the core device certificate for the thing. This example uses the `create-keys-and-certificate` command and requires write permissions to the current directory. Alternatively, you can use the `create-certificate-from-csr` command.

```
aws iot create-keys-and-certificate \
  --set-as-active \
  --certificate-pem-outfile "myCore.cert.pem" \
  --public-key-outfile "myCore.public.key" \
  --private-key-outfile "myCore.private.key"
```

Output:

```
{
  "certificateArn": "arn:aws:iot:us-west-2:123456789012:cert/123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz",
  "certificatePem": "-----BEGIN CERTIFICATE-----
\nMIIDWTCAkGgAwIBATgIUCgq6EGqou6zFqWgIZRndgQEFW+gwDQYJKoZIhvc...KdGewQS\n-----END
CERTIFICATE-----\n",
  "keyPair": {
    "PublicKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBzrqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAAqKpRgnn6yq26U3y...wIDAQAB\n-----END
PUBLIC KEY-----\n",
    "PrivateKey": "-----BEGIN RSA PRIVATE KEY-----
\nMIIEowIABAKCAQEAAqKpRgnn6yq26U3yt5YFZquyukfRjBMXDcNOK4rMCxDR...fvY4+te\n-----END
RSA PRIVATE KEY-----\n"
  },
  "certificateId":
  "123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz"
```

```
}

```

Create an AWS IoT policy that allows `iot` and `greengrass` actions. For simplicity, the following policy allows actions on all resources, but your policy should be more restrictive.

```
aws iot create-policy \
  --policy-name "Core_Devices" \
  --policy-document "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":"
  "\"Allow\",\"Action\":[\"iot:Publish\",\"iot:Subscribe\",\"iot:Connect\","
  "\"iot:Receive\"],\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":"
  "\"iot:GetThingShadow\",\"iot:UpdateThingShadow\",\"iot>DeleteThingShadow\"},\","
  "\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":[\"greengrass:*\"],\"Resource\":"
  "\"*\"]}]}"
```

Output:

```
{
  "policyName": "Core_Devices",
  "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/Core_Devices",
  "policyDocument": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":"
  "\"Allow\",\"Action\":[\"iot:Publish\",\"iot:Subscribe\",\"iot:Connect\","
  "\"iot:Receive\"],\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":"
  "\"iot:GetThingShadow\",\"iot:UpdateThingShadow\",\"iot>DeleteThingShadow\"},\","
  "\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":[\"greengrass:*\"],\"Resource\":"
  "\"*\"]}]}",
  "policyVersionId": "1"
}
```

Attach the policy to the certificate:

```
aws iot attach-policy \
  --policy-name "Core_Devices" \
  --target "arn:aws:iot:us-
west-2:123456789012:cert/123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz"
```

This command produces no output.

Attach the thing to the certificate:

```
aws iot attach-thing-principal \
  --thing-name "MyCoreDevice" \
```



```
--principal "arn:aws:iot:us-west-2:123456789012:cert/123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz"
```

This command produces no output.

Create the core definition:

```
aws greengrass create-core-definition \
  --name "MyCores" \
  --initial-version "{ \"Cores\": [{ \"Id\": \"MyCoreDevice\", \"ThingArn\": \"arn:aws:iot:us-west-2:123456789012:thing/MyCoreDevice\", \"CertificateArn\": \"arn:aws:iot:us-west-2:123456789012:cert/123a15ec415668c2349a76170b64ac0878231c1e21ec83c10e92a1EXAMPLExyz\", \"SyncShadow\": true } ] }"
```

Output:

```
{
  "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/cores/582efe12-b05a-409e-9a24-a2ba1bcc4a12/versions/cc87b5b3-8f4b-465d-944c-1d6de5dbfcdb",
  "Name": "MyCores",
  "LastUpdatedTimestamp": "2019-09-18T00:11:06.197Z",
  "LatestVersion": "cc87b5b3-8f4b-465d-944c-1d6de5dbfcdb",
  "CreationTimestamp": "2019-09-18T00:11:06.197Z",
  "Id": "582efe12-b05a-409e-9a24-a2ba1bcc4a12",
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/cores/582efe12-b05a-409e-9a24-a2ba1bcc4a12"
}
```

For more information, see [Configure the AWS IoT Greengrass Core](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [CreateCoreDefinition](#) in *AWS CLI Command Reference*.

create-deployment

The following code example shows how to use create-deployment.

AWS CLI

To create a deployment for a version of a Greengrass group

The following `create-deployment` example deploys the specified version of a Greengrass group.

```
aws greengrass create-deployment \  
  --deployment-type NewDeployment \  
  --group-id "ce2e7d01-3240-4c24-b8e6-f6f6e7a9eeca" \  
  --group-version-id "dc40c1e9-e8c8-4d28-a84d-a9cad5f599c9"
```

Output:

```
{  
  "DeploymentArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/  
groups/ce2e7d01-3240-4c24-b8e6-f6f6e7a9eeca/deployments/bfceb608-4e97-45bc-  
af5c-460144270308",  
  "DeploymentId": "bfceb608-4e97-45bc-af5c-460144270308"  
}
```

For more information, see [Getting Started with Connectors \(CLI\)](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [CreateDeployment](#) in *AWS CLI Command Reference*.

create-device-definition-version

The following code example shows how to use `create-device-definition-version`.

AWS CLI

To create a device definition version

The following `create-device-definition-version` example creates a device definition version and associates it with the specified device definition. The version defines two devices. Before you can create a Greengrass device, you must first create and provision the corresponding AWS IoT thing. This process includes the following `iot` commands that you must run to get the required information for the Greengrass command:

Create the AWS IoT thing that corresponds to the device:

```
aws iot create-thing \  
  --thing-name "InteriorTherm"
```

Output:

```
{
  "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/InteriorTherm",
  "thingName": "InteriorTherm",
  "thingId": "01d4763c-78a6-46c6-92be-7add080394bf"
}
```

Create public and private keys and the device certificate for the thing. This example uses the `create-keys-and-certificate` command and requires write permissions to the current directory. Alternatively, you can use the `create-certificate-from-csr` command:

```
aws iot create-keys-and-certificate \
  --set-as-active \
  --certificate-pem-outfile "myDevice.cert.pem" \
  --public-key-outfile "myDevice.public.key" \
  --private-key-outfile "myDevice.private.key"
```

Output:

```
{
  "certificateArn": "arn:aws:iot:us-west-2:123456789012:cert/66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92",
  "certificatePem": "-----BEGIN CERTIFICATE-----
\nMIIDWTCakGgAwIBATgIUCGq6EGqou6zFqWgIZRndgQEFW+gwDQYJKoZIhvc...KdGewQS\n-----END
CERTIFICATE-----\n",
  "keyPair": {
    "PublicKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBzrqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAqKpRgnn6yq26U3y...wIDAQAB\n-----END
PUBLIC KEY-----\n",
    "PrivateKey": "-----BEGIN RSA PRIVATE KEY-----
\nMIIEowIABAKCAQEAqKpRgnn6yq26U3yt5YFZquyukfRjbmXDCnOK4rMCxDR...fvY4+te\n-----END
RSA PRIVATE KEY-----\n"
  },
  "certificateId":
  "66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92"
}
```

Create an AWS IoT policy that allows `iot` and `greengrass` actions. For simplicity, the following policy allows actions on all resources, but your policy can be more restrictive:

```
aws iot create-policy \
```

```
--policy-name "GG_Devices" \
--policy-document "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",\"Action\":[\"iot:Publish\",\"iot:Subscribe\",\"iot:Connect\",\"iot:Receive\"],\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":[\"iot:GetThingShadow\",\"iot:UpdateThingShadow\",\"iot:DeleteThingShadow\"],\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":[\"greengrass:*\"],\"Resource\":[\"*\"]}]}"
```

Output:

```
{
  "policyName": "GG_Devices",
  "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/GG_Devices",
  "policyDocument": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",\"Action\":[\"iot:Publish\",\"iot:Subscribe\",\"iot:Connect\",\"iot:Receive\"],\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":[\"iot:GetThingShadow\",\"iot:UpdateThingShadow\",\"iot:DeleteThingShadow\"],\"Resource\":[\"*\"]},{\"Effect\":\"Allow\",\"Action\":[\"greengrass:*\"],\"Resource\":[\"*\"]}]}",
  "policyVersionId": "1"
}
```

Attach the policy to the certificate:

```
aws iot attach-policy \
--policy-name "GG_Devices" \
--target "arn:aws:iot:us-west-2:123456789012:cert/66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92"
```

Attach the thing to the certificate

```
aws iot attach-thing-principal \
--thing-name "InteriorTherm" \
--principal "arn:aws:iot:us-west-2:123456789012:cert/66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92"
```

After you create and configure the IoT thing as shown above, use the `ThingArn` and `CertificateArn` from the first two commands in the following example.

```
aws greengrass create-device-definition-version \
--device-definition-id "f9ba083d-5ad4-4534-9f86-026a45df1ccd" \
```

```
--devices "[{\\"Id\\":\\"InteriorTherm\\",\\"ThingArn\\":\\"arn:aws:iot:us-west-2:123456789012:thing/InteriorTherm\\",\\"CertificateArn\\":\\"arn:aws:iot:us-west-2:123456789012:cert/66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92\\",\\"SyncShadow\\":true},{\\"Id\\":\\"ExteriorTherm\\",\\"ThingArn\\":\\"arn:aws:iot:us-west-2:123456789012:thing/ExteriorTherm\\",\\"CertificateArn\\":\\"arn:aws:iot:us-west-2:123456789012:cert/6c52ce1b47bde88a637e9ccdd45fe4e4c2c0a75a6866f8f63d980ee22fa51e02\\",\\"SyncShadow\\":true}]"
```

Output:

```
{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd/versions/83c13984-6fed-447e-84d5-5b8aa45d5f71",
  "Version": "83c13984-6fed-447e-84d5-5b8aa45d5f71",
  "CreationTimestamp": "2019-09-11T00:15:09.838Z",
  "Id": "f9ba083d-5ad4-4534-9f86-026a45df1ccd"
}
```

- For API details, see [CreateDeviceDefinitionVersion](#) in *AWS CLI Command Reference*.

create-device-definition

The following code example shows how to use create-device-definition.

AWS CLI**To create a device definition**

The following create-device-definition example creates a device definition that contains an initial device definition version. The initial version defines two devices. Before you can create a Greengrass device, you must first create and provision the corresponding AWS IoT thing. This process includes the following `iot` commands that you must run to get the required information for the Greengrass command:

Create the AWS IoT thing that corresponds to the device:

```
aws iot create-thing \
  --thing-name "InteriorTherm"
```

Output:

```
{
  "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/InteriorTherm",
  "thingName": "InteriorTherm",
  "thingId": "01d4763c-78a6-46c6-92be-7add080394bf"
}
```

Create public and private keys and the device certificate for the thing. This example uses the `create-keys-and-certificate` command and requires write permissions to the current directory. Alternatively, you can use the `create-certificate-from-csr` command:

```
aws iot create-keys-and-certificate \
  --set-as-active \
  --certificate-pem-outfile "myDevice.cert.pem" \
  --public-key-outfile "myDevice.public.key" \
  --private-key-outfile "myDevice.private.key"
```

Output:

```
{
  "certificateArn": "arn:aws:iot:us-west-2:123456789012:cert/66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92",
  "certificatePem": "-----BEGIN CERTIFICATE-----
\nMIIDWTCAkGgAwIBATgIUCgq6EGqou6zFqWgIZRndgQEFW+gwDQYJKoZIhvc...KdGewQS\n-----END
CERTIFICATE-----\n",
  "keyPair": {
    "PublicKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBzrqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAqKpRgnn6yq26U3y...wIDAQAB\n-----END
PUBLIC KEY-----\n",
    "PrivateKey": "-----BEGIN RSA PRIVATE KEY-----
\nMIIEowIABAKCAQEAqKpRgnn6yq26U3yt5YFZquyukfRjBMXDcNOK4rMCxDR...fvY4+te\n-----END
RSA PRIVATE KEY-----\n"
  },
  "certificateId":
  "66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92"
}
```

Create an AWS IoT policy that allows `iot` and `greengrass` actions. For simplicity, the following policy allows actions on all resources, but your policy can be more restrictive:

```
aws iot create-policy \
```

```
--policy-name "GG_Devices" \
--policy-document "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\": \"Allow\", \"Action\": [\"iot:Publish\", \"iot:Subscribe\", \"iot:Connect\", \"iot:Receive\"], \"Resource\": [\"*\"]}, {\"Effect\": \"Allow\", \"Action\": [\"iot:GetThingShadow\", \"iot:UpdateThingShadow\", \"iot:DeleteThingShadow\"], \"Resource\": [\"*\"]}, {\"Effect\": \"Allow\", \"Action\": [\"greengrass:*\"], \"Resource\": [\"*\"]}]}"
```

Output:

```
{
  "policyName": "GG_Devices",
  "policyArn": "arn:aws:iot:us-west-2:123456789012:policy/GG_Devices",
  "policyDocument": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\": \"Allow\", \"Action\": [\"iot:Publish\", \"iot:Subscribe\", \"iot:Connect\", \"iot:Receive\"], \"Resource\": [\"*\"]}, {\"Effect\": \"Allow\", \"Action\": [\"iot:GetThingShadow\", \"iot:UpdateThingShadow\", \"iot:DeleteThingShadow\"], \"Resource\": [\"*\"]}, {\"Effect\": \"Allow\", \"Action\": [\"greengrass:*\"], \"Resource\": [\"*\"]}]}",
  "policyVersionId": "1"
}
```

Attach the policy to the certificate:

```
aws iot attach-policy \
  --policy-name "GG_Devices" \
  --target "arn:aws:iot:us-west-2:123456789012:cert/66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92"
```

Attach the thing to the certificate

```
aws iot attach-thing-principal \
  --thing-name "InteriorTherm" \
  --principal "arn:aws:iot:us-west-2:123456789012:cert/66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92"
```

After you create and configure the IoT thing as shown above, use the ThingArn and CertificateArn from the first two commands in the following example.

```
aws greengrass create-device-definition \
```

```
--name "Sensors" \
--initial-version "{ \"Devices\": [{ \"Id\": \"InteriorTherm
\", \"ThingArn\": \"arn:aws:iot:us-west-2:123456789012:thing/
InteriorTherm\", \"CertificateArn\": \"arn:aws:iot:us-
west-2:123456789012:cert/66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92\",
\", \"SyncShadow\": true}, { \"Id\": \"ExteriorTherm\", \"ThingArn\": \"arn:aws:iot:us-
west-2:123456789012:thing/ExteriorTherm\", \"CertificateArn\": \"arn:aws:iot:us-
west-2:123456789012:cert/6c52ce1b47bde88a637e9ccdd45fe4e4c2c0a75a6866f8f63d980ee22fa51e02\",
\", \"SyncShadow\": true} ] } }
```

Output:

```
{
  "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd/
versions/3b5cc510-58c1-44b5-9d98-4ad858ffa795",
  "Name": "Sensors",
  "LastUpdatedTimestamp": "2019-09-11T00:11:06.197Z",
  "LatestVersion": "3b5cc510-58c1-44b5-9d98-4ad858ffa795",
  "CreationTimestamp": "2019-09-11T00:11:06.197Z",
  "Id": "f9ba083d-5ad4-4534-9f86-026a45df1ccd",
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd"
}
```

- For API details, see [CreateDeviceDefinition](#) in *AWS CLI Command Reference*.

create-function-definition-version

The following code example shows how to use `create-function-definition-version`.

AWS CLI

To create a version of the function definition

The following `create-function-definition-version` example creates a new version of the specified function definition. This version specifies a single function whose ID is `Hello-World-function`, allows access to the file system, and specifies a maximum memory size and timeout period.

```
aws greengrass create-function-definition-version \
```



```
--cli-input-json "{\"FunctionDefinitionId\": \"e626e8c9-3b8f-4bf3-9cdc-
d26ecdeb9fa3\", \"Functions\": [{\"Id\": \"Hello-World-function\", \"FunctionArn\":
\"arn:aws:lambda:us-
west-2:123456789012:function:Greengrass_HelloWorld_Counter:gghw-alias\"},
{\"FunctionConfiguration\": {\"Environment\": {\"AccessSysfs\": true}, \"Executable\":
\"greengrassHelloWorldCounter.function_handler\", \"MemorySize\": 16000, \"Pinned\":
false, \"Timeout\": 25}}]}"
```

Output:

```
{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/functions/e626e8c9-3b8f-4bf3-9cdc-d26ecdeb9fa3/
versions/74abd1cc-637e-4abe-8684-9a67890f4043",
  "CreationTimestamp": "2019-06-25T22:03:43.376Z",
  "Id": "e626e8c9-3b8f-4bf3-9cdc-d26ecdeb9fa3",
  "Version": "74abd1cc-637e-4abe-8684-9a67890f4043"
}
```

- For API details, see [CreateFunctionDefinitionVersion](#) in *AWS CLI Command Reference*.

create-function-definition

The following code example shows how to use `create-function-definition`.

AWS CLI

To create a Lambda function definition

The following `create-function-definition` example creates a Lambda function definition and an initial version by providing a list of Lambda functions (in this case, a list of just one function named `TempMonitorFunction`) and their configurations. Before you can create the function definition, you need the Lambda function ARN. To create the function and its alias, use Lambda's `create-function` and `publish-version` commands. Lambda's `create-function` command requires the ARN of the execution role, even though AWS IoT Greengrass doesn't use that role because permissions are specified in the Greengrass group role. You can use the IAM `create-role` command to create an empty role to get an ARN to use with Lambda's `create-function` or you can use an existing execution role.

```
aws greengrass create-function-definition \
```

```
--name MyGreengrassFunctions \
--initial-version "{\"Functions\": [{\"Id\": \"TempMonitorFunction\",
\"FunctionArn\": \"arn:aws:lambda:us-
west-2:123456789012:function:TempMonitor:GG_TempMonitor\", \"FunctionConfiguration
\": {\"Executable\": \"temp_monitor.function_handler\", \"MemorySize\": 16000,
\"Timeout\": 5}}]}}"
```

Output:

```
{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
functions/3b0d0080-87e7-48c6-b182-503ec743a08b",
  "CreationTimestamp": "2019-06-19T22:24:44.585Z",
  "Id": "3b0d0080-87e7-48c6-b182-503ec743a08b",
  "LastUpdatedTimestamp": "2019-06-19T22:24:44.585Z",
  "LatestVersion": "67f918b9-efb4-40b0-b87c-de8c9faf085b",
  "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/functions/3b0d0080-87e7-48c6-b182-503ec743a08b/versions/67f918b9-
efb4-40b0-b87c-de8c9faf085b",
  "Name": "MyGreengrassFunctions"
}
```

For more information, see [How to Configure Local Resource Access Using the AWS Command Line Interface](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [CreateFunctionDefinition](#) in *AWS CLI Command Reference*.

create-group-certificate-authority

The following code example shows how to use `create-group-certificate-authority`.

AWS CLI**To create a certificate authority (CA) for a group**

The following `create-group-certificate-authority` example creates or rotates a CA for the specified group.

```
aws greengrass create-group-certificate-authority \
--group-id "8eaadd72-ce4b-4f15-892a-0cc4f3a343f1"
```

Output:

```
{
  "GroupCertificateAuthorityArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/8eaadd72-ce4b-4f15-892a-0cc4f3a343f1/certificateauthorities/
d31630d674c4437f6c5dbc0dca56312a902171ce2d086c38e509c8EXAMPLEecc5"
}
```

For more information, see [AWS IoT Greengrass Security](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [CreateGroupCertificateAuthority](#) in *AWS CLI Command Reference*.

create-group-version

The following code example shows how to use `create-group-version`.

AWS CLI

To create a version of a Greengrass group

The following `create-group-version` example creates a group version and associates it with the specified group. The version references the core, resource, connector, function, and subscription versions that contain the entities to include in this group version. You must create these entities before you can create the group version.

To create a resource definition with an initial version, use the `create-resource-definition` command. To create a connector definition with an initial version, use the `create-connector-definition` command. To create a function definition with an initial version, use the `create-function-definition` command. To create a subscription definition with an initial version, use the `create-subscription-definition` command. To retrieve the ARN of the latest core definition version, use the `get-group-version` command and specify the ID of the latest group version.

```
aws greengrass create-group-version \
  --group-id "ce2e7d01-3240-4c24-b8e6-f6f6e7a9eeca" \
  --core-definition-version-arn "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/cores/6a630442-8708-4838-ad36-eb98849d975e/
versions/6c87151b-1fb4-4cb2-8b31-6ee715d8f8ba" \
  --resource-definition-version-arn "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/resources/c8bb9ebc-c3fd-40a4-9c6a-568d75569d38/versions/
a5f94d0b-f6bc-40f4-bb78-7a1c5fe13ba1" \
```

```

--connector-definition-version-arn "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/connectors/55d0052b-0d7d-44d6-b56f-21867215e118/versions/78a3331b-895d-489b-8823-17b4f9f418a0" \
--function-definition-version-arn "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/functions/3b0d0080-87e7-48c6-b182-503ec743a08b/versions/67f918b9-efb4-40b0-b87c-de8c9faf085b" \
--subscription-definition-version-arn "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/subscriptions/9d611d57-5d5d-44bd-a3b4-fecbbdd69112/versions/aa645c47-ac90-420d-9091-8c7ffa4f103f"

```

Output:

```

{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/groups/ce2e7d01-3240-4c24-b8e6-f6f6e7a9eeca/versions/e10b0459-4345-4a09-88a4-1af1f5d34638",
  "CreationTimestamp": "2019-06-20T18:42:47.020Z",
  "Id": "ce2e7d01-3240-4c24-b8e6-f6f6e7a9eeca",
  "Version": "e10b0459-4345-4a09-88a4-1af1f5d34638"
}

```

For more information, see [Overview of the AWS IoT Greengrass Group Object Model](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [CreateGroupVersion](#) in *AWS CLI Command Reference*.

create-group

The following code example shows how to use create-group.

AWS CLI

To create a Greengrass group

The following create-group example creates a group named cli-created-group.

```

aws greengrass create-group \
  --name cli-created-group

```

Output:

```

{

```

```

    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/4e22bd92-898c-436b-ade5-434d883ff749",
    "CreationTimestamp": "2019-06-25T18:07:17.688Z",
    "Id": "4e22bd92-898c-436b-ade5-434d883ff749",
    "LastUpdatedTimestamp": "2019-06-25T18:07:17.688Z",
    "Name": "cli-created-group"
}

```

For more information, see [Overview of the AWS IoT Greengrass Group Object Model](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [CreateGroup](#) in *AWS CLI Command Reference*.

create-logger-definition-version

The following code example shows how to use `create-logger-definition-version`.

AWS CLI

To create a logger definition version

The following `create-logger-definition-version` example creates a logger definition version and associates it with a logger definition. The version defines four logging configurations: 1) system component logs on the file system of the core device, 2) user-defined Lambda function logs on the file system of the core device, 3) system component logs in Amazon CloudWatch Logs, and 4) user-defined Lambda function logs in Amazon CloudWatch Logs. Note: For CloudWatch Logs integration, your group role must grant appropriate permissions.

```

aws greengrass create-logger-definition-version \
  --logger-definition-id "a454b62a-5d56-4ca9-bdc4-8254e1662cb0" \
  --loggers "[{\\"Id\\":\\"1\\",\\"Component\\":\\"GreengrassSystem\\",\\"Level\\":\\"ERROR\\",\\"Space\\":10240,\\"Type\\":\\"FileSystem\\"},{\\"Id\\":\\"2\\",\\"Component\\":\\"Lambda\\",\\"Level\\":\\"INFO\\",\\"Space\\":10240,\\"Type\\":\\"FileSystem\\"},{\\"Id\\":\\"3\\",\\"Component\\":\\"GreengrassSystem\\",\\"Level\\":\\"WARN\\",\\"Type\\":\\"AWSCloudWatch\\"},{\\"Id\\":\\"4\\",\\"Component\\":\\"Lambda\\",\\"Level\\":\\"INFO\\",\\"Type\\":\\"AWSCloudWatch\\"}]"

```

Output:

```
{
```

```

"Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/loggers/
a454b62a-5d56-4ca9-bdc4-8254e1662cb0/versions/49aedb1e-01a3-4d39-9871-3a052573f1ea",
"Version": "49aedb1e-01a3-4d39-9871-3a052573f1ea",
"CreationTimestamp": "2019-07-24T00:04:48.523Z",
"Id": "a454b62a-5d56-4ca9-bdc4-8254e1662cb0"
}

```

For more information, see [Monitoring with AWS IoT Greengrass Logs](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [CreateLoggerDefinitionVersion](#) in *AWS CLI Command Reference*.

create-logger-definition

The following code example shows how to use `create-logger-definition`.

AWS CLI

To create a logger definition

The following `create-logger-definition` example creates a logger definition that contains an initial logger definition version. The initial version defines three logging configurations:

1) system component logs on the file system of the core device, 2) user-defined Lambda function logs on the file system of the core device, and 3) user-defined Lambda function logs in Amazon CloudWatch Logs. Note: For CloudWatch Logs integration, your group role must grant appropriate permissions.

```

aws greengrass create-logger-definition \
  --name "LoggingConfigs" \
  --initial-version "{\"Loggers\":{\"Id\":\"1\",\"Component\":\"GreengrassSystem
\", \"Level\":\"ERROR\", \"Space\":10240, \"Type\":\"FileSystem\"}, {\"Id\":
\"2\", \"Component\":\"Lambda\", \"Level\":\"INFO\", \"Space\":10240, \"Type\":
\"FileSystem\"}, {\"Id\":\"3\", \"Component\":\"Lambda\", \"Level\":\"INFO\", \"Type\":
\"AWSCloudWatch\"}}}"

```

Output:

```

{
  "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/loggers/a454b62a-5d56-4ca9-bdc4-8254e1662cb0/versions/de1d9854-1588-4525-
b25e-b378f60f2322",

```

```

    "Name": "LoggingConfigs",
    "LastUpdatedTimestamp": "2019-07-23T23:52:17.165Z",
    "LatestVersion": "de1d9854-1588-4525-b25e-b378f60f2322",
    "CreationTimestamp": "2019-07-23T23:52:17.165Z",
    "Id": "a454b62a-5d56-4ca9-bdc4-8254e1662cb0",
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
loggers/a454b62a-5d56-4ca9-bdc4-8254e1662cb0"
}

```

For more information, see [Monitoring with AWS IoT Greengrass Logs](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [CreateLoggerDefinition](#) in *AWS CLI Command Reference*.

create-resource-definition-version

The following code example shows how to use `create-resource-definition-version`.

AWS CLI

To create a version of a resource definition

The following `create-resource-definition-version` example creates a new version of a `TwilioAuthToken`.

```

aws greengrass create-resource-definition-version \
  --resource-definition-id "c8bb9ebc-c3fd-40a4-9c6a-568d75569d38" \
  --resources "[{"Id": \"TwilioAuthToken\", \"Name\": \"MyTwilioAuthToken
\", \"ResourceDataContainer\": {\"SecretsManagerSecretResourceData\": {\"ARN\":
\"arn:aws:secretsmanager:us-west-2:123456789012:secret:greengrass-TwilioAuthToken-
ntS1p6\"}}}]]"

```

Output:

```

{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
resources/c8bb9ebc-c3fd-40a4-9c6a-568d75569d38/versions/b3bcada0-5fb6-42df-
bf0b-1ee4f15e769e",
  "CreationTimestamp": "2019-06-24T21:17:25.623Z",
  "Id": "c8bb9ebc-c3fd-40a4-9c6a-568d75569d38",
  "Version": "b3bcada0-5fb6-42df-bf0b-1ee4f15e769e"
}

```

- For API details, see [CreateResourceDefinitionVersion](#) in *AWS CLI Command Reference*.

create-resource-definition

The following code example shows how to use `create-resource-definition`.

AWS CLI

To create a resource definition

The following `create-resource-definition` example creates a resource definition that contains a list of resources to be used in a Greengrass group. In this example, an initial version of the resource definition is included by providing a list of resources. The list includes one resource for a Twilio authorization token and the ARN for a secret stored in AWS Secrets Manager. You must create the secret before you can create the resource definition.

```
aws greengrass create-resource-definition \  
  --name MyGreengrassResources \  
  --initial-version "{\"Resources\": [{\"Id\": \"TwilioAuthToken\  
\", \"Name\": \"MyTwilioAuthToken\", \"ResourceDataContainer\":  
  {\"SecretsManagerSecretResourceData\": {\"ARN\": \"arn:aws:secretsmanager:us-  
west-2:123456789012:secret:greengrass-TwilioAuthToken-ntSlp6\"}}}]}"
```

Output:

```
{  
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/  
resources/c8bb9ebc-c3fd-40a4-9c6a-568d75569d38",  
  "CreationTimestamp": "2019-06-19T21:51:28.212Z",  
  "Id": "c8bb9ebc-c3fd-40a4-9c6a-568d75569d38",  
  "LastUpdatedTimestamp": "2019-06-19T21:51:28.212Z",  
  "LatestVersion": "a5f94d0b-f6bc-40f4-bb78-7a1c5fe13ba1",  
  "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/  
definition/resources/c8bb9ebc-c3fd-40a4-9c6a-568d75569d38/versions/a5f94d0b-  
f6bc-40f4-bb78-7a1c5fe13ba1",  
  "Name": "MyGreengrassResources"  
}
```

For more information, see [How to Configure Local Resource Access Using the AWS Command Line Interface](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [CreateResourceDefinition](#) in *AWS CLI Command Reference*.

create-software-update-job

The following code example shows how to use `create-software-update-job`.

AWS CLI

To create a software update job for a core

The following `create-software-update-job` example creates an over-the-air (OTA) update job to update the AWS IoT Greengrass Core software on the core whose name is `MyFirstGroup_Core`. This command requires an IAM role that allows access to software update packages in Amazon S3 and includes `iot.amazonaws.com` as a trusted entity.

```
aws greengrass create-software-update-job \  
  --update-targets-architecture armv7l \  
  --update-targets ["arn:aws:iot:us-west-2:123456789012:thing/MyFirstGroup_Core \  
  \"] \  
  --update-targets-operating-system raspbian \  
  --software-to-update core \  
  --s3-url-signer-role arn:aws:iam::123456789012:role/OTA_signer_role \  
  --update-agent-log-level WARN
```

Output:

```
{  
  "IotJobId": "GreengrassUpdateJob_30b353e3-3af7-4786-be25-4c446663c09e",  
  "IotJobArn": "arn:aws:iot:us-west-2:123456789012:job/  
GreengrassUpdateJob_30b353e3-3af7-4786-be25-4c446663c09e",  
  "PlatformSoftwareVersion": "1.9.3"  
}
```

For more information, see [OTA Updates of AWS IoT Greengrass Core Software](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [CreateSoftwareUpdateJob](#) in *AWS CLI Command Reference*.

create-subscription-definition-version

The following code example shows how to use `create-subscription-definition-version`.

AWS CLI

To create a new version of a subscription definition

The following `create-subscription-definition-version` example creates a new version of a subscription definition that contains three subscriptions: a trigger notification, a temperature input, and an output status.

```
aws greengrass create-subscription-definition-version \
  --subscription-definition-id "9d611d57-5d5d-44bd-a3b4-feccbdd69112" \
  --subscriptions "[{\\"Id\\": \\"TriggerNotification\\", \\"Source\\":
  \\"arn:aws:lambda:us-west-2:123456789012:function:TempMonitor:GG_TempMonitor
  \\", \\"Subject\\": \\"twilio/txt\\", \\"Target\\": \\"arn:aws:greengrass:us-west-2:/:
  connectors/TwilioNotifications/versions/1\\"},{\\"Id\\": \\"TemperatureInput\\", \\"Source
  \\": \\"cloud\\", \\"Subject\\": \\"temperature/input\\", \\"Target\\": \\"arn:aws:lambda:us-
  west-2:123456789012:function:TempMonitor:GG_TempMonitor\\"},{\\"Id\\": \\"OutputStatus
  \\", \\"Source\\": \\"arn:aws:greengrass:us-west-2:/:connectors/TwilioNotifications/
  versions/1\\"},{\\"Subject\\": \\"twilio/message/status\\", \\"Target\\": \\"cloud\\"}]]"
```

Output:

```
{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
  subscriptions/9d611d57-5d5d-44bd-a3b4-feccbdd69112/versions/7b65dfae-50b6-4d0f-
  b3e0-27728bfb0620",
  "CreationTimestamp": "2019-06-24T21:21:33.837Z",
  "Id": "9d611d57-5d5d-44bd-a3b4-feccbdd69112",
  "Version": "7b65dfae-50b6-4d0f-b3e0-27728bfb0620"
}
```

- For API details, see [CreateSubscriptionDefinitionVersion](#) in *AWS CLI Command Reference*.

create-subscription-definition

The following code example shows how to use `create-subscription-definition`.

AWS CLI

To create a subscription definition

The following `create-subscription-definition` example creates a subscription definition and specifies its initial version. The initial version contains three subscriptions: one for the

MQTT topic to which the connector subscribes, one to allow a function to receive temperature readings from AWS IoT, and one to allow AWS IoT to receive status information from the connector. The example provides the ARN for the Lambda function alias that was created earlier by using Lambda's `create-alias` command.

```
aws greengrass create-subscription-definition \
  --initial-version "{\"Subscriptions\": [{\"Id\":
  \"TriggerNotification\", \"Source\": \"arn:aws:lambda:us-
  west-2:123456789012:function:TempMonitor:GG_TempMonitor\", \"Subject\":
  \"twilio/txt\", \"Target\": \"arn:aws:greengrass:us-west-2::/connectors/
  TwilioNotifications/versions/1\"},{\"Id\": \"TemperatureInput\", \"Source\":
  \"cloud\", \"Subject\": \"temperature/input\", \"Target\": \"arn:aws:lambda:us-
  west-2:123456789012:function:TempMonitor:GG_TempMonitor\"},{\"Id\": \"OutputStatus
  \", \"Source\": \"arn:aws:greengrass:us-west-2::/connectors/TwilioNotifications/
  versions/1\", \"Subject\": \"twilio/message/status\", \"Target\": \"cloud\"}]}"
```

Output:

```
{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
  subscriptions/9d611d57-5d5d-44bd-a3b4-feccbdd69112",
  "CreationTimestamp": "2019-06-19T22:34:26.677Z",
  "Id": "9d611d57-5d5d-44bd-a3b4-feccbdd69112",
  "LastUpdatedTimestamp": "2019-06-19T22:34:26.677Z",
  "LatestVersion": "aa645c47-ac90-420d-9091-8c7ffa4f103f",
  "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
  definition/subscriptions/9d611d57-5d5d-44bd-a3b4-feccbdd69112/versions/aa645c47-
  ac90-420d-9091-8c7ffa4f103f"
}
```

For more information, see [Getting Started with Connectors \(CLI\)](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [CreateSubscriptionDefinition](#) in *AWS CLI Command Reference*.

delete-connector-definition

The following code example shows how to use `delete-connector-definition`.

AWS CLI

To delete a connector definition

The following `delete-connector-definition` example deletes the specified Greengrass connector definition. If you delete a connector definition that is used by a group, that group can't be deployed successfully.

```
aws greengrass delete-connector-definition \  
  --connector-definition-id "b5c4ebfd-f672-49a3-83cd-31c7216a7bb8"
```

This command produces no output.

- For API details, see [DeleteConnectorDefinition](#) in *AWS CLI Command Reference*.

delete-core-definition

The following code example shows how to use `delete-core-definition`.

AWS CLI

To delete a core definition

The following `delete-core-definition` example deletes the specified Greengrass core definition, including all versions. If you delete a core that is associated with a Greengrass group, that group can't be deployed successfully.

```
aws greengrass delete-core-definition \  
  --core-definition-id "ff36cc5f-9f98-4994-b468-9d9b6dc52abd"
```

This command produces no output.

- For API details, see [DeleteCoreDefinition](#) in *AWS CLI Command Reference*.

delete-device-definition

The following code example shows how to use `delete-device-definition`.

AWS CLI

To delete a device definition

The following `delete-device-definition` example deletes the specified device definition, including all of its versions. If you delete a device definition version that is used by a group version, the group version cannot be deployed successfully.

```
aws greengrass delete-device-definition \  
  --device-definition-id "f9ba083d-5ad4-4534-9f86-026a45df1ccd"
```

This command produces no output.

- For API details, see [DeleteDeviceDefinition](#) in *AWS CLI Command Reference*.

delete-function-definition

The following code example shows how to use delete-function-definition.

AWS CLI

To delete a function definition

The following delete-function-definition example deletes the specified Greengrass function definition. If you delete a function definition that is used by a group, that group can't be deployed successfully.

```
aws greengrass delete-function-definition \  
  --function-definition-id "fd4b906a-dff3-4c1b-96eb-52ebfcfac06a"
```

This command produces no output.

- For API details, see [DeleteFunctionDefinition](#) in *AWS CLI Command Reference*.

delete-group

The following code example shows how to use delete-group.

AWS CLI

To delete a group

The following delete-group example deletes the specified Greengrass group.

```
aws greengrass delete-group \  
  --group-id "4e22bd92-898c-436b-ade5-434d883ff749"
```

This command produces no output.

- For API details, see [DeleteGroup](#) in *AWS CLI Command Reference*.

delete-logger-definition

The following code example shows how to use `delete-logger-definition`.

AWS CLI

To delete a logger definition

The following `delete-logger-definition` example deletes the specified logger definition, including all logger definition versions. If you delete a logger definition version that is used by a group version, the group version cannot be deployed successfully.

```
aws greengrass delete-logger-definition \  
  --logger-definition-id "a454b62a-5d56-4ca9-bdc4-8254e1662cb0"
```

This command produces no output.

For more information, see [Monitoring with AWS IoT Greengrass Logs](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [DeleteLoggerDefinition](#) in *AWS CLI Command Reference*.

delete-resource-definition

The following code example shows how to use `delete-resource-definition`.

AWS CLI

To delete a resource definition

The following `delete-resource-definition` example deletes the specified resource definition, including all resource versions. If you delete a resource definition that is used by a group, that group can't be deployed successfully.

```
aws greengrass delete-resource-definition \  
  --resource-definition-id "ad8c101d-8109-4b0e-b97d-9cc5802ab658"
```

This command produces no output.

- For API details, see [DeleteResourceDefinition](#) in *AWS CLI Command Reference*.

delete-subscription-definition

The following code example shows how to use `delete-subscription-definition`.

AWS CLI

To delete a subscription definition

The following `delete-subscription-definition` example deletes the specified Greengrass subscription definition. If you delete a subscription that is being used by a group, that group can't be deployed successfully.

```
aws greengrass delete-subscription-definition \  
  --subscription-definition-id "cd6f1c37-d9a4-4e90-be94-01a7404f5967"
```

This command produces no output.

- For API details, see [DeleteSubscriptionDefinition](#) in *AWS CLI Command Reference*.

disassociate-role-from-group

The following code example shows how to use `disassociate-role-from-group`.

AWS CLI

To disassociate the role from a Greengrass group

The following `disassociate-role-from-group` example disassociates the IAM role from the specified Greengrass group.

```
aws greengrass disassociate-role-from-group \  
  --group-id 2494ee3f-7f8a-4e92-a78b-d205f808b84b
```

Output:

```
{  
  "DisassociatedAt": "2019-09-10T20:05:49Z"
```

```
}
```

For more information, see [Configure the Group Role](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [DisassociateRoleFromGroup](#) in *AWS CLI Command Reference*.

disassociate-service-role-from-account

The following code example shows how to use `disassociate-service-role-from-account`.

AWS CLI

To disassociate a service role from your AWS account

The following `disassociate-service-role-from-account` example removes the service role that is associated with your AWS account. If you are not using the service role in any AWS Region, use the `delete-role-policy` command to detach the `AWSGreengrassResourceAccessRolePolicy` managed policy from the role, and then use the `delete-role` command to delete the role.

```
aws greengrass disassociate-service-role-from-account
```

Output:

```
{
  "DisassociatedAt": "2019-06-25T22:12:55Z"
}
```

For more information, see [Greengrass Service Role](#) in the **AWS IoT Greengrass Developer Guide**.

- For API details, see [DisassociateServiceRoleFromAccount](#) in *AWS CLI Command Reference*.

get-associated-role

The following code example shows how to use `get-associated-role`.

AWS CLI

To get the role associated with a Greengrass group

The following `get-associated-role` example gets the IAM role that's associated with the specified Greengrass group. The group role is used by local Lambda functions and connectors to access AWS services.

```
aws greengrass get-associated-role \  
  --group-id 2494ee3f-7f8a-4e92-a78b-d205f808b84b
```

Output:

```
{  
  "RoleArn": "arn:aws:iam::123456789012:role/GG-Group-Role",  
  "AssociatedAt": "2019-09-10T20:03:30Z"  
}
```

For more information, see [Configure the Group Role](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [GetAssociatedRole](#) in *AWS CLI Command Reference*.

get-bulk-deployment-status

The following code example shows how to use `get-bulk-deployment-status`.

AWS CLI

To check the status of your bulk deployment

The following `get-bulk-deployment-status` example retrieves status information for the specified bulk deployment operation. In this example, the file that specified the groups to be deployed has an invalid input record.

```
aws greengrass get-bulk-deployment-status \  
  --bulk-deployment-id "870fb41b-6288-4e0c-bc76-a7ba4b4d3267"
```

Output:

```
{  
  "BulkDeploymentMetrics": {  
    "InvalidInputRecords": 1,  
    "RecordsProcessed": 1,  
    "RetryAttempts": 0  
  },  
}
```

```
"BulkDeploymentStatus": "Completed",
"CreatedAt": "2019-06-25T16:11:33.265Z",
"tags": {}
}
```

For more information, see [Create Bulk Deployments for Groups](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [GetBulkDeploymentStatus](#) in *AWS CLI Command Reference*.

get-connectivity-info

The following code example shows how to use `get-connectivity-info`.

AWS CLI

To get the connectivity information for a Greengrass core

The following `get-connectivity-info` example displays the endpoints that devices can use to connect to the specified Greengrass core. Connectivity information is a list of IP addresses or domain names, with corresponding port numbers and optional customer-defined metadata.

```
aws greengrass get-connectivity-info \
  --thing-name "MyGroup_Core"
```

Output:

```
{
  "ConnectivityInfo": [
    {
      "Metadata": "",
      "PortNumber": 8883,
      "HostAddress": "127.0.0.1",
      "Id": "AUTOIP_127.0.0.1_0"
    },
    {
      "Metadata": "",
      "PortNumber": 8883,
      "HostAddress": "192.168.1.3",
      "Id": "AUTOIP_192.168.1.3_1"
    },
    {
```

```

        "Metadata": "",
        "PortNumber": 8883,
        "HostAddress": ":::1",
        "Id": "AUTOIP_:::1_2"
    },
    {
        "Metadata": "",
        "PortNumber": 8883,
        "HostAddress": "fe80::1e69:ed93:f5b:f6d",
        "Id": "AUTOIP_fe80::1e69:ed93:f5b:f6d_3"
    }
]
}

```

- For API details, see [GetConnectivityInfo](#) in *AWS CLI Command Reference*.

get-connector-definition-version

The following code example shows how to use `get-connector-definition-version`.

AWS CLI

To retrieve information about a specific version of a connector definition

The following `get-connector-definition-version` example retrieves information about the specified version of the specified connector definition. To retrieve the IDs of all versions of the connector definition, use the `list-connector-definition-versions` command. To retrieve the ID of the last version added to the connector definition, use the `get-connector-definition` command and check the `LatestVersion` property.

```

aws greengrass get-connector-definition-version \
  --connector-definition-id "b5c4ebfd-f672-49a3-83cd-31c7216a7bb8" \
  --connector-definition-version-id "63c57963-c7c2-4a26-a7e2-7bf478ea2623"

```

Output:

```

{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
connectors/b5c4ebfd-f672-49a3-83cd-31c7216a7bb8/versions/63c57963-c7c2-4a26-
a7e2-7bf478ea2623",
  "CreationTimestamp": "2019-06-19T19:30:01.300Z",

```

```

    "Definition": {
      "Connectors": [
        {
          "ConnectorArn": "arn:aws:greengrass:us-west-2::/connectors/SNS/
versions/1",
          "Id": "MySNSConnector",
          "Parameters": {
            "DefaultSNSArn": "arn:aws:sns:us-
west-2:123456789012:GGConnectorTopic"
          }
        }
      ]
    },
    "Id": "b5c4ebfd-f672-49a3-83cd-31c7216a7bb8",
    "Version": "63c57963-c7c2-4a26-a7e2-7bf478ea2623"
  }
}

```

For more information, see [Integrate with Services and Protocols Using Greengrass Connectors](#) in the **AWS IoT Greengrass Developer Guide**.

- For API details, see [GetConnectorDefinitionVersion](#) in *AWS CLI Command Reference*.

get-connector-definition

The following code example shows how to use `get-connector-definition`.

AWS CLI

To retrieve information about a connector definition

The following `get-connector-definition` example retrieves information about the specified connector definition. To retrieve the IDs of your connector definitions, use the `list-connector-definitions` command.

```

aws greengrass get-connector-definition \
  --connector-definition-id "b5c4ebfd-f672-49a3-83cd-31c7216a7bb8"

```

Output:

```

{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
connectors/b5c4ebfd-f672-49a3-83cd-31c7216a7bb8",

```

```

    "CreationTimestamp": "2019-06-19T19:30:01.300Z",
    "Id": "b5c4ebfd-f672-49a3-83cd-31c7216a7bb8",
    "LastUpdatedTimestamp": "2019-06-19T19:30:01.300Z",
    "LatestVersion": "63c57963-c7c2-4a26-a7e2-7bf478ea2623",
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/connectors/b5c4ebfd-f672-49a3-83cd-31c7216a7bb8/versions/63c57963-
c7c2-4a26-a7e2-7bf478ea2623",
    "Name": "MySNSConnector",
    "tags": {}
}

```

For more information, see [Integrate with Services and Protocols Using Greengrass Connectors](#) in the **AWS IoT Greengrass Developer Guide**.

- For API details, see [GetConnectorDefinition](#) in *AWS CLI Command Reference*.

get-core-definition-version

The following code example shows how to use `get-core-definition-version`.

AWS CLI

To retrieve details about a specific version of the Greengrass core definition

The following `get-core-definition-version` example retrieves information about the specified version of the specified core definition. To retrieve the IDs of all versions of the core definition, use the `list-core-definition-versions` command. To retrieve the ID of the last version added to the core definition, use the `get-core-definition` command and check the `LatestVersion` property.

```

aws greengrass get-core-definition-version \
  --core-definition-id "c906ed39-a1e3-4822-a981-7b9bd57b4b46" \
  --core-definition-version-id "42aeeac3-fd9d-4312-a8fd-ffa9404a20e0"

```

Output:

```

{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/cores/
c906ed39-a1e3-4822-a981-7b9bd57b4b46/versions/42aeeac3-fd9d-4312-a8fd-ffa9404a20e0",
  "CreationTimestamp": "2019-06-18T16:21:21.351Z",
  "Definition": {

```

```

    "Cores": [
      {
        "CertificateArn": "arn:aws:iot:us-
west-2:123456789012:cert/928dea7b82331b47c3ff77b0e763fc5e64e2f7c884e6ef391baed9b6b8e21b45",
        "Id": "1a39aac7-0885-4417-91f6-23e4cea6c511",
        "SyncShadow": false,
        "ThingArn": "arn:aws:iot:us-west-2:123456789012:thing/
GGGroup4Pi3_Core"
      }
    ],
    "Id": "c906ed39-a1e3-4822-a981-7b9bd57b4b46",
    "Version": "42aeac3-fd9d-4312-a8fd-ffa9404a20e0"
  }

```

- For API details, see [GetCoreDefinitionVersion](#) in *AWS CLI Command Reference*.

get-core-definition

The following code example shows how to use `get-core-definition`.

AWS CLI

To retrieve details for a Greengrass core definition

The following `get-core-definition` example retrieves information about the specified core definition. To retrieve the IDs of your core definitions, use the `list-core-definitions` command.

```

aws greengrass get-core-definition \
  --core-definition-id "c906ed39-a1e3-4822-a981-7b9bd57b4b46"

```

Output:

```

{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
cores/237d6916-27cf-457f-ba0c-e86cfb5d25cd",
  "CreationTimestamp": "2018-10-18T04:47:06.721Z",
  "Id": "237d6916-27cf-457f-ba0c-e86cfb5d25cd",
  "LastUpdatedTimestamp": "2018-10-18T04:47:06.721Z",
  "LatestVersion": "bd2cd6d4-2bc5-468a-8962-39e071e34b68",

```

```
"LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/cores/237d6916-27cf-457f-ba0c-e86cfb5d25cd/versions/bd2cd6d4-2bc5-468a-8962-39e071e34b68",
  "tags": {}
}
```

- For API details, see [GetCoreDefinition](#) in *AWS CLI Command Reference*.

get-deployment-status

The following code example shows how to use `get-deployment-status`.

AWS CLI

To retrieve the status of a deployment

The following `get-deployment-status` example retrieves the status for the specified deployment of the specified Greengrass group. To get the deployment ID, use the `list-deployments` command and specify the group ID.

```
aws greengrass get-deployment-status \
  --group-id "1013db12-8b58-45ff-acc7-704248f66731" \
  --deployment-id "1065b8a0-812b-4f21-9d5d-e89b232a530f"
```

Output:

```
{
  "DeploymentStatus": "Success",
  "DeploymentType": "NewDeployment",
  "UpdatedAt": "2019-06-18T17:04:44.761Z"
}
```

- For API details, see [GetDeploymentStatus](#) in *AWS CLI Command Reference*.

get-device-definition-version

The following code example shows how to use `get-device-definition-version`.

AWS CLI

To get a device definition version

The following `get-device-definition-version` example retrieves information about the specified version of the specified device definition. To retrieve the IDs of all versions of the device definition, use the `list-device-definition-versions` command. To retrieve the ID of the last version added to the device definition, use the `get-device-definition` command and check the `LatestVersion` property.

```
aws greengrass get-device-definition-version \
  --device-definition-id "f9ba083d-5ad4-4534-9f86-026a45df1ccd" \
  --device-definition-version-id "83c13984-6fed-447e-84d5-5b8aa45d5f71"
```

Output:

```
{
  "Definition": {
    "Devices": [
      {
        "CertificateArn": "arn:aws:iot:us-west-2:123456789012:cert/6c52ce1b47bde88a637e9ccdd45fe4e4c2c0a75a6866f8f63d980ee22fa51e02",
        "ThingArn": "arn:aws:iot:us-west-2:123456789012:thing/ExteriorTherm",
        "SyncShadow": true,
        "Id": "ExteriorTherm"
      },
      {
        "CertificateArn": "arn:aws:iot:us-west-2:123456789012:cert/66a415ec415668c2349a76170b64ac0878231c1e21ec83c10e92a18bd568eb92",
        "ThingArn": "arn:aws:iot:us-west-2:123456789012:thing/InteriorTherm",
        "SyncShadow": true,
        "Id": "InteriorTherm"
      }
    ]
  },
  "Version": "83c13984-6fed-447e-84d5-5b8aa45d5f71",
  "CreationTimestamp": "2019-09-11T00:15:09.838Z",
  "Id": "f9ba083d-5ad4-4534-9f86-026a45df1ccd",
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd/versions/83c13984-6fed-447e-84d5-5b8aa45d5f71"
}
```

- For API details, see [GetDeviceDefinitionVersion](#) in *AWS CLI Command Reference*.

get-device-definition

The following code example shows how to use `get-device-definition`.

AWS CLI

To get a device definition

The following `get-device-definition` example retrieves information about the specified device definition. To retrieve the IDs of your device definitions, use the `list-device-definitions` command.

```
aws greengrass get-device-definition \  
  --device-definition-id "f9ba083d-5ad4-4534-9f86-026a45df1ccd"
```

Output:

```
{  
  "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd/versions/83c13984-6fed-447e-84d5-5b8aa45d5f71",  
  "Name": "TemperatureSensors",  
  "tags": {},  
  "LastUpdatedTimestamp": "2019-09-11T00:19:03.698Z",  
  "LatestVersion": "83c13984-6fed-447e-84d5-5b8aa45d5f71",  
  "CreationTimestamp": "2019-09-11T00:11:06.197Z",  
  "Id": "f9ba083d-5ad4-4534-9f86-026a45df1ccd",  
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd"  
}
```

- For API details, see [GetDeviceDefinition](#) in *AWS CLI Command Reference*.

get-function-definition-version

The following code example shows how to use `get-function-definition-version`.

AWS CLI

To retrieve details about a specific version of a Lambda function

The following `get-function-definition-version` retrieves information about the specified version of the specified function definition. To retrieve the IDs of all versions of the function definition, use the `list-function-definition-versions` command. To retrieve the ID of the last version added to the function definition, use the `get-function-definition` command and check the `LatestVersion` property.

```
aws greengrass get-function-definition-version \  
  --function-definition-id "063f5d1a-1dd1-40b4-9b51-56f8993d0f85" \  
  --function-definition-version-id "9748fda7-1589-4fcc-ac94-f5559e88678b"
```

Output:

```
{  
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/  
functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85/versions/9748fda7-1589-4fcc-ac94-  
f5559e88678b",  
  "CreationTimestamp": "2019-06-18T17:04:30.776Z",  
  "Definition": {  
    "Functions": [  
      {  
        "FunctionArn": "arn:aws:lambda::function:GGIPDetector:1",  
        "FunctionConfiguration": {  
          "Environment": {},  
          "MemorySize": 32768,  
          "Pinned": true,  
          "Timeout": 3  
        },  
        "Id": "26b69bdb-e547-46bc-9812-84ec04b6cc8c"  
      },  
      {  
        "FunctionArn": "arn:aws:lambda:us-  
west-2:123456789012:function:Greengrass_HelloWorld:GG_HelloWorld",  
        "FunctionConfiguration": {  
          "EncodingType": "json",  
          "Environment": {  
            "Variables": {}  
          },  
          "MemorySize": 16384,  
          "Pinned": true,  
          "Timeout": 25  
        },  
        "Id": "384465a8-eeef-48c6-b793-4c35f7bfae9b"  
      }  
    ]  
  }  
}
```

```

    }
  ]
},
"Id": "063f5d1a-1dd1-40b4-9b51-56f8993d0f85",
"Version": "9748fda7-1589-4fcc-ac94-f5559e88678b"
}

```

- For API details, see [GetFunctionDefinitionVersion](#) in *AWS CLI Command Reference*.

get-function-definition

The following code example shows how to use `get-function-definition`.

AWS CLI

To retrieve a function definition

The following `get-function-definition` example displays details for the specified function definition. To retrieve the IDs of your function definitions, use the `list-function-definitions` command.

```

aws greengrass get-function-definition \
  --function-definition-id "063f5d1a-1dd1-40b4-9b51-56f8993d0f85"

```

Output:

```

{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85",
  "CreationTimestamp": "2019-06-18T16:21:21.431Z",
  "Id": "063f5d1a-1dd1-40b4-9b51-56f8993d0f85",
  "LastUpdatedTimestamp": "2019-06-18T16:21:21.431Z",
  "LatestVersion": "9748fda7-1589-4fcc-ac94-f5559e88678b",
  "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85/
versions/9748fda7-1589-4fcc-ac94-f5559e88678b",
  "tags": {}
}

```

- For API details, see [GetFunctionDefinition](#) in *AWS CLI Command Reference*.

get-group-certificate-authority

The following code example shows how to use `get-group-certificate-authority`.

AWS CLI

To retrieve the CA associated with a Greengrass group

The following `get-group-certificate-authority` example retrieves the certificate authority (CA) that is associated with the specified Greengrass group. To get the certificate authority ID, use the `list-group-certificate-authorities` command and specify the group ID.

```
aws greengrass get-group-certificate-authority \
  --group-id "1013db12-8b58-45ff-acc7-704248f66731" \
  --certificate-authority-id
  "f0430e1736ea8ed30cc5d5de9af67a7e3586bad9ae4d89c2a44163f65fdd8cf6"
```

Output:

```
{
  "GroupCertificateAuthorityArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/groups/1013db12-8b58-45ff-acc7-704248f66731/certificateauthorities/f0430e1736ea8ed30cc5d5de9af67a7e3586bad9ae4d89c2a44163f65fdd8cf6",
  "GroupCertificateAuthorityId":
  "f0430e1736ea8ed30cc5d5de9af67a7e3586bad9ae4d89c2a44163f65fdd8cf6",
  "PemEncodedCertificate": "-----BEGIN CERTIFICATE-----
MIICiTCCAFICCD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBWEXAMPLEGA1UEBhMC
VVMxCzAJBgNVBAGTAldBMRAwDEXAMPLEEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAEXAMPLESDBb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWVxMzYz
BgkqhkiG9w0BCQEWEG5vb251QGftYXpvbi5jEXAMPLENMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0EXAMPLEBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYD
VQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWEXAMPLEDASBgNVBAwTC01BTSDBb25z
b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWVxMzYzEXAMPLEEgkqhkiG9w0BCQEWEG5vb251QGft
YXpvbi5EXAMPLE8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLyGVIk60CEXAMPLE93vUEI03IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswYEXAMPLEEgpE
Ibb30hjZnzcvcQAaRHhd1QWIMm2nrAgMBAAEwDQYJKEXAMPLEAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJ10ZxBHjJnyp3780D8uTs7fLvJx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----\n"
```

```
}
```

- For API details, see [GetGroupCertificateAuthority](#) in *AWS CLI Command Reference*.

get-group-certificate-configuration

The following code example shows how to use `get-group-certificate-configuration`.

AWS CLI

To retrieve the configuration for the certificate authority used by the Greengrass group

The following `get-group-certificate-configuration` example retrieves the configuration for the certificate authority (CA) used by the specified Greengrass group.

```
aws greengrass get-group-certificate-configuration \  
  --group-id "1013db12-8b58-45ff-acc7-704248f66731"
```

Output:

```
{  
  "CertificateAuthorityExpiryInMilliseconds": 2524607999000,  
  "CertificateExpiryInMilliseconds": 604800000,  
  "GroupId": "1013db12-8b58-45ff-acc7-704248f66731"  
}
```

- For API details, see [GetGroupCertificateConfiguration](#) in *AWS CLI Command Reference*.

get-group-version

The following code example shows how to use `get-group-version`.

AWS CLI

To retrieve information about a version of a Greengrass group

The following `get-group-version` example retrieves information about the specified version of the specified group. To retrieve the IDs of all versions of the group, use the `list-group-versions` command. To retrieve the ID of the last version added to the group, use the `get-group` command and check the `LatestVersion` property.

```
aws greengrass get-group-version \  
  --group-id "1013db12-8b58-45ff-acc7-704248f66731" \  
  --group-version-id "115136b3-cfd7-4462-b77f-8741a4b00e5e"
```

Output:

```
{  
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/  
groups/1013db12-8b58-45ff-acc7-704248f66731/versions/115136b3-cfd7-4462-  
b77f-8741a4b00e5e",  
  "CreationTimestamp": "2019-06-18T17:04:30.915Z",  
  "Definition": {  
    "CoreDefinitionVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/  
greengrass/definition/cores/c906ed39-a1e3-4822-a981-7b9bd57b4b46/versions/42aeeac3-  
fd9d-4312-a8fd-ffa9404a20e0",  
    "FunctionDefinitionVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/  
greengrass/definition/functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85/  
versions/9748fda7-1589-4fcc-ac94-f5559e88678b",  
    "SubscriptionDefinitionVersionArn": "arn:aws:greengrass:us-  
west-2:123456789012:/greengrass/definition/subscriptions/70e49321-83d5-45d2-  
bc09-81f4917ae152/versions/88ae8699-12ac-4663-ba3f-4d7f0519140b"  
  },  
  "Id": "1013db12-8b58-45ff-acc7-704248f66731",  
  "Version": "115136b3-cfd7-4462-b77f-8741a4b00e5e"  
}
```

- For API details, see [GetGroupVersion](#) in *AWS CLI Command Reference*.

get-group

The following code example shows how to use `get-group`.

AWS CLI

To retrieve information about a Greengrass group

The following `get-group` example retrieves information about the specified Greengrass group. To retrieve the IDs of your groups, use the `list-groups` command.

```
aws greengrass get-group \  
  --group-id "1013db12-8b58-45ff-acc7-704248f66731"
```

Output:

```
{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/1013db12-8b58-45ff-acc7-704248f66731",
  "CreationTimestamp": "2019-06-18T16:21:21.457Z",
  "Id": "1013db12-8b58-45ff-acc7-704248f66731",
  "LastUpdatedTimestamp": "2019-06-18T16:21:21.457Z",
  "LatestVersion": "115136b3-cfd7-4462-b77f-8741a4b00e5e",
  "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/1013db12-8b58-45ff-acc7-704248f66731/versions/115136b3-cfd7-4462-
b77f-8741a4b00e5e",
  "Name": "GGGroup4Pi3",
  "tags": {}
}
```

- For API details, see [GetGroup](#) in *AWS CLI Command Reference*.

get-logger-definition-version

The following code example shows how to use `get-logger-definition-version`.

AWS CLI**To retrieve information about a version of a logger definition**

The following `get-logger-definition-version` example retrieves information about the specified version of the specified logger definition. To retrieve the IDs of all versions of the logger definition, use the `list-logger-definition-versions` command. To retrieve the ID of the last version added to the logger definition, use the `get-logger-definition` command and check the `LatestVersion` property.

```
aws greengrass get-logger-definition-version \
  --logger-definition-id "49eeeb66-f1d3-4e34-86e3-3617262abf23" \
  --logger-definition-version-id "5e3f6f64-a565-491e-8de0-3c0d8e0f2073"
```

Output:

```
{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/loggers/49eeeb66-f1d3-4e34-86e3-3617262abf23/versions/5e3f6f64-
a565-491e-8de0-3c0d8e0f2073",
```

```
"CreationTimestamp": "2019-05-08T16:10:13.866Z",
"Definition": {
  "Loggers": []
},
"Id": "49eeeb66-f1d3-4e34-86e3-3617262abf23",
"Version": "5e3f6f64-a565-491e-8de0-3c0d8e0f2073"
}
```

- For API details, see [GetLoggerDefinitionVersion](#) in *AWS CLI Command Reference*.

get-logger-definition

The following code example shows how to use `get-logger-definition`.

AWS CLI

To retrieve information about a logger definition

The following `get-logger-definition` example retrieves information about the specified logger definition. To retrieve the IDs of your logger definitions, use the `list-logger-definitions` command.

```
aws greengrass get-logger-definition \
  --logger-definition-id "49eeeb66-f1d3-4e34-86e3-3617262abf23"
```

Output:

```
{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
loggers/49eeeb66-f1d3-4e34-86e3-3617262abf23",
  "CreationTimestamp": "2019-05-08T16:10:13.809Z",
  "Id": "49eeeb66-f1d3-4e34-86e3-3617262abf23",
  "LastUpdatedTimestamp": "2019-05-08T16:10:13.809Z",
  "LatestVersion": "5e3f6f64-a565-491e-8de0-3c0d8e0f2073",
  "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/loggers/49eeeb66-f1d3-4e34-86e3-3617262abf23/versions/5e3f6f64-
a565-491e-8de0-3c0d8e0f2073",
  "tags": {}
}
```

- For API details, see [GetLoggerDefinition](#) in *AWS CLI Command Reference*.

get-resource-definition-version

The following code example shows how to use `get-resource-definition-version`.

AWS CLI

To retrieve information about a specific version of a resource definition

The following `get-resource-definition-version` example retrieves information about the specified version of the specified resource definition. To retrieve the IDs of all versions of the resource definition, use the `list-resource-definition-versions` command. To retrieve the ID of the last version added to the resource definition, use the `get-resource-definition` command and check the `LatestVersion` property.

```
aws greengrass get-resource-definition-version \
  --resource-definition-id "ad8c101d-8109-4b0e-b97d-9cc5802ab658" \
  --resource-definition-version-id "26e8829a-491a-464d-9c87-664bf6f6f2be"
```

Output:

```
{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/resources/ad8c101d-8109-4b0e-b97d-9cc5802ab658/
versions/26e8829a-491a-464d-9c87-664bf6f6f2be",
  "CreationTimestamp": "2019-06-19T16:40:59.392Z",
  "Definition": {
    "Resources": [
      {
        "Id": "26ff3f7b-839a-4217-9fdc-a218308b3963",
        "Name": "usb-port",
        "ResourceDataContainer": {
          "LocalDeviceResourceData": {
            "GroupOwnerSetting": {
              "AutoAddGroupOwner": false
            },
            "SourcePath": "/dev/bus/usb"
          }
        }
      }
    ]
  },
  "Id": "ad8c101d-8109-4b0e-b97d-9cc5802ab658",
```

```
"Version": "26e8829a-491a-464d-9c87-664bf6f6f2be"
}
```

- For API details, see [GetResourceDefinitionVersion](#) in *AWS CLI Command Reference*.

get-resource-definition

The following code example shows how to use `get-resource-definition`.

AWS CLI

To retrieve information about a resource definition

The following `get-resource-definition` example retrieves information about the specified resource definition. To retrieve the IDs of your resource definitions, use the `list-resource-definitions` command.

```
aws greengrass get-resource-definition \
  --resource-definition-id "ad8c101d-8109-4b0e-b97d-9cc5802ab658"
```

Output:

```
{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
resources/ad8c101d-8109-4b0e-b97d-9cc5802ab658",
  "CreationTimestamp": "2019-06-19T16:40:59.261Z",
  "Id": "ad8c101d-8109-4b0e-b97d-9cc5802ab658",
  "LastUpdatedTimestamp": "2019-06-19T16:40:59.261Z",
  "LatestVersion": "26e8829a-491a-464d-9c87-664bf6f6f2be",
  "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/resources/ad8c101d-8109-4b0e-b97d-9cc5802ab658/
versions/26e8829a-491a-464d-9c87-664bf6f6f2be",
  "tags": {}
}
```

- For API details, see [GetResourceDefinition](#) in *AWS CLI Command Reference*.

get-service-role-for-account

The following code example shows how to use `get-service-role-for-account`.

AWS CLI

To retrieve the details for the service role that is attached to your account

The following `get-service-role-for-account` example retrieves information about the service role that is attached to your AWS account.

```
aws greengrass get-service-role-for-account
```

Output:

```
{
  "AssociatedAt": "2018-10-18T15:59:20Z",
  "RoleArn": "arn:aws:iam::123456789012:role/service-role/Greengrass_ServiceRole"
}
```

For more information, see [Greengrass Service Role](#) in the **AWS IoT Greengrass Developer Guide**.

- For API details, see [GetServiceRoleForAccount](#) in *AWS CLI Command Reference*.

get-subscription-definition-version

The following code example shows how to use `get-subscription-definition-version`.

AWS CLI

To retrieve information about a specific version of a subscription definition

The following `get-subscription-definition-version` example retrieves information about the specified version of the specified subscription definition. To retrieve the IDs of all versions of the subscription definition, use the `list-subscription-definition-versions` command. To retrieve the ID of the last version added to the subscription definition, use the `get-subscription-definition` command and check the `LatestVersion` property.

```
aws greengrass get-subscription-definition-version \
  --subscription-definition-id "70e49321-83d5-45d2-bc09-81f4917ae152" \
  --subscription-definition-version-id "88ae8699-12ac-4663-ba3f-4d7f0519140b"
```

Output:

```
{
  "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
subscriptions/70e49321-83d5-45d2-bc09-81f4917ae152/versions/88ae8699-12ac-4663-
ba3f-4d7f0519140b",
  "CreationTimestamp": "2019-06-18T17:03:52.499Z",
  "Definition": {
    "Subscriptions": [
      {
        "Id": "692c4484-d89f-4f64-8edd-1a041a65e5b6",
        "Source": "arn:aws:lambda:us-
west-2:123456789012:function:Greengrass_HelloWorld:GG_HelloWorld",
        "Subject": "hello/world",
        "Target": "cloud"
      }
    ]
  },
  "Id": "70e49321-83d5-45d2-bc09-81f4917ae152",
  "Version": "88ae8699-12ac-4663-ba3f-4d7f0519140b"
}
```

- For API details, see [GetSubscriptionDefinitionVersion](#) in *AWS CLI Command Reference*.

get-subscription-definition

The following code example shows how to use `get-subscription-definition`.

AWS CLI**To retrieve information about a subscription definition**

The following `get-subscription-definition` example retrieves information about the specified subscription definition. To retrieve the IDs of your subscription definitions, use the `list-subscription-definitions` command.

```
aws greengrass get-subscription-definition \
  --subscription-definition-id "70e49321-83d5-45d2-bc09-81f4917ae152"
```

Output:

```
{
```

```

    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/
subscriptions/70e49321-83d5-45d2-bc09-81f4917ae152",
    "CreationTimestamp": "2019-06-18T17:03:52.392Z",
    "Id": "70e49321-83d5-45d2-bc09-81f4917ae152",
    "LastUpdatedTimestamp": "2019-06-18T17:03:52.392Z",
    "LatestVersion": "88ae8699-12ac-4663-ba3f-4d7f0519140b",
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/subscriptions/70e49321-83d5-45d2-bc09-81f4917ae152/
versions/88ae8699-12ac-4663-ba3f-4d7f0519140b",
    "tags": {}
}

```

- For API details, see [GetSubscriptionDefinition](#) in *AWS CLI Command Reference*.

get-thing-runtime-configuration

The following code example shows how to use `get-thing-runtime-configuration`.

AWS CLI

To retrieve the runtime configuration of a Greengrass core

The following `get-thing-runtime-configuration` example retrieves the runtime configuration of a Greengrass core. Before you can retrieve the runtime configuration, you must use the `update-thing-runtime-configuration` command to create a runtime configuration for the core.

```

aws greengrass get-thing-runtime-configuration \
  --thing-name SampleGreengrassCore

```

Output:

```

{
  "RuntimeConfiguration": {
    "TelemetryConfiguration": {
      "ConfigurationSyncStatus": "OutOfSync",
      "Telemetry": "On"
    }
  }
}

```

For more information, see [Configuring telemetry settings](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [GetThingRuntimeConfiguration](#) in *AWS CLI Command Reference*.

list-bulk-deployment-detailed-reports

The following code example shows how to use `list-bulk-deployment-detailed-reports`.

AWS CLI

To list information about individual deployments in a bulk deployment

The following `list-bulk-deployment-detailed-reports` example displays information about the individual deployments in a bulk deployment operation, including status.

```
aws greengrass list-bulk-deployment-detailed-reports \
  --bulk-deployment-id 42ce9c42-489b-4ed4-b905-8996aa50ef9d
```

Output:

```
{
  "Deployments": [
    {
      "DeploymentType": "NewDeployment",
      "DeploymentStatus": "Success",
      "DeploymentId": "123456789012:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "DeploymentArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/a1b2c3d4-5678-90ab-cdef-EXAMPLE33333/
deployments/123456789012:123456789012:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "GroupArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/a1b2c3d4-5678-90ab-cdef-EXAMPLE33333/
versions/123456789012:a1b2c3d4-5678-90ab-cdef-EXAMPLE44444",
      "CreatedAt": "2020-01-21T21:34:16.501Z"
    },
    {
      "DeploymentType": "NewDeployment",
      "DeploymentStatus": "InProgress",
      "DeploymentId": "123456789012:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
      "DeploymentArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/a1b2c3d4-5678-90ab-cdef-EXAMPLE55555/
deployments/123456789012:123456789012:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
```

```
        "GroupArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/a1b2c3d4-5678-90ab-cdef-EXAMPLE55555/versions/a1b2c3d4-5678-90ab-cdef-
EXAMPLE66666",
        "CreatedAt": "2020-01-21T21:34:16.486Z"
    },
    ...
]
}
```

For more information, see [Create Bulk Deployments for Groups](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [ListBulkDeploymentDetailedReports](#) in *AWS CLI Command Reference*.

list-bulk-deployments

The following code example shows how to use `list-bulk-deployments`.

AWS CLI

To list bulk deployments

The following `list-bulk-deployments` example lists all bulk deployments.

```
aws greengrass list-bulk-deployments
```

Output:

```
{
  "BulkDeployments": [
    {
      "BulkDeploymentArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/bulk/deployments/870fb41b-6288-4e0c-bc76-a7ba4b4d3267",
      "BulkDeploymentId": "870fb41b-6288-4e0c-bc76-a7ba4b4d3267",
      "CreatedAt": "2019-06-25T16:11:33.265Z"
    }
  ]
}
```

For more information, see [Create Bulk Deployments for Groups](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [ListBulkDeployments](#) in *AWS CLI Command Reference*.

list-connector-definition-versions

The following code example shows how to use `list-connector-definition-versions`.

AWS CLI

To list the versions that are available for a connector definition

The following `list-connector-definition-versions` example lists the versions that are available for the specified connector definition. Use the `list-connector-definitions` command to get the connector definition ID.

```
aws greengrass list-connector-definition-versions \  
  --connector-definition-id "b5c4ebfd-f672-49a3-83cd-31c7216a7bb8"
```

Output:

```
{  
  "Versions": [  
    {  
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/  
definition/connectors/b5c4ebfd-f672-49a3-83cd-31c7216a7bb8/versions/63c57963-  
c7c2-4a26-a7e2-7bf478ea2623",  
      "CreationTimestamp": "2019-06-19T19:30:01.300Z",  
      "Id": "b5c4ebfd-f672-49a3-83cd-31c7216a7bb8",  
      "Version": "63c57963-c7c2-4a26-a7e2-7bf478ea2623"  
    }  
  ]  
}
```

For more information, see [Integrate with Services and Protocols Using Greengrass Connectors](#) in the **AWS IoT Greengrass Developer Guide**.

- For API details, see [ListConnectorDefinitionVersions](#) in *AWS CLI Command Reference*.

list-connector-definitions

The following code example shows how to use `list-connector-definitions`.

AWS CLI

To list the Greengrass connectors that are defined

The following `list-connector-definitions` example lists all of the Greengrass connectors that are defined for your AWS account.

```
aws greengrass list-connector-definitions
```

Output:

```
{
  "Definitions": [
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/connectors/b5c4ebfd-f672-49a3-83cd-31c7216a7bb8",
      "CreationTimestamp": "2019-06-19T19:30:01.300Z",
      "Id": "b5c4ebfd-f672-49a3-83cd-31c7216a7bb8",
      "LastUpdatedTimestamp": "2019-06-19T19:30:01.300Z",
      "LatestVersion": "63c57963-c7c2-4a26-a7e2-7bf478ea2623",
      "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/connectors/b5c4ebfd-f672-49a3-83cd-31c7216a7bb8/
versions/63c57963-c7c2-4a26-a7e2-7bf478ea2623",
      "Name": "MySNSConnector"
    }
  ]
}
```

For more information, see [Integrate with Services and Protocols Using Greengrass Connectors](#) in the **AWS IoT Greengrass Developer Guide**.

- For API details, see [ListConnectorDefinitions](#) in *AWS CLI Command Reference*.

list-core-definition-versions

The following code example shows how to use `list-core-definition-versions`.

AWS CLI

To list the versions of a Greengrass core definition

The following `list-core-definitions` example lists all versions of the specified Greengrass core definition. You can use the `list-core-definitions` command to get the version ID.

```
aws greengrass list-core-definition-versions \
```

```
--core-definition-id "eaf280cb-138c-4d15-af36-6f681a1348f7"
```

Output:

```
{
  "Versions": [
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/cores/eaf280cb-138c-4d15-af36-6f681a1348f7/versions/467c36e4-c5da-440c-
a97b-084e62593b4c",
      "CreationTimestamp": "2019-06-18T16:14:17.709Z",
      "Id": "eaf280cb-138c-4d15-af36-6f681a1348f7",
      "Version": "467c36e4-c5da-440c-a97b-084e62593b4c"
    }
  ]
}
```

- For API details, see [ListCoreDefinitionVersions](#) in *AWS CLI Command Reference*.

list-core-definitions

The following code example shows how to use `list-core-definitions`.

AWS CLI

To list Greengrass core definitions

The following `list-core-definitions` example lists all of the Greengrass core definitions for your AWS account.

```
aws greengrass list-core-definitions
```

Output:

```
{
  "Definitions": [
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/cores/0507843c-c1ef-4f06-b051-817030df7e7d",
      "CreationTimestamp": "2018-10-17T04:30:32.786Z",
```

```

    "Id": "0507843c-c1ef-4f06-b051-817030df7e7d",
    "LastUpdatedTimestamp": "2018-10-17T04:30:32.786Z",
    "LatestVersion": "bcd9e86-3793-491e-93af-3cdfbf4e22b7",
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/cores/0507843c-c1ef-4f06-b051-817030df7e7d/versions/bcd9e86-3793-491e-93af-3cdfbf4e22b7"
  },
  {
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/cores/31c22500-3509-4271-bafd-cf0655cda438",
    "CreationTimestamp": "2019-06-18T16:24:16.064Z",
    "Id": "31c22500-3509-4271-bafd-cf0655cda438",
    "LastUpdatedTimestamp": "2019-06-18T16:24:16.064Z",
    "LatestVersion": "2f350395-6d09-4c8a-8336-9ae5b57ace84",
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/cores/31c22500-3509-4271-bafd-cf0655cda438/versions/2f350395-6d09-4c8a-8336-9ae5b57ace84"
  },
  {
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/cores/c906ed39-a1e3-4822-a981-7b9bd57b4b46",
    "CreationTimestamp": "2019-06-18T16:21:21.351Z",
    "Id": "c906ed39-a1e3-4822-a981-7b9bd57b4b46",
    "LastUpdatedTimestamp": "2019-06-18T16:21:21.351Z",
    "LatestVersion": "42aeec3-fd9d-4312-a8fd-ffa9404a20e0",
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/cores/c906ed39-a1e3-4822-a981-7b9bd57b4b46/versions/42aeec3-fd9d-4312-a8fd-ffa9404a20e0"
  },
  {
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/cores/eaf280cb-138c-4d15-af36-6f681a1348f7",
    "CreationTimestamp": "2019-06-18T16:14:17.709Z",
    "Id": "eaf280cb-138c-4d15-af36-6f681a1348f7",
    "LastUpdatedTimestamp": "2019-06-18T16:14:17.709Z",
    "LatestVersion": "467c36e4-c5da-440c-a97b-084e62593b4c",
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/cores/eaf280cb-138c-4d15-af36-6f681a1348f7/versions/467c36e4-c5da-440c-a97b-084e62593b4c"
  }
]
}

```

- For API details, see [ListCoreDefinitions](#) in *AWS CLI Command Reference*.

list-deployments

The following code example shows how to use `list-deployments`.

AWS CLI

To list the deployments for a Greengrass group

The following `list-deployments` example lists the deployments for the specified Greengrass group. You can use the `list-groups` command to look up your group ID.

```
aws greengrass list-deployments \
  --group-id "1013db12-8b58-45ff-acc7-704248f66731"
```

Output:

```
{
  "Deployments": [
    {
      "CreatedAt": "2019-06-18T17:04:32.702Z",
      "DeploymentId": "1065b8a0-812b-4f21-9d5d-e89b232a530f",
      "DeploymentType": "NewDeployment",
      "GroupArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/1013db12-8b58-45ff-acc7-704248f66731/versions/115136b3-cfd7-4462-
b77f-8741a4b00e5e"
    }
  ]
}
```

- For API details, see [ListDeployments](#) in *AWS CLI Command Reference*.

list-device-definition-versions

The following code example shows how to use `list-device-definition-versions`.

AWS CLI

To list the versions of a device definition

The following `list-device-definition-versions` example displays the device definition versions associated with the specified device definition.

```
aws greengrass list-device-definition-versions \  
  --device-definition-id "f9ba083d-5ad4-4534-9f86-026a45df1ccd"
```

Output:

```
{  
  "Versions": [  
    {  
      "Version": "83c13984-6fed-447e-84d5-5b8aa45d5f71",  
      "CreationTimestamp": "2019-09-11T00:15:09.838Z",  
      "Id": "f9ba083d-5ad4-4534-9f86-026a45df1ccd",  
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/  
greengrass/definition/devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd/  
versions/83c13984-6fed-447e-84d5-5b8aa45d5f71"  
    },  
    {  
      "Version": "3b5cc510-58c1-44b5-9d98-4ad858ffa795",  
      "CreationTimestamp": "2019-09-11T00:11:06.197Z",  
      "Id": "f9ba083d-5ad4-4534-9f86-026a45df1ccd",  
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/  
greengrass/definition/devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd/  
versions/3b5cc510-58c1-44b5-9d98-4ad858ffa795"  
    }  
  ]  
}
```

- For API details, see [ListDeviceDefinitionVersions](#) in *AWS CLI Command Reference*.

list-device-definitions

The following code example shows how to use `list-device-definitions`.

AWS CLI

To list your device definitions

The following `list-device-definitions` example displays details about the device definitions in your AWS account in the specified AWS Region.

```
aws greengrass list-device-definitions \  
  --region us-west-2
```

Output:

```
{
  "Definitions": [
    {
      "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/devices/50f3274c-3f0a-4f57-b114-6f46085281ab/versions/c777b0f5-1059-449b-beaa-f003ebc56c34",
      "LastUpdatedTimestamp": "2019-06-14T15:42:09.059Z",
      "LatestVersion": "c777b0f5-1059-449b-beaa-f003ebc56c34",
      "CreationTimestamp": "2019-06-14T15:42:09.059Z",
      "Id": "50f3274c-3f0a-4f57-b114-6f46085281ab",
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/devices/50f3274c-3f0a-4f57-b114-6f46085281ab"
    },
    {
      "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/devices/e01951c9-6134-479a-969a-1a15cac11c40/versions/514d57aa-4ee6-401c-9fac-938a9f7a51e5",
      "Name": "TestDeviceDefinition",
      "LastUpdatedTimestamp": "2019-04-16T23:17:43.245Z",
      "LatestVersion": "514d57aa-4ee6-401c-9fac-938a9f7a51e5",
      "CreationTimestamp": "2019-04-16T23:17:43.245Z",
      "Id": "e01951c9-6134-479a-969a-1a15cac11c40",
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/devices/e01951c9-6134-479a-969a-1a15cac11c40"
    },
    {
      "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd/versions/83c13984-6fed-447e-84d5-5b8aa45d5f71",
      "Name": "TemperatureSensors",
      "LastUpdatedTimestamp": "2019-09-10T00:19:03.698Z",
      "LatestVersion": "83c13984-6fed-447e-84d5-5b8aa45d5f71",
      "CreationTimestamp": "2019-09-11T00:11:06.197Z",
      "Id": "f9ba083d-5ad4-4534-9f86-026a45df1ccd",
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/devices/f9ba083d-5ad4-4534-9f86-026a45df1ccd"
    }
  ]
}
```

- For API details, see [ListDeviceDefinitions](#) in *AWS CLI Command Reference*.

list-function-definition-versions

The following code example shows how to use `list-function-definition-versions`.

AWS CLI

To list the versions of a Lambda function

The following `list-function-definition-versions` example lists all of the versions of the specified Lambda function. You can use the `list-function-definitions` command to get the ID.

```
aws greengrass list-function-definition-versions \  
  --function-definition-id "063f5d1a-1dd1-40b4-9b51-56f8993d0f85"
```

Output:

```
{  
  "Versions": [  
    {  
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/  
greengrass/definition/functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85/  
versions/9748fda7-1589-4fcc-ac94-f5559e88678b",  
      "CreationTimestamp": "2019-06-18T17:04:30.776Z",  
      "Id": "063f5d1a-1dd1-40b4-9b51-56f8993d0f85",  
      "Version": "9748fda7-1589-4fcc-ac94-f5559e88678b"  
    },  
    {  
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/  
greengrass/definition/functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85/  
versions/9b08df77-26f2-4c29-93d2-769715edcfec",  
      "CreationTimestamp": "2019-06-18T17:02:44.087Z",  
      "Id": "063f5d1a-1dd1-40b4-9b51-56f8993d0f85",  
      "Version": "9b08df77-26f2-4c29-93d2-769715edcfec"  
    },  
    {  
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/  
greengrass/definition/functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85/  
versions/4236239f-94f7-4b90-a2f8-2a24c829d21e",  
      "CreationTimestamp": "2019-06-18T17:01:42.284Z",  
      "Id": "063f5d1a-1dd1-40b4-9b51-56f8993d0f85",  
      "Version": "4236239f-94f7-4b90-a2f8-2a24c829d21e"  
    }  
  ]  
}
```

```

    },
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85/versions/343408bb-549a-4fbe-b043-853643179a39",
      "CreationTimestamp": "2019-06-18T16:21:21.431Z",
      "Id": "063f5d1a-1dd1-40b4-9b51-56f8993d0f85",
      "Version": "343408bb-549a-4fbe-b043-853643179a39"
    }
  ]
}

```

- For API details, see [ListFunctionDefinitionVersions](#) in *AWS CLI Command Reference*.

list-function-definitions

The following code example shows how to use `list-function-definitions`.

AWS CLI

To list Lambda functions

The following `list-function-definitions` example lists all of the Lambda functions defined for your AWS account.

```
aws greengrass list-function-definitions
```

Output:

```

{
  "Definitions": [
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/functions/017970a5-8952-46dd-b1c1-020b3ae8e960",
      "CreationTimestamp": "2018-10-17T04:30:32.884Z",
      "Id": "017970a5-8952-46dd-b1c1-020b3ae8e960",
      "LastUpdatedTimestamp": "2018-10-17T04:30:32.884Z",
      "LatestVersion": "4380b302-790d-4ed8-92bf-02e88afecb15",
      "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/definition/functions/017970a5-8952-46dd-b1c1-020b3ae8e960/versions/4380b302-790d-4ed8-92bf-02e88afecb15"
    }
  ]
}

```



```

    },
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85",
      "CreationTimestamp": "2019-06-18T16:21:21.431Z",
      "Id": "063f5d1a-1dd1-40b4-9b51-56f8993d0f85",
      "LastUpdatedTimestamp": "2019-06-18T16:21:21.431Z",
      "LatestVersion": "9748fda7-1589-4fcc-ac94-f5559e88678b",
      "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/functions/063f5d1a-1dd1-40b4-9b51-56f8993d0f85/
versions/9748fda7-1589-4fcc-ac94-f5559e88678b"
    },
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/functions/6598e653-a262-440c-9967-e2697f64da7b",
      "CreationTimestamp": "2019-06-18T16:24:16.123Z",
      "Id": "6598e653-a262-440c-9967-e2697f64da7b",
      "LastUpdatedTimestamp": "2019-06-18T16:24:16.123Z",
      "LatestVersion": "38bc6ccd-98a2-4ce7-997e-16c84748fae4",
      "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/functions/6598e653-a262-440c-9967-e2697f64da7b/
versions/38bc6ccd-98a2-4ce7-997e-16c84748fae4"
    },
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/functions/c668df84-fad2-491b-95f4-655d2cad7885",
      "CreationTimestamp": "2019-06-18T16:14:17.784Z",
      "Id": "c668df84-fad2-491b-95f4-655d2cad7885",
      "LastUpdatedTimestamp": "2019-06-18T16:14:17.784Z",
      "LatestVersion": "37dd68c4-a64f-40ba-aa13-71fecc3ebded",
      "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/functions/c668df84-fad2-491b-95f4-655d2cad7885/
versions/37dd68c4-a64f-40ba-aa13-71fecc3ebded"
    }
  ]
}

```

- For API details, see [ListFunctionDefinitions](#) in *AWS CLI Command Reference*.

list-group-certificate-authorities

The following code example shows how to use `list-group-certificate-authorities`.

AWS CLI

To list the current CAs for a group

The following `list-group-certificate-authorities` example lists the current certificate authorities (CAs) for the specified Greengrass group.

```
aws greengrass list-group-certificate-authorities \  
  --group-id "1013db12-8b58-45ff-acc7-704248f66731"
```

Output:

```
{  
  "GroupCertificateAuthorities": [  
    {  
      "GroupCertificateAuthorityArn": "arn:aws:greengrass:us-  
west-2:123456789012:/greengrass/groups/1013db12-8b58-45ff-acc7-704248f66731/  
certificateauthorities/  
f0430e1736ea8ed30cc5d5de9af67a7e3586bad9ae4d89c2a44163f65fdd8cf6",  
      "GroupCertificateAuthorityId":  
      "f0430e1736ea8ed30cc5d5de9af67a7e3586bad9ae4d89c2a44163f65fdd8cf6"  
    }  
  ]  
}
```

- For API details, see [ListGroupCertificateAuthorities](#) in *AWS CLI Command Reference*.

list-group-versions

The following code example shows how to use `list-group-versions`.

AWS CLI

To list the versions of a Greengrass group

The following `list-group-versions` example lists the versions of the specified Greengrass group.

```
aws greengrass list-group-versions \  
  --group-id "1013db12-8b58-45ff-acc7-704248f66731"
```

Output:

```
{
  "Versions": [
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/1013db12-8b58-45ff-acc7-704248f66731/versions/115136b3-cfd7-4462-
b77f-8741a4b00e5e",
      "CreationTimestamp": "2019-06-18T17:04:30.915Z",
      "Id": "1013db12-8b58-45ff-acc7-704248f66731",
      "Version": "115136b3-cfd7-4462-b77f-8741a4b00e5e"
    },
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/1013db12-8b58-45ff-acc7-704248f66731/versions/4340669d-
d14d-44e3-920c-46c928750750",
      "CreationTimestamp": "2019-06-18T17:03:52.663Z",
      "Id": "1013db12-8b58-45ff-acc7-704248f66731",
      "Version": "4340669d-d14d-44e3-920c-46c928750750"
    },
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/1013db12-8b58-45ff-acc7-704248f66731/
versions/1b06e099-2d5b-4f10-91b9-78c4e060f5da",
      "CreationTimestamp": "2019-06-18T17:02:44.189Z",
      "Id": "1013db12-8b58-45ff-acc7-704248f66731",
      "Version": "1b06e099-2d5b-4f10-91b9-78c4e060f5da"
    },
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/1013db12-8b58-45ff-acc7-704248f66731/versions/2d3f27f1-3b43-4554-
ab7a-73ec30477efe",
      "CreationTimestamp": "2019-06-18T17:01:42.401Z",
      "Id": "1013db12-8b58-45ff-acc7-704248f66731",
      "Version": "2d3f27f1-3b43-4554-ab7a-73ec30477efe"
    },
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/1013db12-8b58-45ff-acc7-704248f66731/versions/d20f7ae9-3444-4c1c-b025-
e2ede23cdd31",
      "CreationTimestamp": "2019-06-18T16:21:21.457Z",
      "Id": "1013db12-8b58-45ff-acc7-704248f66731",
      "Version": "d20f7ae9-3444-4c1c-b025-e2ede23cdd31"
    }
  ]
}
```

```
]
}
```

- For API details, see [ListGroupVersions](#) in *AWS CLI Command Reference*.

list-groups

The following code example shows how to use `list-groups`.

AWS CLI

To list the Greengrass groups

The following `list-groups` example lists all Greengrass groups that are defined in your AWS account.

```
aws greengrass list-groups
```

Output:

```
{
  "Groups": [
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/groups/1013db12-8b58-45ff-acc7-704248f66731",
      "CreationTimestamp": "2019-06-18T16:21:21.457Z",
      "Id": "1013db12-8b58-45ff-acc7-704248f66731",
      "LastUpdatedTimestamp": "2019-06-18T16:21:21.457Z",
      "LatestVersion": "115136b3-cfd7-4462-b77f-8741a4b00e5e",
      "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/groups/1013db12-8b58-45ff-acc7-704248f66731/versions/115136b3-cfd7-4462-b77f-8741a4b00e5e",
      "Name": "GGGroup4Pi3"
    },
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/groups/1402daf9-71cf-4cfe-8be0-d5e80526d0d8",
      "CreationTimestamp": "2018-10-31T21:52:46.603Z",
      "Id": "1402daf9-71cf-4cfe-8be0-d5e80526d0d8",
      "LastUpdatedTimestamp": "2018-10-31T21:52:46.603Z",
      "LatestVersion": "749af901-60ab-456f-a096-91b12d983c29",

```

```

    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/1402daf9-71cf-4cfe-8be0-d5e80526d0d8/versions/749af901-60ab-456f-
a096-91b12d983c29",
    "Name": "MyTestGroup"
  },
  {
    "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
groups/504b5c8d-bbed-4635-aff1-48ec5b586db5",
    "CreationTimestamp": "2018-12-31T21:39:36.771Z",
    "Id": "504b5c8d-bbed-4635-aff1-48ec5b586db5",
    "LastUpdatedTimestamp": "2018-12-31T21:39:36.771Z",
    "LatestVersion": "46911e8e-f9bc-4898-8b63-59c7653636ec",
    "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/groups/504b5c8d-bbed-4635-aff1-48ec5b586db5/versions/46911e8e-
f9bc-4898-8b63-59c7653636ec",
    "Name": "smp-ggrass-group"
  }
]
}

```

- For API details, see [ListGroups](#) in *AWS CLI Command Reference*.

list-logger-definition-versions

The following code example shows how to use `list-logger-definition-versions`.

AWS CLI

To get a list of versions of a logger definition

The following `list-logger-definition-versions` example gets a list of all versions of the specified logger definition.

```
aws greengrass list-logger-definition-versions \
  --logger-definition-id "49eeeb66-f1d3-4e34-86e3-3617262abf23"
```

Output:

```
{
  "Versions": [
    {
```

```

        "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/loggers/49eeeb66-f1d3-4e34-86e3-3617262abf23/versions/5e3f6f64-
a565-491e-8de0-3c0d8e0f2073",
        "CreationTimestamp": "2019-05-08T16:10:13.866Z",
        "Id": "49eeeb66-f1d3-4e34-86e3-3617262abf23",
        "Version": "5e3f6f64-a565-491e-8de0-3c0d8e0f2073"
    },
    {
        "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/loggers/49eeeb66-f1d3-4e34-86e3-3617262abf23/versions/3ec6d3af-eb85-48f9-
a16d-1c795fe696d7",
        "CreationTimestamp": "2019-05-08T16:10:13.809Z",
        "Id": "49eeeb66-f1d3-4e34-86e3-3617262abf23",
        "Version": "3ec6d3af-eb85-48f9-a16d-1c795fe696d7"
    }
]
}

```

- For API details, see [ListLoggerDefinitionVersions](#) in *AWS CLI Command Reference*.

list-logger-definitions

The following code example shows how to use `list-logger-definitions`.

AWS CLI

To get a list of logger definitions

The following `list-logger-definitions` example lists all of the logger definitions for your AWS account.

```
aws greengrass list-logger-definitions
```

Output:

```

{
  "Definitions": [
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/loggers/49eeeb66-f1d3-4e34-86e3-3617262abf23",
      "CreationTimestamp": "2019-05-08T16:10:13.809Z",

```

```

        "Id": "49eeeb66-f1d3-4e34-86e3-3617262abf23",
        "LastUpdatedTimestamp": "2019-05-08T16:10:13.809Z",
        "LatestVersion": "5e3f6f64-a565-491e-8de0-3c0d8e0f2073",
        "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/loggers/49eeeb66-f1d3-4e34-86e3-3617262abf23/
versions/5e3f6f64-a565-491e-8de0-3c0d8e0f2073"
    }
]
}

```

- For API details, see [ListLoggerDefinitions](#) in *AWS CLI Command Reference*.

list-resource-definition-versions

The following code example shows how to use `list-resource-definition-versions`.

AWS CLI

To list the versions of a resource definition

The following `list-resource-definition-versions` example lists the versions for the specified Greengrass resource.

```

aws greengrass list-resource-definition-versions \
  --resource-definition-id "ad8c101d-8109-4b0e-b97d-9cc5802ab658"

```

Output:

```

{
  "Versions": [
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/resources/ad8c101d-8109-4b0e-b97d-9cc5802ab658/
versions/26e8829a-491a-464d-9c87-664bf6f6f2be",
      "CreationTimestamp": "2019-06-19T16:40:59.392Z",
      "Id": "ad8c101d-8109-4b0e-b97d-9cc5802ab658",
      "Version": "26e8829a-491a-464d-9c87-664bf6f6f2be"
    },
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/resources/ad8c101d-8109-4b0e-b97d-9cc5802ab658/
versions/432d92f6-12de-4ec9-a704-619a942a62aa",

```

```

    "CreationTimestamp": "2019-06-19T16:40:59.261Z",
    "Id": "ad8c101d-8109-4b0e-b97d-9cc5802ab658",
    "Version": "432d92f6-12de-4ec9-a704-619a942a62aa"
  }
]
}

```

- For API details, see [ListResourceDefinitionVersions](#) in *AWS CLI Command Reference*.

list-resource-definitions

The following code example shows how to use `list-resource-definitions`.

AWS CLI

To list the resources that are defined

The following `list-resource-definitions` example lists the resources that are defined for AWS IoT Greengrass to use.

```
aws greengrass list-resource-definitions
```

Output:

```

{
  "Definitions": [
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/resources/ad8c101d-8109-4b0e-b97d-9cc5802ab658",
      "CreationTimestamp": "2019-06-19T16:40:59.261Z",
      "Id": "ad8c101d-8109-4b0e-b97d-9cc5802ab658",
      "LastUpdatedTimestamp": "2019-06-19T16:40:59.261Z",
      "LatestVersion": "26e8829a-491a-464d-9c87-664bf6f6f2be",
      "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/resources/ad8c101d-8109-4b0e-b97d-9cc5802ab658/
versions/26e8829a-491a-464d-9c87-664bf6f6f2be"
    },
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/resources/c8bb9ebc-c3fd-40a4-9c6a-568d75569d38",
      "CreationTimestamp": "2019-06-19T21:51:28.212Z",

```



```

        "Id": "c8bb9ebc-c3fd-40a4-9c6a-568d75569d38",
        "LastUpdatedTimestamp": "2019-06-19T21:51:28.212Z",
        "LatestVersion": "a5f94d0b-f6bc-40f4-bb78-7a1c5fe13ba1",
        "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/resources/c8bb9ebc-c3fd-40a4-9c6a-568d75569d38/versions/
a5f94d0b-f6bc-40f4-bb78-7a1c5fe13ba1",
        "Name": "MyGreengrassResources"
    }
]
}

```

- For API details, see [ListResourceDefinitions](#) in *AWS CLI Command Reference*.

list-subscription-definition-versions

The following code example shows how to use `list-subscription-definition-versions`.

AWS CLI

To list the versions of a subscription definition

The following `list-subscription-definition-versions` example lists all versions of the specified subscription. You can use the `list-subscription-definitions` command to look up the subscription ID.

```
aws greengrass list-subscription-definition-versions \
  --subscription-definition-id "70e49321-83d5-45d2-bc09-81f4917ae152"
```

Output:

```

{
  "Versions": [
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/subscriptions/70e49321-83d5-45d2-bc09-81f4917ae152/
versions/88ae8699-12ac-4663-ba3f-4d7f0519140b",
      "CreationTimestamp": "2019-06-18T17:03:52.499Z",
      "Id": "70e49321-83d5-45d2-bc09-81f4917ae152",
      "Version": "88ae8699-12ac-4663-ba3f-4d7f0519140b"
    },
    {

```

```

        "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/subscriptions/70e49321-83d5-45d2-bc09-81f4917ae152/versions/7e320ba3-
c369-4069-a2f0-90acb7f219d6",
        "CreationTimestamp": "2019-06-18T17:03:52.392Z",
        "Id": "70e49321-83d5-45d2-bc09-81f4917ae152",
        "Version": "7e320ba3-c369-4069-a2f0-90acb7f219d6"
    }
]
}

```

- For API details, see [ListSubscriptionDefinitionVersions](#) in *AWS CLI Command Reference*.

list-subscription-definitions

The following code example shows how to use `list-subscription-definitions`.

AWS CLI

To get a list subscription definitions

The following `list-subscription-definitions` example lists all of the AWS IoT Greengrass subscriptions that are defined in your AWS account.

```
aws greengrass list-subscription-definitions
```

Output:

```

{
  "Definitions": [
    {
      "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/subscriptions/70e49321-83d5-45d2-bc09-81f4917ae152",
      "CreationTimestamp": "2019-06-18T17:03:52.392Z",
      "Id": "70e49321-83d5-45d2-bc09-81f4917ae152",
      "LastUpdatedTimestamp": "2019-06-18T17:03:52.392Z",
      "LatestVersion": "88ae8699-12ac-4663-ba3f-4d7f0519140b",
      "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/subscriptions/70e49321-83d5-45d2-bc09-81f4917ae152/
versions/88ae8699-12ac-4663-ba3f-4d7f0519140b"
    },
    {

```

```

        "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/subscriptions/cd6f1c37-d9a4-4e90-be94-01a7404f5967",
        "CreationTimestamp": "2018-10-18T15:45:34.024Z",
        "Id": "cd6f1c37-d9a4-4e90-be94-01a7404f5967",
        "LastUpdatedTimestamp": "2018-10-18T15:45:34.024Z",
        "LatestVersion": "d1cf8fac-284f-4f6a-98fe-a2d36d089373",
        "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/subscriptions/cd6f1c37-d9a4-4e90-be94-01a7404f5967/versions/
d1cf8fac-284f-4f6a-98fe-a2d36d089373"
    },
    {
        "Arn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/subscriptions/fa81bc84-3f59-4377-a84b-5d0134da359b",
        "CreationTimestamp": "2018-10-22T17:09:31.429Z",
        "Id": "fa81bc84-3f59-4377-a84b-5d0134da359b",
        "LastUpdatedTimestamp": "2018-10-22T17:09:31.429Z",
        "LatestVersion": "086d1b08-b25a-477c-a16f-6f9b3a9c295a",
        "LatestVersionArn": "arn:aws:greengrass:us-west-2:123456789012:/
greengrass/definition/subscriptions/fa81bc84-3f59-4377-a84b-5d0134da359b/
versions/086d1b08-b25a-477c-a16f-6f9b3a9c295a"
    }
]
}

```

- For API details, see [ListSubscriptionDefinitions](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags attached to a resource

The following `list-tags-for-resource` example lists the tags and their values that are attached to the specified resource.

```

aws greengrass list-tags-for-resource \
  --resource-arn "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
definition/resources/ad8c101d-8109-4b0e-b97d-9cc5802ab658"

```

Output:

```
{
  "tags": {
    "ResourceSubType": "USB",
    "ResourceType": "Device"
  }
}
```

For more information, see [Tagging Your Greengrass Resources](#) in the **AWS IoT Greengrass Developer Guide**.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

reset-deployments

The following code example shows how to use `reset-deployments`.

AWS CLI

To clean up deployment information for a Greengrass group

The following `reset-deployments` example cleans up deployment information for the specified Greengrass group. When you add the `--force` option, the deployment information is reset without waiting for the core device to respond.

```
aws greengrass reset-deployments \
  --group-id "1402daf9-71cf-4cfe-8be0-d5e80526d0d8" \
  --force
```

Output:

```
{
  "DeploymentArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/groups/1402daf9-71cf-4cfe-8be0-d5e80526d0d8/deployments/7dd4e356-9882-46a3-9e28-6d21900c011a",
  "DeploymentId": "7dd4e356-9882-46a3-9e28-6d21900c011a"
}
```

For more information, see [Reset Deployments](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [ResetDeployments](#) in *AWS CLI Command Reference*.

start-bulk-deployment

The following code example shows how to use `start-bulk-deployment`.

AWS CLI

To start a bulk deployment operation

The following `start-bulk-deployment` example starts a bulk deployment operation, using a file stored in an S3 bucket to specify the groups to be deployed.

```
aws greengrass start-bulk-deployment \
  --cli-input-json "{\"InputFileUri\":\"https://gg-group-deployment1.s3-us-
west-2.amazonaws.com/MyBulkDeploymentInputFile.txt\", \"ExecutionRoleArn\":
  \"arn:aws:iam::123456789012:role/ggCreateDeploymentRole\", \"AmznClientToken\":
  \"yourAmazonClientToken\"}"
```

Output:

```
{
  "BulkDeploymentArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/
bulk/deployments/870fb41b-6288-4e0c-bc76-a7ba4b4d3267",
  "BulkDeploymentId": "870fb41b-6288-4e0c-bc76-a7ba4b4d3267"
}
```

For more information, see [Create Bulk Deployments for Groups](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [StartBulkDeployment](#) in *AWS CLI Command Reference*.

stop-bulk-deployment

The following code example shows how to use `stop-bulk-deployment`.

AWS CLI

To stop a bulk deployment

The following `stop-bulk-deployment` example stops the specified bulk deployment. If you try to stop a bulk deployment that is complete, you receive an error:
`InvalidInputException: Cannot change state of finished execution.`

```
aws greengrass stop-bulk-deployment \  
  --bulk-deployment-id "870fb41b-6288-4e0c-bc76-a7ba4b4d3267"
```

This command produces no output.

For more information, see [Create Bulk Deployments for Groups](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [StopBulkDeployment](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To apply tags to a resource

The following `tag-resource` example applies two tags, `ResourceType` and `ResourceSubType`, to the specified Greengrass resource. This operation can both add new tags and values or update the value for existing tags. Use the `untag-resource` command to remove a tag.

```
aws greengrass tag-resource \  
  --resource-arn "arn:aws:greengrass:us-west-2:123456789012:/greengrass/  
definition/resources/ad8c101d-8109-4b0e-b97d-9cc5802ab658" \  
  --tags "ResourceType=Device,ResourceSubType=USB"
```

This command produces no output.

For more information, see [Tagging Your Greengrass Resources](#) in the **AWS IoT Greengrass Developer Guide**.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag and its value from a resource

The following `untag-resource` example removes the tag whose key is `Category` from the specified Greengrass group. If the key `Category` does not exist for the specified resource, no error is returned.

```
aws greengrass untag-resource \  
  --resource-arn "arn:aws:greengrass:us-west-2:123456789012:/greengrass/  
groups/1013db12-8b58-45ff-acc7-704248f66731" \  
  --tag-keys "Category"
```

This command produces no output.

For more information, see [Tagging Your Greengrass Resources](#) in the **AWS IoT Greengrass Developer Guide**.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-connectivity-info

The following code example shows how to use `update-connectivity-info`.

AWS CLI

To update the connectivity information for a Greengrass core

The following `update-connectivity-info` example changes the endpoints that devices can use to connect to the specified Greengrass core. Connectivity information is a list of IP addresses or domain names, with corresponding port numbers and optional customer-defined metadata. You might need to update connectivity information when the local network changes.

```
aws greengrass update-connectivity-info \  
  --thing-name "MyGroup_Core" \  
  --connectivity-info "[{\"Metadata\":\"\",\"PortNumber\":8883,\"HostAddress\":  
\"127.0.0.1\",\"Id\":\"localhost_127.0.0.1_0\"},{\"Metadata\":\"\",\"PortNumber  
\":8883,\"HostAddress\":\"192.168.1.3\",\"Id\":\"localIP_192.168.1.3\"}]"
```

Output:

```
{  
  "Version": "312de337-59af-4cf9-a278-2a23bd39c300"  
}
```

- For API details, see [UpdateConnectivityInfo](#) in *AWS CLI Command Reference*.

update-connector-definition

The following code example shows how to use `update-connector-definition`.

AWS CLI

To update the name for a connector definition

The following `update-connector-definition` example updates the name for the specified connector definition. If you want to update the details for the connector, use the `create-connector-definition-version` command to create a new version.

```
aws greengrass update-connector-definition \  
  --connector-definition-id "55d0052b-0d7d-44d6-b56f-21867215e118" \  
  --name "GreengrassConnectors2019"
```

For more information, see [Integrate with Services and Protocols Using Connectors](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [UpdateConnectorDefinition](#) in *AWS CLI Command Reference*.

update-core-definition

The following code example shows how to use `update-core-definition`.

AWS CLI

To update a core definition

The following `update-core-definition` example changes the name of the specified core definition. You can update only the name property of a core definition.

```
aws greengrass update-core-definition \  
  --core-definition-id "582efe12-b05a-409e-9a24-a2ba1bcc4a12" \  
  --name "MyCoreDevices"
```

This command produces no output.

For more information, see [Configure the AWS IoT Greengrass Core](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [UpdateCoreDefinition](#) in *AWS CLI Command Reference*.

update-device-definition

The following code example shows how to use `update-device-definition`.

AWS CLI

To update a device definition

The following `update-device-definition` example changes the name of the specified device definition. You can only update the name property of a device definition.

```
aws greengrass update-device-definition \  
  --device-definition-id "f9ba083d-5ad4-4534-9f86-026a45df1ccd" \  
  --name "TemperatureSensors"
```

This command produces no output.

- For API details, see [UpdateDeviceDefinition](#) in *AWS CLI Command Reference*.

update-function-definition

The following code example shows how to use `update-function-definition`.

AWS CLI

To update the name for a function definition

The following `update-function-definition` example updates the name for the specified function definition. If you want to update the details for the function, use the `create-function-definition-version` command to create a new version.

```
aws greengrass update-function-definition \  
  --function-definition-id "e47952bd-dea9-4e2c-a7e1-37bbe8807f46" \  
  --name ObsoleteFunction
```

This command produces no output.

For more information, see [Run Local Lambda Functions](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [UpdateFunctionDefinition](#) in *AWS CLI Command Reference*.

update-group-certificate-configuration

The following code example shows how to use `update-group-certificate-configuration`.

AWS CLI

To update the expiry of a group's certificates

The following `update-group-certificate-configuration` example sets a 10-day expiry for the certificates generated for the specified group.

```
aws greengrass update-group-certificate-configuration \  
  --group-id "8eaadd72-ce4b-4f15-892a-0cc4f3a343f1" \  
  --certificate-expiry-in-milliseconds 864000000
```

Output:

```
{  
  "CertificateExpiryInMilliseconds": 864000000,  
  "CertificateAuthorityExpiryInMilliseconds": 2524607999000,  
  "GroupId": "8eaadd72-ce4b-4f15-892a-0cc4f3a343f1"  
}
```

For more information, see [AWS IoT Greengrass Security](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [UpdateGroupCertificateConfiguration](#) in *AWS CLI Command Reference*.

update-group

The following code example shows how to use `update-group`.

AWS CLI

To update the group name

The following `update-group` example updates the name of the specified Greengrass group. If you want to update the details for the group, use the `create-group-version` command to create a new version.

```
aws greengrass update-group \  
  --group-id "1402daf9-71cf-4cfe-8be0-d5e80526d0d8" \  
  --name TestGroup4of6
```

For more information, see [Configure AWS IoT Greengrass on AWS IoT](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [UpdateGroup](#) in *AWS CLI Command Reference*.

update-logger-definition

The following code example shows how to use `update-logger-definition`.

AWS CLI

To update a logger definition

The following `update-logger-definition` example changes the name of the specified logger definition. You can only update the name property of a logger definition.

```
aws greengrass update-logger-definition \  
  --logger-definition-id "a454b62a-5d56-4ca9-bdc4-8254e1662cb0" \  
  --name "LoggingConfigsForSensors"
```

This command produces no output.

For more information, see [Monitoring with AWS IoT Greengrass Logs](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [UpdateLoggerDefinition](#) in *AWS CLI Command Reference*.

update-resource-definition

The following code example shows how to use `update-resource-definition`.

AWS CLI

To update the name for a resource definition

The following `update-resource-definition` example updates the name for the specified resource definition. If you want to change the details for the resource, use the `create-resource-definition-version` command to create a new version.

```
aws greengrass update-resource-definition \  
  --resource-definition-id "c8bb9ebc-c3fd-40a4-9c6a-568d75569d38" \  
  --name GreengrassConnectorResources
```

This command produces no output.

For more information, see [Access Local Resources with Lambda Functions and Connectors](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [UpdateResourceDefinition](#) in *AWS CLI Command Reference*.

update-subscription-definition

The following code example shows how to use `update-subscription-definition`.

AWS CLI

To update the name for a subscription definition

The following `update-subscription-definition` example updates the name for the specified subscription definition. If you want to change details for the subscription, use the `create-subscription-definition-version` command to create a new version.

```
aws greengrass update-subscription-definition \  
  --subscription-definition-id "fa81bc84-3f59-4377-a84b-5d0134da359b" \  
  --name "ObsoleteSubscription"
```

This command produces no output.

For more information, see title in the *guide*.

- For API details, see [UpdateSubscriptionDefinition](#) in *AWS CLI Command Reference*.

update-thing-runtime-configuration

The following code example shows how to use `update-thing-runtime-configuration`.

AWS CLI

To turn on telemetry in the runtime configuration of a Greengrass core

The following `update-thing-runtime-configuration` example updates the runtime configuration of a Greengrass core to turn on telemetry.

```
aws greengrass update-thing-runtime-configuration \  
  --thing-name SampleGreengrassCore \  
  --telemetry-configuration {"Telemetry\":"On\"}
```

This command produces no output.

For more information, see [Configuring telemetry settings](#) in the *AWS IoT Greengrass Developer Guide*.

- For API details, see [UpdateThingRuntimeConfiguration](#) in *AWS CLI Command Reference*.

AWS IoT Greengrass V2 examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS IoT Greengrass V2.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-service-role-to-account

The following code example shows how to use `associate-service-role-to-account`.

AWS CLI

To associate the Greengrass service role to your AWS account

The following `associate-service-role-to-account` example associates a service role with AWS IoT Greengrass for your AWS account.

```
aws greengrassv2 associate-service-role-to-account \  
  --role-arn arn:aws:iam::123456789012:role/service-role/Greengrass_ServiceRole
```

Output:

```
{  
  "associatedAt": "2022-01-19T19:21:53Z"  
}
```

For more information, see [Greengrass service role](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [AssociateServiceRoleToAccount](#) in *AWS CLI Command Reference*.

batch-associate-client-device-with-core-device

The following code example shows how to use `batch-associate-client-device-with-core-device`.

AWS CLI

To associate client devices with a core device

The following `batch-associate-client-device-with-core-device` example associates two client devices with a core device.

```
aws greengrassv2 batch-associate-client-device-with-core-device \  
  --core-device-thing-name MyGreengrassCore \  
  --entries thingName=MyClientDevice1 thingName=MyClientDevice2
```

Output:

```
{  
  "errorEntries": []  
}
```

For more information, see [Interact with local IoT devices](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [BatchAssociateClientDeviceWithCoreDevice](#) in *AWS CLI Command Reference*.

batch-disassociate-client-device-from-core-device

The following code example shows how to use `batch-disassociate-client-device-from-core-device`.

AWS CLI

To disassociate client devices from a core device

The following `batch-disassociate-client-device-from-core-device` example disassociates two client devices from a core device.

```
aws greengrassv2 batch-disassociate-client-device-from-core-device \  
  --core-device-thing-name MyGreengrassCore \  
  --entries thingName=MyClientDevice1 thingName=MyClientDevice2
```

Output:

```
{  
  "errorEntries": []  
}
```

For more information, see [Interact with local IoT devices](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [BatchDisassociateClientDeviceFromCoreDevice](#) in *AWS CLI Command Reference*.

cancel-deployment

The following code example shows how to use `cancel-deployment`.

AWS CLI

To cancel a deployment

The following `cancel-deployment` example stops a continuous deployment to a thing group.

```
aws greengrassv2 cancel-deployment \  
  --deployment-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{  
  "message": "SUCCESS"  
}
```

For more information, see [Cancel deployments](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [CancelDeployment](#) in *AWS CLI Command Reference*.

create-component-version

The following code example shows how to use `create-component-version`.

AWS CLI

Example 1: To create a component version from a recipe

The following `create-component-version` example creates a version of a Hello World component from a recipe file.

```
aws greengrassv2 create-component-version \  
  --inline-recipe fileb://com.example.HelloWorld-1.0.0.json
```

Contents of `com.example.HelloWorld-1.0.0.json`:

```
{  
  "RecipeFormatVersion": "2020-01-25",  
  "ComponentName": "com.example.HelloWorld",  
  "ComponentVersion": "1.0.0",  
  "ComponentDescription": "My first AWS IoT Greengrass component.",  
  "ComponentPublisher": "Amazon",  
  "ComponentConfiguration": {  
    "DefaultConfiguration": {  
      "Message": "world"  
    }  
  },  
  "Manifests": [  
    {  
      "ComponentName": "com.example.HelloWorld",  
      "ComponentVersion": "1.0.0",  
      "RecipeFormatVersion": "2020-01-25",  
      "RecipePath": "fileb://com.example.HelloWorld-1.0.0.json"  
    }  
  ]  
}
```



```

    {
      "Platform": {
        "os": "linux"
      },
      "Lifecycle": {
        "Run": "echo 'Hello {configuration:/Message}'"
      }
    }
  ]
}

```

Output:

```

{
  "arn": "arn:aws:greengrass:us-
west-2:123456789012:components:com.example.HelloWorld:versions:1.0.0",
  "componentName": "com.example.HelloWorld",
  "componentVersion": "1.0.0",
  "creationTimestamp": "2021-01-07T16:24:33.650000-08:00",
  "status": {
    "componentState": "REQUESTED",
    "message": "NONE",
    "errors": {}
  }
}

```

For more information, see [Create custom components](#) and [Upload components to deploy](#) in the *AWS IoT Greengrass V2 Developer Guide*.

Example 2: To create a component version from an AWS Lambda function

The following `create-component-version` example creates a version of a Hello World component from an AWS Lambda function.

```

aws greengrassv2 create-component-version \
  --cli-input-json file://lambda-function-component.json

```

Contents of `lambda-function-component.json`:

```

{
  "lambdaFunction": {

```

```
    "lambdaArn": "arn:aws:lambda:us-  
west-2:123456789012:function>HelloWorldPythonLambda:1",  
    "componentName": "com.example>HelloWorld",  
    "componentVersion": "1.0.0",  
    "componentLambdaParameters": {  
      "eventSources": [  
        {  
          "topic": "hello/world/+",  
          "type": "IOT_CORE"  
        }  
      ]  
    }  
  }  
}
```

Output:

```
{  
  "arn": "arn:aws:greengrass:us-  
west-2:123456789012:components:com.example>HelloWorld:versions:1.0.0",  
  "componentName": "com.example>HelloWorld",  
  "componentVersion": "1.0.0",  
  "creationTimestamp": "2021-01-07T17:05:27.347000-08:00",  
  "status": {  
    "componentState": "REQUESTED",  
    "message": "NONE",  
    "errors": {}  
  }  
}
```

For more information, see [Run AWS Lambda functions](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [CreateComponentVersion](#) in *AWS CLI Command Reference*.

create-deployment

The following code example shows how to use create-deployment.

AWS CLI

Example 1: To create a deployment

The following create-deployment example deploys the AWS IoT Greengrass Command Line Interface to a core device.

```
aws greengrassv2 create-deployment \  
  --cli-input-json file://cli-deployment.json
```

Contents of `cli-deployment.json`:

```
{  
  "targetArn": "arn:aws:iot:us-west-2:123456789012:thing/MyGreengrassCore",  
  "deploymentName": "Deployment for MyGreengrassCore",  
  "components": {  
    "aws.greengrass.Cli": {  
      "componentVersion": "2.0.3"  
    }  
  },  
  "deploymentPolicies": {  
    "failureHandlingPolicy": "DO_NOTHING",  
    "componentUpdatePolicy": {  
      "timeoutInSeconds": 60,  
      "action": "NOTIFY_COMPONENTS"  
    },  
    "configurationValidationPolicy": {  
      "timeoutInSeconds": 60  
    }  
  },  
  "iotJobConfiguration": {}  
}
```

Output:

```
{  
  "deploymentId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"  
}
```

For more information, see [Create deployments](#) in the *AWS IoT Greengrass V2 Developer Guide*.

Example 2: To create a deployment that updates component configurations

The following create-deployment example deploys the AWS IoT Greengrass nucleus component to a group of core devices. This deployment applies the following configuration updates for the nucleus component:

Reset the target devices' proxy settings to their default no proxy settings. Reset the target devices' MQTT settings to their defaults. Sets the JVM options for the nucleus' JVM. Sets the logging level for the nucleus.

```
aws greengrassv2 create-deployment \  
  --cli-input-json file://nucleus-deployment.json
```

Contents of nucleus-deployment.json:

```
{  
  "targetArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/  
MyGreengrassCoreGroup",  
  "deploymentName": "Deployment for MyGreengrassCoreGroup",  
  "components": {  
    "aws.greengrass.Nucleus": {  
      "componentVersion": "2.0.3",  
      "configurationUpdate": {  
        "reset": [  
          "/networkProxy",  
          "/mqtt"  
        ],  
        "merge": "{\"jvmOptions\":\"-Xmx64m\",\"logging\":{\"level\":\"WARN  
\"}}"  
      }  
    }  
  },  
  "deploymentPolicies": {  
    "failureHandlingPolicy": "ROLLBACK",  
    "componentUpdatePolicy": {  
      "timeoutInSeconds": 60,  
      "action": "NOTIFY_COMPONENTS"  
    },  
    "configurationValidationPolicy": {  
      "timeoutInSeconds": 60  
    }  
  },  
  "iotJobConfiguration": {}  
}
```

Output:

```
{
```

```
"deploymentId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
"iotJobId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
"iotJobArn": "arn:aws:iot:us-west-2:123456789012:job/a1b2c3d4-5678-90ab-cdef-
EXAMPLE22222"
}
```

For more information, see [Create deployments](#) and [Update component configurations](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [CreateDeployment](#) in *AWS CLI Command Reference*.

delete-component

The following code example shows how to use `delete-component`.

AWS CLI

To delete a component version

The following `delete-component` example deletes a Hello World component.

```
aws greengrassv2 delete-component \
  --arn arn:aws:greengrass:us-
west-2:123456789012:components:com.example>HelloWorld:versions:1.0.0
```

This command produces no output.

For more information, see [Manage components](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [DeleteComponent](#) in *AWS CLI Command Reference*.

delete-core-device

The following code example shows how to use `delete-core-device`.

AWS CLI

To delete a core device

The following `delete-core-device` example deletes an AWS IoT Greengrass core device.

```
aws greengrassv2 delete-core-device \
  --core-device-thing-name MyGreengrassCore
```

This command produces no output.

For more information, see [Uninstall the AWS IoT Greengrass Core software](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [DeleteCoreDevice](#) in *AWS CLI Command Reference*.

describe-component

The following code example shows how to use describe-component.

AWS CLI

To describe a component version

The following describe-component example describes a Hello World component.

```
aws greengrassv2 describe-component \  
  --arn arn:aws:greengrass:us-  
west-2:123456789012:components:com.example>HelloWorld:versions:1.0.0
```

Output:

```
{  
  "arn": "arn:aws:greengrass:us-  
west-2:123456789012:components:com.example>HelloWorld:versions:1.0.0",  
  "componentName": "com.example>HelloWorld",  
  "componentVersion": "1.0.0",  
  "creationTimestamp": "2021-01-07T17:12:11.133000-08:00",  
  "publisher": "Amazon",  
  "description": "My first AWS IoT Greengrass component.",  
  "status": {  
    "componentState": "DEPLOYABLE",  
    "message": "NONE",  
    "errors": {}  
  },  
  "platforms": [  
    {  
      "attributes": {  
        "os": "linux"  
      }  
    }  
  ]  
}
```

```
}
```

For more information, see [Manage components](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [DescribeComponent](#) in *AWS CLI Command Reference*.

disassociate-service-role-from-account

The following code example shows how to use `disassociate-service-role-from-account`.

AWS CLI

To disassociate the Greengrass service role from your AWS account

The following `disassociate-service-role-from-account` example disassociates the Greengrass service role from AWS IoT Greengrass for your AWS account.

```
aws greengrassv2 disassociate-service-role-from-account
```

Output:

```
{
  "disassociatedAt": "2022-01-19T19:26:09Z"
}
```

For more information, see [Greengrass service role](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [DisassociateServiceRoleFromAccount](#) in *AWS CLI Command Reference*.

get-component-version-artifact

The following code example shows how to use `get-component-version-artifact`.

AWS CLI

To get a URL to download a component artifact

The following `get-component-version-artifact` example gets a URL to download the local debug console component's JAR file.

```
aws greengrassv2 get-component-version-artifact \
```

```
--arn arn:aws:greengrass:us-west-2:aws:components:aws.greengrass.LocalDebugConsole:versions:2.0.3 \
--artifact-name "Uvt6ZEzQ9TKiAuLbfXBX_APdY0TWks3uc46tHFHTzBM=/aws.greengrass.LocalDebugConsole.jar"
```

Output:

```
{
  "preSignedUrl": "https://evergreencomponentmanageme-
artifactbucket7410c9ef-g18n1iya8kwr.s3.us-west-2.amazonaws.com/public/
aws.greengrass.LocalDebugConsole/2.0.3/s3/ggv2-component-releases-prod-pdx/
EvergreenHttpDebugView/2ffc496ba41b39568968b22c582b4714a937193ee7687a45527238e696672521/
aws.greengrass.LocalDebugConsole/aws.greengrass.LocalDebugConsole.jar?X-Amz-
Security-Token=KwfLKSdEXAMPLE..."
}
```

For more information, see [Manage components](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [GetComponentVersionArtifact](#) in *AWS CLI Command Reference*.

get - component

The following code example shows how to use `get - component`.

AWS CLI

Example 1: To download a component's recipe in YAML format (Linux, macOS, or Unix)

The following `get - component` example downloads a Hello World component's recipe to a file in YAML format. This command does the following:

Uses the `--output` and `--query` parameters to control the command's output. These parameters extract the recipe blob from the command's output. For more information about controlling output, see [Controlling Command Output](#) in the *AWS Command Line Interface User Guide*. Uses the `base64` utility. This utility decodes the extracted blob to the original text. The blob that is returned by a successful `get - component` command is base64-encoded text. You must decode this blob to obtain the original text. Saves the decoded text to a file. The final section of the command (`> com.example.HelloWorld-1.0.0.json`) saves the decoded text to a file.

```
aws greengrassv2 get-component \
```



```

--arn arn:aws:greengrass:us-
west-2:123456789012:components:com.example.HelloWorld:versions:1.0.0 \
--recipe-output-format YAML \
--query recipe \
--output text | base64 --decode > com.example.HelloWorld-1.0.0.json

```

For more information, see [Manage components](#) in the *AWS IoT Greengrass V2 Developer Guide*.

Example 2: To download a component's recipe in YAML format (Windows CMD)

The following `get-component` example downloads a Hello World component's recipe to a file in YAML format. This command uses the `certutil` utility.

```

aws greengrassv2 get-component ^
  --arn arn:aws:greengrass:us-
west-2:675946970638:components:com.example.HelloWorld:versions:1.0.0 ^
  --recipe-output-format YAML ^
  --query recipe ^
  --output text > com.example.HelloWorld-1.0.0.yaml.b64

certutil -decode com.example.HelloWorld-1.0.0.yaml.b64
com.example.HelloWorld-1.0.0.yaml

```

For more information, see [Manage components](#) in the *AWS IoT Greengrass V2 Developer Guide*.

Example 3: To download a component's recipe in YAML format (Windows PowerShell)

The following `get-component` example downloads a Hello World component's recipe to a file in YAML format. This command uses the `certutil` utility.

```

aws greengrassv2 get-component `
  --arn arn:aws:greengrass:us-
west-2:675946970638:components:com.example.HelloWorld:versions:1.0.0 `
  --recipe-output-format YAML `
  --query recipe `
  --output text > com.example.HelloWorld-1.0.0.yaml.b64

certutil -decode com.example.HelloWorld-1.0.0.yaml.b64
com.example.HelloWorld-1.0.0.yaml

```

For more information, see [Manage components](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [GetComponent](#) in *AWS CLI Command Reference*.

get-connectivity-info

The following code example shows how to use `get-connectivity-info`.

AWS CLI

To get the connectivity information for a Greengrass core device

The following `get-connectivity-info` example gets the connectivity information for a Greengrass core device. Client devices use this information to connect to the MQTT broker that runs on this core device.

```
aws greengrassv2 get-connectivity-info \  
  --thing-name MyGreengrassCore
```

Output:

```
{  
  "connectivityInfo": [  
    {  
      "id": "localIP_192.0.2.0",  
      "hostAddress": "192.0.2.0",  
      "portNumber": 8883  
    }  
  ]  
}
```

For more information, see [Manage core device endpoints](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [GetConnectivityInfo](#) in *AWS CLI Command Reference*.

get-core-device

The following code example shows how to use `get-core-device`.

AWS CLI

To get a core device

The following `get-core-device` example gets information about an AWS IoT Greengrass core device.

```
aws greengrassv2 get-core-device \  
  --core-device-thing-name MyGreengrassCore
```

Output:

```
{  
  "coreDeviceThingName": "MyGreengrassCore",  
  "coreVersion": "2.0.3",  
  "platform": "linux",  
  "architecture": "amd64",  
  "status": "HEALTHY",  
  "lastStatusUpdateTimestamp": "2021-01-08T04:57:58.838000-08:00",  
  "tags": {}  
}
```

For more information, see [Check core device status](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [GetCoreDevice](#) in *AWS CLI Command Reference*.

get-deployment

The following code example shows how to use `get-deployment`.

AWS CLI

To get a deployment

The following `get-deployment` example gets information about the deployment of the AWS IoT Greengrass nucleus component to a group of core devices.

```
aws greengrassv2 get-deployment \  
  --deployment-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{  
  "targetArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/  
MyGreengrassCoreGroup",  
  "revisionId": "14",
```

```

"deploymentId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
"deploymentName": "Deployment for MyGreengrassCoreGroup",
"deploymentStatus": "ACTIVE",
"iotJobId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
"iotJobArn": "arn:aws:iot:us-west-2:123456789012:job/a1b2c3d4-5678-90ab-cdef-
EXAMPLE22222",
"components": {
  "aws.greengrass.Nucleus": {
    "componentVersion": "2.0.3",
    "configurationUpdate": {
      "merge": "{\"jvmOptions\":\"-Xmx64m\",\"logging\":{\"level\":\"WARN
\"}}\",
      "reset": [
        "/networkProxy",
        "/mqtt"
      ]
    }
  }
},
"deploymentPolicies": {
  "failureHandlingPolicy": "ROLLBACK",
  "componentUpdatePolicy": {
    "timeoutInSeconds": 60,
    "action": "NOTIFY_COMPONENTS"
  },
  "configurationValidationPolicy": {
    "timeoutInSeconds": 60
  }
},
"iotJobConfiguration": {},
"creationTimestamp": "2021-01-07T17:21:20.691000-08:00",
"isLatestForTarget": false,
"tags": {}
}

```

For more information, see [Deploy components to devices](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [GetDeployment](#) in *AWS CLI Command Reference*.

get-service-role-for-account

The following code example shows how to use `get-service-role-for-account`.

AWS CLI

To get the Greengrass service role for your AWS account

The following `get-service-role-for-account` example gets the service role that's associated with AWS IoT Greengrass for your AWS account.

```
aws greengrassv2 get-service-role-for-account
```

Output:

```
{
  "associatedAt": "2022-01-19T19:21:53Z",
  "roleArn": "arn:aws:iam::123456789012:role/service-role/Greengrass_ServiceRole"
}
```

For more information, see [Greengrass service role](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [GetServiceRoleForAccount](#) in *AWS CLI Command Reference*.

list-client-devices-associated-with-core-device

The following code example shows how to use `list-client-devices-associated-with-core-device`.

AWS CLI

To list the client devices associated with a core device

The following `list-client-devices-associated-with-core-device` example lists all client devices associated with a core device.

```
aws greengrassv2 list-client-devices-associated-with-core-device \
  --core-device-thing-name MyTestGreengrassCore
```

Output:

```
{
  "associatedClientDevices": [
```

```
{
  "thingName": "MyClientDevice2",
  "associationTimestamp": "2021-07-12T16:33:55.843000-07:00"
},
{
  "thingName": "MyClientDevice1",
  "associationTimestamp": "2021-07-12T16:33:55.843000-07:00"
}
]
```

For more information, see [Interact with local IoT devices](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [ListClientDevicesAssociatedWithCoreDevice](#) in *AWS CLI Command Reference*.

list-component-versions

The following code example shows how to use `list-component-versions`.

AWS CLI

To list the versions of a component

The following `list-component-versions` example lists all versions of a Hello World component.

```
aws greengrassv2 list-component-versions \
  --arn arn:aws:greengrass:us-
west-2:123456789012:components:com.example.HelloWorld
```

Output:

```
{
  "componentVersions": [
    {
      "componentName": "com.example.HelloWorld",
      "componentVersion": "1.0.1",
      "arn": "arn:aws:greengrass:us-
west-2:123456789012:components:com.example.HelloWorld:versions:1.0.1"
    },
```

```
    {
      "componentName": "com.example.HelloWorld",
      "componentVersion": "1.0.0",
      "arn": "arn:aws:greengrass:us-
west-2:123456789012:components:com.example.HelloWorld:versions:1.0.0"
    }
  ]
}
```

For more information, see [Manage components](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [ListComponentVersions](#) in *AWS CLI Command Reference*.

list-components

The following code example shows how to use `list-components`.

AWS CLI

To list components

The following `list-components` example lists each component and its latest version defined in your AWS account in the current Region.

```
aws greengrassv2 list-components
```

Output:

```
{
  "components": [
    {
      "arn": "arn:aws:greengrass:us-
west-2:123456789012:components:com.example.HelloWorld",
      "componentName": "com.example.HelloWorld",
      "latestVersion": {
        "arn": "arn:aws:greengrass:us-
west-2:123456789012:components:com.example.HelloWorld:versions:1.0.1",
        "componentVersion": "1.0.1",
        "creationTimestamp": "2021-01-08T16:51:07.352000-08:00",
        "description": "My first AWS IoT Greengrass component.",
        "publisher": "Amazon",
        "platforms": [
```

```
    {
      "attributes": {
        "os": "linux"
      }
    }
  ]
}
```

For more information, see [Manage components](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [ListComponents](#) in *AWS CLI Command Reference*.

list-core-devices

The following code example shows how to use `list-core-devices`.

AWS CLI

To list core devices

The following `list-core-devices` example lists the AWS IoT Greengrass core devices in your AWS account in the current Region.

```
aws greengrassv2 list-core-devices
```

Output:

```
{
  "coreDevices": [
    {
      "coreDeviceThingName": "MyGreengrassCore",
      "status": "HEALTHY",
      "lastStatusUpdateTimestamp": "2021-01-08T04:57:58.838000-08:00"
    }
  ]
}
```

For more information, see [Check core device status](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [ListCoreDevices](#) in *AWS CLI Command Reference*.

list-deployments

The following code example shows how to use `list-deployments`.

AWS CLI

To list deployments

The following `list-deployments` example lists the latest revision of each deployment defined in your AWS account in the current Region.

```
aws greengrassv2 list-deployments
```

Output:

```
{
  "deployments": [
    {
      "targetArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/MyGreengrassCoreGroup",
      "revisionId": "14",
      "deploymentId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "deploymentName": "Deployment for MyGreengrassCoreGroup",
      "creationTimestamp": "2021-01-07T17:21:20.691000-08:00",
      "deploymentStatus": "ACTIVE",
      "isLatestForTarget": false
    },
    {
      "targetArn": "arn:aws:iot:us-west-2:123456789012:thing/MyGreengrassCore",
      "revisionId": "1",
      "deploymentId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
      "deploymentName": "Deployment for MyGreengrassCore",
      "creationTimestamp": "2021-01-06T16:10:42.407000-08:00",
      "deploymentStatus": "COMPLETED",
      "isLatestForTarget": false
    }
  ]
}
```

For more information, see [Deploy components to devices](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [ListDeployments](#) in *AWS CLI Command Reference*.

list-effective-deployments

The following code example shows how to use `list-effective-deployments`.

AWS CLI

To list deployment jobs

The following `list-effective-deployments` example lists the deployments that apply to an AWS IoT Greengrass core device.

```
aws greengrassv2 list-effective-deployments \  
  --core-device-thing-name MyGreengrassCore
```

Output:

```
{  
  "effectiveDeployments": [  
    {  
      "deploymentId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
      "deploymentName": "Deployment for MyGreengrassCore",  
      "iotJobId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",  
      "targetArn": "arn:aws:iot:us-west-2:123456789012:thing/  
MyGreengrassCore",  
      "coreDeviceExecutionStatus": "COMPLETED",  
      "reason": "SUCCESSFUL",  
      "creationTimestamp": "2021-01-06T16:10:42.442000-08:00",  
      "modifiedTimestamp": "2021-01-08T17:21:27.830000-08:00"  
    },  
    {  
      "deploymentId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",  
      "deploymentName": "Deployment for MyGreengrassCoreGroup",  
      "iotJobId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE44444",  
      "iotJobArn": "arn:aws:iot:us-west-2:123456789012:job/a1b2c3d4-5678-90ab-  
cdef-EXAMPLE44444",  
      "targetArn": "arn:aws:iot:us-west-2:123456789012:thinggroup/  
MyGreengrassCoreGroup",  
      "coreDeviceExecutionStatus": "SUCCEEDED",  
    }  
  ]  
}
```

```
        "reason": "SUCCESSFUL",
        "creationTimestamp": "2021-01-07T17:19:20.394000-08:00",
        "modifiedTimestamp": "2021-01-07T17:21:20.721000-08:00"
    }
]
}
```

For more information, see [Check core device status](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [ListEffectiveDeployments](#) in *AWS CLI Command Reference*.

list-installed-components

The following code example shows how to use `list-installed-components`.

AWS CLI

To list components installed on a core device

The following `list-installed-components` example lists the components that are installed on an AWS IoT Greengrass core device.

```
aws greengrassv2 list-installed-components \
    --core-device-thing-name MyGreengrassCore
```

Output:

```
{
  "installedComponents": [
    {
      "componentName": "aws.greengrass.Cli",
      "componentVersion": "2.0.3",
      "lifecycleState": "RUNNING",
      "isRoot": true
    },
    {
      "componentName": "aws.greengrass.Nucleus",
      "componentVersion": "2.0.3",
      "lifecycleState": "FINISHED",
      "isRoot": true
    }
  ]
}
```

```
]
}
```

For more information, see [Check core device status](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [ListInstalledComponents](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags for a resource

The following `list-tags-for-resource` example lists all tags for an AWS IoT Greengrass core device.

```
aws greengrassv2 list-tags-for-resource \
  --resource-arn arn:aws:greengrass:us-
west-2:123456789012:coreDevices:MyGreengrassCore
```

Output:

```
{
  "tags": {
    "Owner": "richard-roe"
  }
}
```

For more information, see [Tag your resources](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add a tag to a resource

The following `tag-resource` example adds an owner tag to an AWS IoT Greengrass core device. You can use this tag to control access to the core device based on who owns it.

```
aws greengrassv2 tag-resource \  
  --resource-arn arn:aws:greengrass:us-  
west-2:123456789012:coreDevices:MyGreengrassCore \  
  --tags Owner=richard-roe
```

This command produces no output.

For more information, see [Tag your resources](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag from a resource

The following `untag-resource` example removes an owner tag from an AWS IoT Greengrass core device.

```
aws iotsitewise untag-resource \  
  --resource-arn arn:aws:greengrass:us-  
west-2:123456789012:coreDevices:MyGreengrassCore \  
  --tag-keys Owner
```

This command produces no output.

For more information, see [Tag your resources](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-connectivity-info

The following code example shows how to use `update-connectivity-info`.

AWS CLI

To update connectivity information for a Greengrass core device

The following `update-connectivity-info` example gets the connectivity information for a Greengrass core device. Client devices use this information to connect to the MQTT broker that runs on this core device.

```
aws greengrassv2 update-connectivity-info \  
  --thing-name MyGreengrassCore \  
  --cli-input-json file://core-device-connectivity-info.json
```

Contents of `core-device-connectivity-info.json`:

```
{  
  "connectivityInfo": [  
    {  
      "hostAddress": "192.0.2.0",  
      "portNumber": 8883,  
      "id": "localIP_192.0.2.0"  
    }  
  ]  
}
```

Output:

```
{  
  "version": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"  
}
```

For more information, see [Manage core device endpoints](#) in the *AWS IoT Greengrass V2 Developer Guide*.

- For API details, see [UpdateConnectivityInfo](#) in *AWS CLI Command Reference*.

AWS IoT Jobs SDK release examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS IoT Jobs SDK release.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

describe-job-execution

The following code example shows how to use `describe-job-execution`.

AWS CLI

To get the details of a job execution

The following `describe-job-execution` example retrieves the details of the latest execution of the specified job and thing.

```
aws iot-jobs-data describe-job-execution \  
  --job-id SampleJob \  
  --thing-name MotionSensor1 \  
  --endpoint-url https://1234567890abcd.jobs.iot.us-west-2.amazonaws.com
```

Output:

```
{  
  "execution": {  
    "approximateSecondsBeforeTimedOut": 88,  
    "executionNumber": 2939653338,  
    "jobId": "SampleJob",  
    "lastUpdatedAt": 1567701875.743,  
    "queuedAt": 1567701902.444,  
    "status": "QUEUED",  
    "thingName": "MotionSensor1 ",  
    "versionNumber": 3  
  }  
}
```

For more information, see [Devices and Jobs](#) in the *AWS IoT Developer Guide*.

- For API details, see [DescribeJobExecution](#) in *AWS CLI Command Reference*.

get-pending-job-executions

The following code example shows how to use `get-pending-job-executions`.

AWS CLI

To get a list of all jobs that are not in a terminal status for a thing

The following `get-pending-job-executions` example displays a list of all jobs that aren't in a terminal state for the specified thing.

```
aws iot-jobs-data get-pending-job-executions \  
  --thing-name MotionSensor1 \  
  --endpoint-url https://1234567890abcd.jobs.iot.us-west-2.amazonaws.com
```

Output:

```
{  
  "InProgressJobs": [  
  ],  
  "queuedJobs": [  
    {  
      "executionNumber": 2939653338,  
      "jobId": "SampleJob",  
      "lastUpdatedAt": 1567701875.743,  
      "queuedAt": 1567701902.444,  
      "versionNumber": 3  
    }  
  ]  
}
```

For more information, see [Devices and Jobs](#) in the *AWS IoT Developer Guide*.

- For API details, see [GetPendingJobExecutions](#) in *AWS CLI Command Reference*.

start-next-pending-job-execution

The following code example shows how to use `start-next-pending-job-execution`.

AWS CLI

To get and start the next pending job execution for a thing

The following `start-next-pending-job-execution` example retrieves and starts the next job execution whose status is `IN_PROGRESS` or `QUEUED` for the specified thing.

```
aws iot-jobs-data start-next-pending-job-execution \  
  --thing-name MotionSensor1 \  
  --endpoint-url https://1234567890abcd.jobs.iot.us-west-2.amazonaws.com
```

Output:

```
{  
  "execution": {  
    "approximateSecondsBeforeTimedOut": 88,  
    "executionNumber": 2939653338,  
    "jobId": "SampleJob",  
    "lastUpdatedAt": 1567714853.743,  
    "queuedAt": 1567701902.444,  
    "startedAt": 1567714871.690,  
    "status": "IN_PROGRESS",  
    "thingName": "MotionSensor1 ",  
    "versionNumber": 3  
  }  
}
```

For more information, see [Devices and Jobs](#) in the *AWS IoT Developer Guide*.

- For API details, see [StartNextPendingJobExecution](#) in *AWS CLI Command Reference*.

update-job-execution

The following code example shows how to use `update-job-execution`.

AWS CLI

To update the status of a job execution

The following `update-job-execution` example updates the status of the specified job and thing.

```
aws iot-jobs-data update-job-execution \  
  --job-id SampleJob \  
  --thing-name MotionSensor1 \  
  --status IN_PROGRESS
```

```
--job-id SampleJob \  
--thing-name MotionSensor1 \  
--status REMOVED \  
--endpoint-url https://1234567890abcd.jobs.iot.us-west-2.amazonaws.com
```

Output:

```
{  
  "executionState": {  
    "status": "REMOVED",  
    "versionNumber": 3  
  },  
}
```

For more information, see [Devices and Jobs](#) in the *AWS IoT Developer Guide*.

- For API details, see [UpdateJobExecution](#) in *AWS CLI Command Reference*.

AWS IoT SiteWise examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS IoT SiteWise.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-assets

The following code example shows how to use `associate-assets`.

AWS CLI

To associate a child asset to a parent asset

The following `associate-assets` example associates a wind turbine asset to a wind farm asset, where the wind turbine asset model exists as a hierarchy in the wind farm asset model.

```
aws iotsitewise associate-assets \  
  --asset-id a1b2c3d4-5678-90ab-cdef-44444EXAMPLE \  
  --hierarchy-id a1b2c3d4-5678-90ab-cdef-77777EXAMPLE \  
  --child-asset-id a1b2c3d4-5678-90ab-cdef-33333EXAMPLE
```

This command produces no output.

For more information, see [Associating assets](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [AssociateAssets](#) in *AWS CLI Command Reference*.

batch-associate-project-assets

The following code example shows how to use `batch-associate-project-assets`.

AWS CLI

To associate an asset to a project

The following `batch-associate-project-assets` example associates a wind farm asset to a project.

```
aws iotsitewise batch-associate-project-assets \  
  --project-id a1b2c3d4-5678-90ab-cdef-eeeeeeEXAMPLE \  
  --asset-ids a1b2c3d4-5678-90ab-cdef-44444EXAMPLE
```

This command produces no output.

For more information, see [Adding assets to projects](#) in the *AWS IoT SiteWise Monitor Application Guide*.

- For API details, see [BatchAssociateProjectAssets](#) in *AWS CLI Command Reference*.

batch-disassociate-project-assets

The following code example shows how to use `batch-disassociate-project-assets`.

AWS CLI

To disassociate an asset from a project

The following `batch-disassociate-project-assets` example disassociates a wind farm asset from a project.

```
aws iotsitewise batch-disassociate-project-assets \  
  --project-id a1b2c3d4-5678-90ab-cdef-eeeeEXAMPLE \  
  --asset-ids a1b2c3d4-5678-90ab-cdef-44444EXAMPLE
```

This command produces no output.

For more information, see [Adding assets to projects](#) in the *AWS IoT SiteWise Monitor Application Guide*.

- For API details, see [BatchDisassociateProjectAssets](#) in *AWS CLI Command Reference*.

batch-put-asset-property-value

The following code example shows how to use `batch-put-asset-property-value`.

AWS CLI

To send data to asset properties

The following `batch-put-asset-property-value` example sends power and temperature data to the asset properties identified by property aliases.

```
aws iotsitewise batch-put-asset-property-value \  
  --cli-input-json file://batch-put-asset-property-value.json
```

Contents of `batch-put-asset-property-value.json`:

```
{  
  "entries": [  
    {  
      "entryId": "1575691200-company-windfarm-3-turbine-7-power",  
      "propertyAlias": "company-windfarm-3-turbine-7-power",  
      "propertyValues": [  
        {  
          "value": {  
            "doubleValue": 4.92          }  
        }  
      ]  
    }  
  ]  
}
```

```

    },
    "timestamp": {
      "timeInSeconds": 1575691200
    },
    "quality": "GOOD"
  }
]
},
{
  "entryId": "1575691200-company-windfarm-3-turbine-7-temperature",
  "propertyAlias": "company-windfarm-3-turbine-7-temperature",
  "propertyValues": [
    {
      "value": {
        "integerValue": 38
      },
      "timestamp": {
        "timeInSeconds": 1575691200
      }
    }
  ]
}
]
}
}

```

Output:

```

{
  "errorEntries": []
}

```

For more information, see [Ingesting data using the AWS IoT SiteWise API](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [BatchPutAssetPropertyValue](#) in *AWS CLI Command Reference*.

create-access-policy

The following code example shows how to use `create-access-policy`.

AWS CLI**Example 1: To grant a user administrative access to a portal**

The following `create-access-policy` example creates an access policy that grants a user administrative access to a web portal for a wind farm company.

```
aws iotsitewise create-access-policy \  
  --cli-input-json file://create-portal-administrator-access-policy.json
```

Contents of `create-portal-administrator-access-policy.json`:

```
{  
  "accessPolicyIdentity": {  
    "user": {  
      "id": "a1b2c3d4e5-a1b2c3d4-5678-90ab-cdef-bbbbbbEXAMPLE"  
    }  
  },  
  "accessPolicyPermission": "ADMINISTRATOR",  
  "accessPolicyResource": {  
    "portal": {  
      "id": "a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE"  
    }  
  }  
}
```

Output:

```
{  
  "accessPolicyId": "a1b2c3d4-5678-90ab-cdef-cccccEXAMPLE",  
  "accessPolicyArn": "arn:aws:iotsitewise:us-west-2:123456789012:access-policy/  
a1b2c3d4-5678-90ab-cdef-cccccEXAMPLE"  
}
```

For more information, see [Adding or removing portal administrators](#) in the *AWS IoT SiteWise User Guide*.

Example 2: To grant a user read-only access to a project

The following `create-access-policy` example creates an access policy that grants a user read-only access to a wind farm project.

```
aws iotsitewise create-access-policy \  
  --cli-input-json file://create-project-viewer-access-policy.json
```

Contents of create-project-viewer-access-policy.json:

```
{
  "accessPolicyIdentity": {
    "user": {
      "id": "a1b2c3d4e5-a1b2c3d4-5678-90ab-cdef-bbbbbEXAMPLE"
    }
  },
  "accessPolicyPermission": "VIEWER",
  "accessPolicyResource": {
    "project": {
      "id": "a1b2c3d4-5678-90ab-cdef-eeeeEXAMPLE"
    }
  }
}
```

Output:

```
{
  "accessPolicyId": "a1b2c3d4-5678-90ab-cdef-dddddEXAMPLE",
  "accessPolicyArn": "arn:aws:iotsitewise:us-west-2:123456789012:access-policy/a1b2c3d4-5678-90ab-cdef-dddddEXAMPLE"
}
```

For more information, see [Assigning project viewers](#) in the *AWS IoT SiteWise Monitor Application Guide*.

- For API details, see [CreateAccessPolicy](#) in *AWS CLI Command Reference*.

create-asset-model

The following code example shows how to use create-asset-model.

AWS CLI

To create an asset model

The following create-asset-model example creates an asset model that defines a wind turbine with the following properties:

Serial number - The serial number of a wind turbine
Generated power - The generated power data stream from a wind turbine
Temperature C - The temperature data stream from a wind

turbine in CelsiusTemperature F - The mapped temperature data points from Celsius to Fahrenheit

```
aws iotsitewise create-asset-model \  
  --cli-input-json file://create-wind-turbine-model.json
```

Contents of create-wind-turbine-model.json:

```
{  
  "assetModelName": "Wind Turbine Model",  
  "assetModelDescription": "Represents a wind turbine",  
  "assetModelProperties": [  
    {  
      "name": "Serial Number",  
      "dataType": "STRING",  
      "type": {  
        "attribute": {}  
      }  
    },  
    {  
      "name": "Generated Power",  
      "dataType": "DOUBLE",  
      "unit": "kW",  
      "type": {  
        "measurement": {}  
      }  
    },  
    {  
      "name": "Temperature C",  
      "dataType": "DOUBLE",  
      "unit": "Celsius",  
      "type": {  
        "measurement": {}  
      }  
    },  
    {  
      "name": "Temperature F",  
      "dataType": "DOUBLE",  
      "unit": "Fahrenheit",  
      "type": {  
        "transform": {  
          "expression": "temp_c * 9 / 5 + 32",  
          "variables": [  

```



```

        {
            "name": "temp_c",
            "value": {
                "propertyId": "Temperature C"
            }
        }
    ]
}
},
{
    "name": "Total Generated Power",
    "dataType": "DOUBLE",
    "unit": "kW",
    "type": {
        "metric": {
            "expression": "sum(power)",
            "variables": [
                {
                    "name": "power",
                    "value": {
                        "propertyId": "Generated Power"
                    }
                }
            ],
            "window": {
                "tumbling": {
                    "interval": "1h"
                }
            }
        }
    }
}
]
}

```

Output:

```

{
    "assetModelId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "assetModelArn": "arn:aws:iotsitewise:us-west-2:123456789012:asset-model/a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "assetModelStatus": {

```

```
    "state": "CREATING"
  }
}
```

For more information, see [Defining asset models](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [CreateAssetModel](#) in *AWS CLI Command Reference*.

create-asset

The following code example shows how to use `create-asset`.

AWS CLI

To create an asset

The following `create-asset` example creates a wind turbine asset from a wind turbine asset model.

```
aws iotsitewise create-asset \
  --asset-model-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \
  --asset-name "Wind Turbine 1"
```

Output:

```
{
  "assetId": "a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
  "assetArn": "arn:aws:iotsitewise:us-west-2:123456789012:asset/
a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
  "assetStatus": {
    "state": "CREATING"
  }
}
```

For more information, see [Creating assets](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [CreateAsset](#) in *AWS CLI Command Reference*.

create-dashboard

The following code example shows how to use `create-dashboard`.

AWS CLI

To create a dashboard

The following `create-dashboard` example creates a dashboard with a line chart that displays total generated power for a wind farm.

```
aws iotsitewise create-dashboard \  
  --project-id a1b2c3d4-5678-90ab-cdef-eeeeeeEXAMPLE \  
  --dashboard-name "Wind Farm" \  
  --dashboard-definition file://create-wind-farm-dashboard.json
```

Contents of `create-wind-farm-dashboard.json`:

```
{  
  "widgets": [  
    {  
      "type": "monitor-line-chart",  
      "title": "Generated Power",  
      "x": 0,  
      "y": 0,  
      "height": 3,  
      "width": 3,  
      "metrics": [  
        {  
          "label": "Power",  
          "type": "iotsitewise",  
          "assetId": "a1b2c3d4-5678-90ab-cdef-44444EXAMPLE",  
          "propertyId": "a1b2c3d4-5678-90ab-cdef-99999EXAMPLE"  
        }  
      ]  
    }  
  ]  
}
```

Output:

```
{  
  "dashboardId": "a1b2c3d4-5678-90ab-cdef-ffffffEXAMPLE",  
  "dashboardArn": "arn:aws:iotsitewise:us-west-2:123456789012:dashboard/  
a1b2c3d4-5678-90ab-cdef-ffffffEXAMPLE"  
}
```

For more information, see [Creating dashboards \(CLI\)](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [CreateDashboard](#) in *AWS CLI Command Reference*.

create-gateway

The following code example shows how to use `create-gateway`.

AWS CLI

To create a gateway

The following `create-gateway` example creates a gateway that runs on AWS IoT Greengrass.

```
aws iotsitewise create-gateway \  
  --gateway-name ExampleCorpGateway \  
  --gateway-platform greengrass={groupArn=arn:aws:greengrass:us-  
west-2:123456789012:/greengrass/groups/a1b2c3d4-5678-90ab-cdef-1b1b1EXAMPLE}
```

Output:

```
{  
  "gatewayId": "a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE",  
  "gatewayArn": "arn:aws:iotsitewise:us-west-2:123456789012:gateway/  
a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE"  
}
```

For more information, see [Configuring a gateway](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [CreateGateway](#) in *AWS CLI Command Reference*.

create-portal

The following code example shows how to use `create-portal`.

AWS CLI

To create a portal

The following `create-portal` example creates a web portal for a wind farm company. You can create portals only in the same Region where you enabled AWS Single Sign-On.

```
aws iotsitewise create-portal \  
  --portal-name WindFarmPortal \  
  --portal-description "A portal that contains wind farm projects for Example  
Corp." \  
  --portal-contact-email support@example.com \  
  --role-arn arn:aws:iam::123456789012:role/service-role/  
MySiteWiseMonitorServiceRole
```

Output:

```
{  
  "portalId": "a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE",  
  "portalArn": "arn:aws:iotsitewise:us-west-2:123456789012:portal/  
a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE",  
  "portalStartUrl": "https://a1b2c3d4-5678-90ab-cdef-  
aaaaaEXAMPLE.app.iotsitewise.aws",  
  "portalStatus": {  
    "state": "CREATING"  
  },  
  "ssoApplicationId": "ins-a1b2c3d4-EXAMPLE"  
}
```

For more information, see [Getting started with AWS IoT SiteWise Monitor](#) in the *AWS IoT SiteWise User Guide* and [Enabling AWS SSO](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [CreatePortal](#) in *AWS CLI Command Reference*.

create-project

The following code example shows how to use `create-project`.

AWS CLI

To create a project

The following `create-project` example creates a wind farm project.

```
aws iotsitewise create-project \  
  --portal-id a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE \  
  --project-name "Wind Farm 1" \  
  --project-description "Contains asset visualizations for Wind Farm #1 for  
Example Corp."
```

Output:

```
{
  "projectId": "a1b2c3d4-5678-90ab-cdef-eeeeeeEXAMPLE",
  "projectArn": "arn:aws:iotsitewise:us-west-2:123456789012:project/
a1b2c3d4-5678-90ab-cdef-eeeeeeEXAMPLE"
}
```

For more information, see [Creating projects](#) in the *AWS IoT SiteWise Monitor Application Guide*.

- For API details, see [CreateProject](#) in *AWS CLI Command Reference*.

delete-access-policy

The following code example shows how to use `delete-access-policy`.

AWS CLI**To revoke a user's access to a project or portal**

The following `delete-access-policy` example deletes an access policy that grants a user administrative access to a portal.

```
aws iotsitewise delete-access-policy \
  --access-policy-id a1b2c3d4-5678-90ab-cdef-ccccEXAMPLE
```

This command produces no output.

For more information, see [Adding or removing portal administrators](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [DeleteAccessPolicy](#) in *AWS CLI Command Reference*.

delete-asset-model

The following code example shows how to use `delete-asset-model`.

AWS CLI**To delete an asset model**

The following `delete-asset-model` example deletes a wind turbine asset model.

```
aws iotsitewise delete-asset-model \  
  --asset-model-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
```

Output:

```
{  
  "assetModelStatus": {  
    "state": "DELETING"  
  }  
}
```

For more information, see [Deleting asset models](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [DeleteAssetModel](#) in *AWS CLI Command Reference*.

delete-asset

The following code example shows how to use `delete-asset`.

AWS CLI

To delete an asset

The following `delete-asset` example deletes a wind turbine asset.

```
aws iotsitewise delete-asset \  
  --asset-id a1b2c3d4-5678-90ab-cdef-33333EXAMPLE
```

Output:

```
{  
  "assetStatus": {  
    "state": "DELETING"  
  }  
}
```

For more information, see [Deleting assets](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [DeleteAsset](#) in *AWS CLI Command Reference*.

delete-dashboard

The following code example shows how to use `delete-dashboard`.

AWS CLI

To delete a dashboard

The following `delete-dashboard` example deletes a wind turbine dashboard.

```
aws iotsitewise delete-dashboard \  
  --dashboard-id a1b2c3d4-5678-90ab-cdef-fffffEXAMPLE
```

This command produces no output.

For more information, see [Deleting dashboards](#) in the *AWS IoT SiteWise Monitor Application Guide*.

- For API details, see [DeleteDashboard](#) in *AWS CLI Command Reference*.

delete-gateway

The following code example shows how to use `delete-gateway`.

AWS CLI

To delete a gateway

The following `delete-gateway` example deletes a gateway.

```
aws iotsitewise delete-gateway \  
  --gateway-id a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE
```

This command produces no output.

For more information, see [Ingesting data using a gateway](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [DeleteGateway](#) in *AWS CLI Command Reference*.

delete-portal

The following code example shows how to use `delete-portal`.

AWS CLI

To delete a portal

The following `delete-portal` example deletes a web portal for a wind farm company.

```
aws iotsitewise delete-portal \  
  --portal-id a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE
```

Output:

```
{  
  "portalStatus": {  
    "state": "DELETING"  
  }  
}
```

For more information, see [Deleting a portal](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [DeletePortal](#) in *AWS CLI Command Reference*.

delete-project

The following code example shows how to use `delete-project`.

AWS CLI

To delete a project

The following `delete-project` example deletes a wind farm project.

```
aws iotsitewise delete-project \  
  --project-id a1b2c3d4-5678-90ab-cdef-eeeeeeEXAMPLE
```

This command produces no output.

For more information, see [Deleting projects](#) in the *AWS IoT SiteWise Monitor Application Guide*.

- For API details, see [DeleteProject](#) in *AWS CLI Command Reference*.

describe-access-policy

The following code example shows how to use `describe-access-policy`.

AWS CLI

To describe an access policy

The following `describe-access-policy` example describes an access policy that grants a user administrative access to a web portal for a wind farm company.

```
aws iotsitewise describe-access-policy \  
  --access-policy-id a1b2c3d4-5678-90ab-cdef-ccccEXAMPLE
```

Output:

```
{  
  "accessPolicyId": "a1b2c3d4-5678-90ab-cdef-ccccEXAMPLE",  
  "accessPolicyArn": "arn:aws:iotsitewise:us-west-2:123456789012:access-policy/  
a1b2c3d4-5678-90ab-cdef-ccccEXAMPLE",  
  "accessPolicyIdentity": {  
    "user": {  
      "id": "a1b2c3d4e5-a1b2c3d4-5678-90ab-cdef-bbbbbbEXAMPLE"  
    }  
  },  
  "accessPolicyResource": {  
    "portal": {  
      "id": "a1b2c3d4-5678-90ab-cdef-aaaaEXAMPLE"  
    }  
  },  
  "accessPolicyPermission": "ADMINISTRATOR",  
  "accessPolicyCreationDate": "2020-02-20T22:35:15.552880124Z",  
  "accessPolicyLastUpdateDate": "2020-02-20T22:35:15.552880124Z"  
}
```

For more information, see [Adding or removing portal administrators](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [DescribeAccessPolicy](#) in *AWS CLI Command Reference*.

describe-asset-model

The following code example shows how to use `describe-asset-model`.

AWS CLI

To describe an asset model

The following `describe-asset-model` example describes a wind farm asset model.

```
aws iotsitewise describe-asset-model \  
  --asset-model-id a1b2c3d4-5678-90ab-cdef-22222EXAMPLE
```

Output:

```
{  
  "assetModelId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",  
  "assetModelArn": "arn:aws:iotsitewise:us-west-2:123456789012:asset-model/  
a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",  
  "assetModelName": "Wind Farm Model",  
  "assetModelDescription": "Represents a wind farm that comprises many wind  
turbines",  
  "assetModelProperties": [  
    {  
      "id": "a1b2c3d4-5678-90ab-cdef-99999EXAMPLE",  
      "name": "Total Generated Power",  
      "dataType": "DOUBLE",  
      "unit": "kW",  
      "type": {  
        "metric": {  
          "expression": "sum(power)",  
          "variables": [  
            {  
              "name": "power",  
              "value": {  
                "propertyId": "a1b2c3d4-5678-90ab-  
cdef-66666EXAMPLE",  
                "hierarchyId": "a1b2c3d4-5678-90ab-  
cdef-77777EXAMPLE"  
              }  
            }  
          ],  
          "window": {
```

```

        "tumbling": {
            "interval": "1h"
        }
    }
},
{
    "id": "a1b2c3d4-5678-90ab-cdef-88888EXAMPLE",
    "name": "Region",
    "dataType": "STRING",
    "type": {
        "attribute": {
            "defaultValue": " "
        }
    }
},
],
"assetModelHierarchies": [
    {
        "id": "a1b2c3d4-5678-90ab-cdef-77777EXAMPLE",
        "name": "Wind Turbines",
        "childAssetModelId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"
    }
],
"assetModelCreationDate": 1575671284.0,
"assetModelLastUpdateDate": 1575671988.0,
"assetModelStatus": {
    "state": "ACTIVE"
}
}

```

For more information, see [Describing a specific asset model](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [DescribeAssetModel](#) in *AWS CLI Command Reference*.

describe-asset-property

The following code example shows how to use describe-asset-property.

AWS CLI

To describe an asset property

The following `describe-asset-property` example describes a wind farm asset's total generated power property.

```
aws iotsitewise describe-asset-property \  
  --asset-id a1b2c3d4-5678-90ab-cdef-44444EXAMPLE \  
  --property-id a1b2c3d4-5678-90ab-cdef-99999EXAMPLE
```

Output:

```
{  
  "assetId": "a1b2c3d4-5678-90ab-cdef-44444EXAMPLE",  
  "assetName": "Wind Farm 1",  
  "assetModelId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",  
  "assetProperty": {  
    "id": "a1b2c3d4-5678-90ab-cdef-99999EXAMPLE",  
    "name": "Total Generated Power",  
    "notification": {  
      "topic": "$aws/sitewise/asset-models/a1b2c3d4-5678-90ab-cdef-22222EXAMPLE/assets/a1b2c3d4-5678-90ab-cdef-44444EXAMPLE/properties/a1b2c3d4-5678-90ab-cdef-99999EXAMPLE",  
      "state": "DISABLED"  
    },  
    "dataType": "DOUBLE",  
    "unit": "kW",  
    "type": {  
      "metric": {  
        "expression": "sum(power)",  
        "variables": [  
          {  
            "name": "power",  
            "value": {  
              "propertyId": "a1b2c3d4-5678-90ab-cdef-66666EXAMPLE",  
              "hierarchyId": "a1b2c3d4-5678-90ab-cdef-77777EXAMPLE"  
            }  
          }  
        ],  
        "window": {  
          "tumbling": {  
            "interval": "1h"  
          }  
        }  
      }  
    }  
  }  
}
```

```
}  
}
```

For more information, see [Describing a specific asset property](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [DescribeAssetProperty](#) in *AWS CLI Command Reference*.

describe-asset

The following code example shows how to use `describe-asset`.

AWS CLI

To describe an asset

The following `describe-asset` example describes a wind farm asset.

```
aws iotsitewise describe-asset \  
  --asset-id a1b2c3d4-5678-90ab-cdef-44444EXAMPLE
```

Output:

```
{  
  "assetId": "a1b2c3d4-5678-90ab-cdef-44444EXAMPLE",  
  "assetArn": "arn:aws:iotsitewise:us-west-2:123456789012:asset/  
a1b2c3d4-5678-90ab-cdef-44444EXAMPLE",  
  "assetName": "Wind Farm 1",  
  "assetModelId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",  
  "assetProperties": [  
    {  
      "id": "a1b2c3d4-5678-90ab-cdef-88888EXAMPLE",  
      "name": "Region",  
      "dataType": "STRING"  
    },  
    {  
      "id": "a1b2c3d4-5678-90ab-cdef-99999EXAMPLE",  
      "name": "Total Generated Power",  
      "dataType": "DOUBLE",  
      "unit": "kW"  
    }  
  ],  
}
```

```

    "assetHierarchies": [
      {
        "id": "a1b2c3d4-5678-90ab-cdef-77777EXAMPLE",
        "name": "Wind Turbines"
      }
    ],
    "assetCreationDate": 1575672453.0,
    "assetLastUpdateDate": 1575672453.0,
    "assetStatus": {
      "state": "ACTIVE"
    }
  }
}

```

For more information, see [Describing a specific asset](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [DescribeAsset](#) in *AWS CLI Command Reference*.

describe-dashboard

The following code example shows how to use describe-dashboard.

AWS CLI

To describe a dashboard

The following describe-dashboard example describes the specified wind farm dashboard.

```

aws iotsitewise describe-dashboard \
  --dashboard-id a1b2c3d4-5678-90ab-cdef-fffffEXAMPLE

```

Output:

```

{
  "dashboardId": "a1b2c3d4-5678-90ab-cdef-fffffEXAMPLE",
  "dashboardArn": "arn:aws:iotsitewise:us-west-2:123456789012:dashboard/a1b2c3d4-5678-90ab-cdef-fffffEXAMPLE",
  "dashboardName": "Wind Farm",
  "projectId": "a1b2c3d4-5678-90ab-cdef-eeeeeeEXAMPLE",
  "dashboardDefinition": "{\"widgets\": [{\"type\": \"monitor-line-chart\", \"title\": \"Generated Power\", \"x\": 0, \"y\": 0, \"height\": 3, \"width\": 3, \"metrics\": [{\"label\": \"Power\", \"type\": \"iotsitewise\", \"assetId\": \"a1b2c3d4-5678-90ab-cdef-44444EXAMPLE\", \"propertyId\": \"a1b2c3d4-5678-90ab-cdef-99999EXAMPLE\"}]}]}",
  "dashboardCreationDate": "2020-05-01T20:32:12.228476348Z",
}

```

```
"dashboardLastUpdateDate": "2020-05-01T20:32:12.228476348Z"
}
```

For more information, see [Viewing dashboards](#) in the *AWS IoT SiteWise Monitor Application Guide*.

- For API details, see [DescribeDashboard](#) in *AWS CLI Command Reference*.

describe-gateway-capability-configuration

The following code example shows how to use `describe-gateway-capability-configuration`.

AWS CLI

To describe a gateway capability

The following `describe-gateway-capability-configuration` example describes an OPC-UA source capability.

```
aws iotsitewise describe-gateway-capability-configuration \
  --gateway-id a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE \
  --capability-namespace "iotsitewise:opcuacollector:1"
```

Output:

```
{
  "gatewayId": "a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE",
  "capabilityNamespace": "iotsitewise:opcuacollector:1",
  "capabilityConfiguration": "{\"sources\": [{\"name\": \"Wind Farm #1\",
  \"endpoint\": {\"certificateTrust\": {\"type\": \"TrustAny\"}, \"endpointUri\": \"opc.tcp://203.0.113.0:49320\", \"securityPolicy\": \"BASIC256\",
  \"messageSecurityMode\": \"SIGN_AND_ENCRYPT\", \"identityProvider\": {\"type\": \"Username\", \"usernameSecretArn\": \"arn:aws:secretsmanager:us-east-1:123456789012:secret:greenrass-factory1-auth-3QNDmM\"}, \"nodeFilterRules\": []}, \"measurementDataStreamPrefix\": \"\"}]}",
  "capabilitySyncStatus": "IN_SYNC"
}
```

For more information, see [Configuring data sources](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [DescribeGatewayCapabilityConfiguration](#) in *AWS CLI Command Reference*.

describe-gateway

The following code example shows how to use `describe-gateway`.

AWS CLI

To describe a gateway

The following `describe-gateway` example describes a gateway.

```
aws iotsitewise describe-gateway \  
  --gateway-id a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE
```

Output:

```
{  
  "gatewayId": "a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE",  
  "gatewayName": "ExampleCorpGateway",  
  "gatewayArn": "arn:aws:iotsitewise:us-west-2:123456789012:gateway/  
a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE",  
  "gatewayPlatform": {  
    "greengrass": {  
      "groupArn": "arn:aws:greengrass:us-west-2:123456789012:/greengrass/  
groups/a1b2c3d4-5678-90ab-cdef-1b1b1EXAMPLE"  
    }  
  },  
  "gatewayCapabilitySummaries": [  
    {  
      "capabilityNamespace": "iotsitewise:opcuacollector:1",  
      "capabilitySyncStatus": "IN_SYNC"  
    }  
  ],  
  "creationDate": 1588369971.457,  
  "lastUpdateDate": 1588369971.457  
}
```

For more information, see [Ingesting data using a gateway](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [DescribeGateway](#) in *AWS CLI Command Reference*.

describe-logging-options

The following code example shows how to use `describe-logging-options`.

AWS CLI

To retrieve the current AWS IoT SiteWise logging options

The following `describe-logging-options` example retrieves the current AWS IoT SiteWise logging options for your AWS account in the current Region.

```
aws iotsitewise describe-logging-options
```

Output:

```
{
  "loggingOptions": {
    "level": "INFO"
  }
}
```

For more information, see [Monitoring AWS IoT SiteWise with Amazon CloudWatch Logs](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [DescribeLoggingOptions](#) in *AWS CLI Command Reference*.

describe-portal

The following code example shows how to use `describe-portal`.

AWS CLI

To describe a portal

The following `describe-portal` example describes a web portal for a wind farm company.

```
aws iotsitewise describe-portal \
  --portal-id a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE
```

Output:

```
{
  "portalId": "a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE",
  "portalArn": "arn:aws:iotsitewise:us-west-2:123456789012:portal/
a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE",
  "portalName": "WindFarmPortal",
}
```

```
"portalDescription": "A portal that contains wind farm projects for Example Corp.",
"portalClientId": "E-a1b2c3d4e5f6_a1b2c3d4e5f6EXAMPLE",
"portalStartUrl": "https://a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE.app.iotsitewise.aws",
"portalContactEmail": "support@example.com",
"portalStatus": {
  "state": "ACTIVE"
},
"portalCreationDate": "2020-02-04T23:01:52.90248068Z",
"portalLastUpdateDate": "2020-02-04T23:01:52.90248078Z",
"roleArn": "arn:aws:iam::123456789012:role/MySiteWiseMonitorServiceRole"
}
```

For more information, see [Administering your portals](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [DescribePortal](#) in *AWS CLI Command Reference*.

describe-project

The following code example shows how to use `describe-project`.

AWS CLI

To describe a project

The following `describe-project` example describes a wind farm project.

```
aws iotsitewise describe-project \
  --project-id a1b2c3d4-5678-90ab-cdef-eeeeeeEXAMPLE
```

Output:

```
{
  "projectId": "a1b2c3d4-5678-90ab-cdef-eeeeeeEXAMPLE",
  "projectArn": "arn:aws:iotsitewise:us-west-2:123456789012:project/a1b2c3d4-5678-90ab-cdef-eeeeeeEXAMPLE",
  "projectName": "Wind Farm 1",
  "portalId": "a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE",
  "projectDescription": "Contains asset visualizations for Wind Farm #1 for Example Corp.",
  "projectCreationDate": "2020-02-20T21:58:43.362246001Z",
  "projectLastUpdateDate": "2020-02-20T21:58:43.362246095Z"
```

```
}
```

For more information, see [Viewing project details](#) in the *AWS IoT SiteWise Monitor Application Guide*.

- For API details, see [DescribeProject](#) in *AWS CLI Command Reference*.

disassociate-assets

The following code example shows how to use `disassociate-assets`.

AWS CLI

To disassociate a child asset from a parent asset

The following `disassociate-assets` example disassociates a wind turbine asset from a wind farm asset.

```
aws iotsitewise disassociate-assets \  
  --asset-id a1b2c3d4-5678-90ab-cdef-44444EXAMPLE \  
  --hierarchy-id a1b2c3d4-5678-90ab-cdef-77777EXAMPLE \  
  --child-asset-id a1b2c3d4-5678-90ab-cdef-33333EXAMPLE
```

This command produces no output.

For more information, see [Associating assets](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [DisassociateAssets](#) in *AWS CLI Command Reference*.

get-asset-property-aggregates

The following code example shows how to use `get-asset-property-aggregates`.

AWS CLI

To retrieve an asset property's aggregated average and count values

The following `get-asset-property-aggregates` example retrieves a wind turbine asset's average total power and count of total power data points for a 1 hour period in time.

```
aws iotsitewise get-asset-property-aggregates \  
  --asset-id a1b2c3d4-5678-90ab-cdef-33333EXAMPLE \  
  --
```

```
--property-id a1b2c3d4-5678-90ab-cdef-66666EXAMPLE \  
--start-date 1580849400 \  
--end-date 1580853000 \  
--aggregate-types AVERAGE COUNT \  
--resolution 1h
```

Output:

```
{  
  "aggregatedValues": [  
    {  
      "timestamp": 1580850000.0,  
      "quality": "GOOD",  
      "value": {  
        "average": 8723.46538886233,  
        "count": 12.0  
      }  
    }  
  ]  
}
```

For more information, see [Querying asset property aggregates](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [GetAssetPropertyAggregates](#) in *AWS CLI Command Reference*.

get-asset-property-value-history

The following code example shows how to use `get-asset-property-value-history`.

AWS CLI

To retrieve an asset property's historical values

The following `get-asset-property-value-history` example retrieves a wind turbine asset's total power values for a 20 minute period in time.

```
aws iotsitewise get-asset-property-value-history \  
  --asset-id a1b2c3d4-5678-90ab-cdef-33333EXAMPLE \  
  --property-id a1b2c3d4-5678-90ab-cdef-66666EXAMPLE \  
  --start-date 1580851800 \  
  --end-date 1580853000
```

Output:

```
{
  "assetPropertyValueHistory": [
    {
      "value": {
        "doubleValue": 7217.787046814844
      },
      "timestamp": {
        "timeInSeconds": 1580852100,
        "offsetInNanos": 0
      },
      "quality": "GOOD"
    },
    {
      "value": {
        "doubleValue": 6941.242811875451
      },
      "timestamp": {
        "timeInSeconds": 1580852400,
        "offsetInNanos": 0
      },
      "quality": "GOOD"
    },
    {
      "value": {
        "doubleValue": 6976.797662266717
      },
      "timestamp": {
        "timeInSeconds": 1580852700,
        "offsetInNanos": 0
      },
      "quality": "GOOD"
    },
    {
      "value": {
        "doubleValue": 6890.8677520453875
      },
      "timestamp": {
        "timeInSeconds": 1580853000,
        "offsetInNanos": 0
      },
      "quality": "GOOD"
    }
  ]
}
```

```
]
}
```

For more information, see [Querying historical asset property values](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [GetAssetPropertyValueHistory](#) in *AWS CLI Command Reference*.

get-asset-property-value

The following code example shows how to use `get-asset-property-value`.

AWS CLI

To retrieve an asset property's current value

The following `get-asset-property-value` example retrieves a wind turbine asset's current total power.

```
aws iotsitewise get-asset-property-value \
  --asset-id a1b2c3d4-5678-90ab-cdef-33333EXAMPLE \
  --property-id a1b2c3d4-5678-90ab-cdef-66666EXAMPLE
```

Output:

```
{
  "propertyValue": {
    "value": {
      "doubleValue": 6890.8677520453875
    },
    "timestamp": {
      "timeInSeconds": 1580853000,
      "offsetInNanos": 0
    },
    "quality": "GOOD"
  }
}
```

For more information, see [Querying current asset property values](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [GetAssetPropertyValue](#) in *AWS CLI Command Reference*.

list-access-policies

The following code example shows how to use `list-access-policies`.

AWS CLI

To list all access policies

The following `list-access-policies` example lists all access policies for a user who is a portal administrator.

```
aws iotsitewise list-access-policies \  
  --identity-type USER \  
  --identity-id a1b2c3d4e5-a1b2c3d4-5678-90ab-cdef-bbbbbEXAMPLE
```

Output:

```
{  
  "accessPolicySummaries": [  
    {  
      "id": "a1b2c3d4-5678-90ab-cdef-ccccEXAMPLE",  
      "identity": {  
        "user": {  
          "id": "a1b2c3d4e5-a1b2c3d4-5678-90ab-cdef-bbbbbEXAMPLE"  
        }  
      },  
      "resource": {  
        "portal": {  
          "id": "a1b2c3d4-5678-90ab-cdef-aaaaEXAMPLE"  
        }  
      },  
      "permission": "ADMINISTRATOR"  
    }  
  ]  
}
```

For more information, see [Administering your portals](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [ListAccessPolicies](#) in *AWS CLI Command Reference*.

list-asset-models

The following code example shows how to use `list-asset-models`.

AWS CLI

To list all asset models

The following `list-asset-models` example lists all asset models that are defined in your AWS account in the current Region.

```
aws iotsitewise list-asset-models
```

Output:

```
{
  "assetModelSummaries": [
    {
      "id": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
      "arn": "arn:aws:iotsitewise:us-west-2:123456789012:asset-model/a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
      "name": "Wind Farm Model",
      "description": "Represents a wind farm that comprises many wind turbines",
      "creationDate": 1575671284.0,
      "lastUpdateDate": 1575671988.0,
      "status": {
        "state": "ACTIVE"
      }
    },
    {
      "id": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
      "arn": "arn:aws:iotsitewise:us-west-2:123456789012:asset-model/a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
      "name": "Wind Turbine Model",
      "description": "Represents a wind turbine manufactured by Example Corp",
      "creationDate": 1575671207.0,
      "lastUpdateDate": 1575686273.0,
      "status": {
        "state": "ACTIVE"
      }
    }
  ]
}
```

For more information, see [Listing all asset models](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [ListAssetModels](#) in *AWS CLI Command Reference*.

list-assets

The following code example shows how to use `list-assets`.

AWS CLI

Example 1: To list all top-level assets

The following `list-assets` example lists all assets that are top-level in the asset hierarchy tree and defined in your AWS account in the current Region.

```
aws iotsitewise list-assets \  
  --filter TOP_LEVEL
```

Output:

```
{  
  "assetSummaries": [  
    {  
      "id": "a1b2c3d4-5678-90ab-cdef-44444EXAMPLE",  
      "arn": "arn:aws:iotsitewise:us-west-2:123456789012:asset/  
a1b2c3d4-5678-90ab-cdef-44444EXAMPLE",  
      "name": "Wind Farm 1",  
      "assetModelId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",  
      "creationDate": 1575672453.0,  
      "lastUpdateDate": 1575672453.0,  
      "status": {  
        "state": "ACTIVE"  
      },  
      "hierarchies": [  
        {  
          "id": "a1b2c3d4-5678-90ab-cdef-77777EXAMPLE",  
          "name": "Wind Turbines"  
        }  
      ]  
    }  
  ]  
}
```

For more information, see [Listing assets](#) in the *AWS IoT SiteWise User Guide*.

Example 2: To list all assets based on an asset model

The following `list-assets` example lists all assets based on an asset model and defined in your AWS account in the current Region.

```
aws iotsitewise list-assets \  
  --asset-model-id a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
```

Output:

```
{  
  "assetSummaries": [  
    {  
      "id": "a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",  
      "arn": "arn:aws:iotsitewise:us-west-2:123456789012:asset/  
a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",  
      "name": "Wind Turbine 1",  
      "assetModelId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
      "creationDate": 1575671550.0,  
      "lastUpdateDate": 1575686308.0,  
      "status": {  
        "state": "ACTIVE"  
      },  
      "hierarchies": []  
    }  
  ]  
}
```

For more information, see [Listing assets](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [ListAssets](#) in *AWS CLI Command Reference*.

list-associated-assets

The following code example shows how to use `list-associated-assets`.

AWS CLI

To list all assets associated to an asset in a specific hierarchy

The following `list-associated-assets` example lists all wind turbine assets associated to the specified wind farm asset.

```
aws iotsitewise list-associated-assets \  
  --asset-id a1b2c3d4-5678-90ab-cdef-44444EXAMPLE \  
  --hierarchy-id a1b2c3d4-5678-90ab-cdef-77777EXAMPLE
```

Output:

```
{  
  "assetSummaries": [  
    {  
      "id": "a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",  
      "arn": "arn:aws:iotsitewise:us-west-2:123456789012:asset/  
a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",  
      "name": "Wind Turbine 1",  
      "assetModelId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
      "creationDate": 1575671550.0,  
      "lastUpdateDate": 1575686308.0,  
      "status": {  
        "state": "ACTIVE"  
      },  
      "hierarchies": []  
    }  
  ]  
}
```

For more information, see [Listing assets associated to a specific asset](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [ListAssociatedAssets](#) in *AWS CLI Command Reference*.

list-dashboards

The following code example shows how to use `list-dashboards`.

AWS CLI

To list all dashboards in a project

The following `list-dashboards` example lists all dashboards that are defined in a project.

```
aws iotsitewise list-dashboards \  
  --project-id a1b2c3d4-5678-90ab-cdef-eeeeeeEXAMPLE
```

Output:

```
{
  "dashboardSummaries": [
    {
      "id": "a1b2c3d4-5678-90ab-cdef-fffffEXAMPLE",
      "name": "Wind Farm",
      "creationDate": "2020-05-01T20:32:12.228476348Z",
      "lastUpdateDate": "2020-05-01T20:32:12.228476348Z"
    }
  ]
}
```

For more information, see [Viewing dashboards](#) in the *AWS IoT SiteWise Monitor Application Guide*.

- For API details, see [ListDashboards](#) in *AWS CLI Command Reference*.

list-gateways

The following code example shows how to use `list-gateways`.

AWS CLI**To list all gateways**

The following `list-gateways` example lists all gateways that are defined in your AWS account in the current Region.

```
aws iotsitewise list-gateways
```

Output:

```
{
  "gatewaySummaries": [
    {
      "gatewayId": "a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE",
      "gatewayName": "ExampleCorpGateway",
      "gatewayCapabilitySummaries": [
        {
          "capabilityNamespace": "iotsitewise:opcuacollector:1",
          "capabilitySyncStatus": "IN_SYNC"
        }
      ]
    }
  ]
}
```

```
    }
  ],
  "creationDate": 1588369971.457,
  "lastUpdateDate": 1588369971.457
}
]
```

For more information, see [Ingesting data using a gateway](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [ListGateways](#) in *AWS CLI Command Reference*.

list-portals

The following code example shows how to use `list-portals`.

AWS CLI

To list all portals

The following `list-portals` example lists all portals that are defined in your AWS account in the current Region.

```
aws iotsitewise list-portals
```

Output:

```
{
  "portalSummaries": [
    {
      "id": "a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE",
      "name": "WindFarmPortal",
      "description": "A portal that contains wind farm projects for Example Corp.",
      "startUrl": "https://a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE.app.iotsitewise.aws",
      "creationDate": "2020-02-04T23:01:52.90248068Z",
      "lastUpdateDate": "2020-02-04T23:01:52.90248078Z",
      "roleArn": "arn:aws:iam::123456789012:role/service-role/MySiteWiseMonitorServiceRole"
    }
  ]
}
```

```
}
```

For more information, see [Administering your portals](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [ListPortals](#) in *AWS CLI Command Reference*.

list-project-assets

The following code example shows how to use `list-project-assets`.

AWS CLI

To list all assets associated to a project

The following `list-project-assets` example lists all assets that are associated to a wind farm project.

```
aws iotsitewise list-projects \  
  --project-id a1b2c3d4-5678-90ab-cdef-eeeeEXAMPLE
```

Output:

```
{  
  "assetIds": [  
    "a1b2c3d4-5678-90ab-cdef-44444EXAMPLE"  
  ]  
}
```

For more information, see [Adding assets to projects](#) in the *AWS IoT SiteWise Monitor Application Guide*.

- For API details, see [ListProjectAssets](#) in *AWS CLI Command Reference*.

list-projects

The following code example shows how to use `list-projects`.

AWS CLI

To list all projects in a portal

The following `list-projects` example lists all projects that are defined in a portal.

```
aws iotsitewise list-projects \  
  --portal-id a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE
```

Output:

```
{  
  "projectSummaries": [  
    {  
      "id": "a1b2c3d4-5678-90ab-cdef-eeeeeeEXAMPLE",  
      "name": "Wind Farm 1",  
      "description": "Contains asset visualizations for Wind Farm #1 for  
Example Corp.",  
      "creationDate": "2020-02-20T21:58:43.362246001Z",  
      "lastUpdateDate": "2020-02-20T21:58:43.362246095Z"  
    }  
  ]  
}
```

For more information, see [Viewing project details](#) in the *AWS IoT SiteWise Monitor Application Guide*.

- For API details, see [ListProjects](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list all tags for a resource

The following `list-tags-for-resource` example lists all tags for a wind turbine asset.

```
aws iotsitewise list-tags-for-resource \  
  --resource-arn arn:aws:iotsitewise:us-west-2:123456789012:asset/  
a1b2c3d4-5678-90ab-cdef-33333EXAMPLE
```

Output:

```
{  
  "tags": {
```



```
    "Owner": "richard-roe"  
  }  
}
```

For more information, see [Tagging your resources](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

put-logging-options

The following code example shows how to use `put-logging-options`.

AWS CLI

To specify the level of logging

The following `put-logging-options` example enables INFO level logging in AWS IoT SiteWise. Other levels include DEBUG and OFF.

```
aws iotsitewise put-logging-options \  
  --logging-options level=INFO
```

This command produces no output.

For more information, see [Monitoring AWS IoT SiteWise with Amazon CloudWatch Logs](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [PutLoggingOptions](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add a tag to a resource

The following `tag-resource` example adds an owner tag to a wind turbine asset. This lets you control access to the asset based on who owns it.

```
aws iotsitewise tag-resource \  
  --tag-key Owner
```

```
--resource-arn arn:aws:iotsitewise:us-west-2:123456789012:asset/
a1b2c3d4-5678-90ab-cdef-33333EXAMPLE \
--tags Owner=richard-roe
```

This command produces no output.

For more information, see [Tagging your resources](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag from a resource

The following `untag-resource` example removes an owner tag from a wind turbine asset.

```
aws iotsitewise untag-resource \
  --resource-arn arn:aws:iotsitewise:us-west-2:123456789012:asset/
a1b2c3d4-5678-90ab-cdef-33333EXAMPLE \
  --tag-keys Owner
```

This command produces no output.

For more information, see [Tagging your resources](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-access-policy

The following code example shows how to use `update-access-policy`.

AWS CLI

To grant a project viewer ownership of a project

The following `update-access-policy` example updates an access policy that grants a project viewer ownership of a project.

```
aws iotsitewise update-access-policy \  
  --access-policy-id a1b2c3d4-5678-90ab-cdef-dddddEXAMPLE \  
  --cli-input-json file://update-project-viewer-access-policy.json
```

Contents of `update-project-viewer-access-policy.json`:

```
{  
  "accessPolicyIdentity": {  
    "user": {  
      "id": "a1b2c3d4e5-a1b2c3d4-5678-90ab-cdef-bbbbbEXAMPLE"  
    }  
  },  
  "accessPolicyPermission": "ADMINISTRATOR",  
  "accessPolicyResource": {  
    "project": {  
      "id": "a1b2c3d4-5678-90ab-cdef-eeeeeeEXAMPLE"  
    }  
  }  
}
```

This command produces no output.

For more information, see [Assigning project owners](#) in the *AWS IoT SiteWise Monitor Application Guide*.

- For API details, see [UpdateAccessPolicy](#) in *AWS CLI Command Reference*.

update-asset-model

The following code example shows how to use `update-asset-model`.

AWS CLI

To update an asset model

The following `update-asset-model` example updates a wind farm asset model's description. This example includes the model's existing IDs and definitions, because `update-asset-model` overwrites the existing model with the new model.

```
aws iotsitewise update-asset-model \  
  --cli-input-json file://update-wind-farm-model.json
```

Contents of update-wind-farm-model.json:

```

{
  "assetModelName": "Wind Farm Model",
  "assetModelDescription": "Represents a wind farm that comprises many wind
turbines",
  "assetModelProperties": [
    {
      "id": "a1b2c3d4-5678-90ab-cdef-88888EXAMPLE",
      "name": "Region",
      "dataType": "STRING",
      "type": {
        "attribute": {}
      }
    },
    {
      "id": "a1b2c3d4-5678-90ab-cdef-99999EXAMPLE",
      "name": "Total Generated Power",
      "dataType": "DOUBLE",
      "unit": "kW",
      "type": {
        "metric": {
          "expression": "sum(power)",
          "variables": [
            {
              "name": "power",
              "value": {
                "hierarchyId": "a1b2c3d4-5678-90ab-
cdef-77777EXAMPLE",
                "propertyId": "a1b2c3d4-5678-90ab-cdef-66666EXAMPLE"
              }
            }
          ]
        },
        "window": {
          "tumbling": {
            "interval": "1h"
          }
        }
      }
    }
  ],
  "assetModelHierarchies": [
    {

```

```

        "id": "a1b2c3d4-5678-90ab-cdef-77777EXAMPLE",
        "name": "Wind Turbines",
        "childAssetModelId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"
    }
]
}

```

Output:

```

{
  "assetModelId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
  "assetModelArn": "arn:aws:iotsitewise:us-west-2:123456789012:asset-model/a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
  "assetModelStatus": {
    "state": "CREATING"
  }
}

```

For more information, see [Updating asset models](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [UpdateAssetModel](#) in *AWS CLI Command Reference*.

update-asset-property

The following code example shows how to use `update-asset-property`.

AWS CLI

Example 1: To update an asset property's alias

The following `update-asset-property` example updates a wind turbine asset's power property alias.

```

aws iotsitewise update-asset-property \
  --asset-id a1b2c3d4-5678-90ab-cdef-33333EXAMPLE \
  --property-id a1b2c3d4-5678-90ab-cdef-55555EXAMPLE \
  --property-alias "/examplecorp/windfarm/1/turbine/1/power" \
  --property-notification-state DISABLED

```

This command produces no output.

For more information, see [Mapping industrial data streams to asset properties](#) in the *AWS IoT SiteWise User Guide*.

Example 2: To enable asset property notifications

The following `update-asset-property` example enables asset property update notifications for a wind turbine asset's power property. Property value updates are published to the MQTT topic `$aws/sitewise/asset-models/<assetModelId>/assets/<assetId>/properties/<propertyId>`, where each ID is replaced by the property, asset, and model ID of the asset property.

```
aws iotsitewise update-asset-property \  
  --asset-id a1b2c3d4-5678-90ab-cdef-33333EXAMPLE \  
  --property-id a1b2c3d4-5678-90ab-cdef-66666EXAMPLE \  
  --property-notification-state ENABLED \  
  --property-alias "/examplecorp/windfarm/1/turbine/1/power"
```

This command produces no output.

For more information, see [Interacting with other services](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [UpdateAssetProperty](#) in *AWS CLI Command Reference*.

update-asset

The following code example shows how to use `update-asset`.

AWS CLI

To update an asset's name

The following `update-asset` example updates a wind turbine asset's name.

```
aws iotsitewise update-asset \  
  --asset-id a1b2c3d4-5678-90ab-cdef-33333EXAMPLE \  
  --asset-name "Wind Turbine 2"
```

Output:

```
{  
  "assetStatus": {
```

```
    "state": "UPDATING"
  }
}
```

For more information, see [Updating assets](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [UpdateAsset](#) in *AWS CLI Command Reference*.

update-dashboard

The following code example shows how to use `update-dashboard`.

AWS CLI

To update a dashboard

The following `update-dashboard` example changes the title of a dashboard's line chart that displays total generated power for a wind farm.

```
aws iotsitewise update-dashboard \
  --project-id a1b2c3d4-5678-90ab-cdef-fffffEXAMPLE \
  --dashboard-name "Wind Farm" \
  --dashboard-definition file://update-wind-farm-dashboard.json
```

Contents of `update-wind-farm-dashboard.json`:

```
{
  "widgets": [
    {
      "type": "monitor-line-chart",
      "title": "Total Generated Power",
      "x": 0,
      "y": 0,
      "height": 3,
      "width": 3,
      "metrics": [
        {
          "label": "Power",
          "type": "iotsitewise",
          "assetId": "a1b2c3d4-5678-90ab-cdef-44444EXAMPLE",
          "propertyId": "a1b2c3d4-5678-90ab-cdef-99999EXAMPLE"
        }
      ]
    }
  ]
}
```

```

    ]
  }
]
}

```

This command produces no output.

For more information, see [Creating dashboards \(CLI\)](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [UpdateDashboard](#) in *AWS CLI Command Reference*.

update-gateway-capability-configuration

The following code example shows how to use `update-gateway-capability-configuration`.

AWS CLI

To update a gateway capability

The following `update-gateway-capability-configuration` example configures an OPC-UA source with the following properties:

Trusts any certificate. Uses the Basic256 algorithm to secure messages. Uses the SignAndEncrypt mode to secure connections. Uses authentication credentials stored in an AWS Secrets Manager secret.

```

aws iotsitewise update-gateway-capability-configuration \
  --gateway-id a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE \
  --capability-namespace "iotsitewise:opcuacollector:1" \
  --capability-configuration file://opc-ua-capability-configuration.json

```

Contents of `opc-ua-capability-configuration.json`:

```

{
  "sources": [
    {
      "name": "Wind Farm #1",
      "endpoint": {
        "certificateTrust": {
          "type": "TrustAny"
        },
        "endpointUri": "opc.tcp://203.0.113.0:49320",

```



```

        "securityPolicy": "BASIC256",
        "messageSecurityMode": "SIGN_AND_ENCRYPT",
        "identityProvider": {
            "type": "Username",
            "usernameSecretArn": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:greenrass-windfarm1-auth-1ABCDE"
        },
        "nodeFilterRules": []
    },
    "measurementDataStreamPrefix": ""
}
]
}

```

Output:

```

{
  "capabilityNamespace": "iotsitewise:opcuacollector:1",
  "capabilitySyncStatus": "OUT_OF_SYNC"
}

```

For more information, see [Configuring data sources](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [UpdateGatewayCapabilityConfiguration](#) in *AWS CLI Command Reference*.

update-gateway

The following code example shows how to use `update-gateway`.

AWS CLI**To update a gateway's name**

The following `update-gateway` example updates a gateway's name.

```

aws iotsitewise update-gateway \
  --gateway-id a1b2c3d4-5678-90ab-cdef-1a1a1EXAMPLE \
  --gateway-name ExampleCorpGateway1

```

This command produces no output.

For more information, see [Ingesting data using a gateway](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [UpdateGateway](#) in *AWS CLI Command Reference*.

update-portal

The following code example shows how to use `update-portal`.

AWS CLI

To update a portal's details

The following `update-portal` example updates a web portal for a wind farm company.

```
aws iotsitewise update-portal \  
  --portal-id a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE \  
  --portal-name WindFarmPortal \  
  --portal-description "A portal that contains wind farm projects for Example  
Corp." \  
  --portal-contact-email support@example.com \  
  --role-arn arn:aws:iam::123456789012:role/MySiteWiseMonitorServiceRole
```

Output:

```
{  
  "portalStatus": {  
    "state": "UPDATING"  
  }  
}
```

For more information, see [Administering your portals](#) in the *AWS IoT SiteWise User Guide*.

- For API details, see [UpdatePortal](#) in *AWS CLI Command Reference*.

update-project

The following code example shows how to use `update-project`.

AWS CLI

To update a project's details

The following `update-project` example updates a wind farm project.

```
aws iotsitewise update-project \  
  --project-id a1b2c3d4-5678-90ab-cdef-eeeeEXAMPLE \  
  --project-name "Wind Farm 1" \  
  --project-description "Contains asset visualizations for Wind Farm #1 for  
Example Corp."
```

This command produces no output.

For more information, see [Changing project details](#) in the *AWS IoT SiteWise Monitor Application Guide*.

- For API details, see [UpdateProject](#) in *AWS CLI Command Reference*.

AWS IoT Things Graph examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS IoT Things Graph.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-entity-to-thing

The following code example shows how to use `associate-entity-to-thing`.

AWS CLI

To associate a thing with a device

The following `associate-entity-to-thing` example associates a thing with a device. The example uses a motion sensor device that is in the public namespace.

```
aws iotthingsgraph associate-entity-to-thing \  
  --thing-name "MotionSensorName" \  
  --entity-id "urn:tdm:aws/examples:Device:HCSR501MotionSensor"
```

This command produces no output.

For more information, see [Creating and Uploading Models](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [AssociateEntityToThing](#) in *AWS CLI Command Reference*.

create-flow-template

The following code example shows how to use `create-flow-template`.

AWS CLI

To create a flow

The following `create-flow-template` example creates a flow (workflow). The value of `MyFlowDefinition` is the GraphQL that models the flow.

```
aws iotthingsgraph create-flow-template \  
  --definition language=GRAPHQL,text="MyFlowDefinition"
```

Output:

```
{  
  "summary": {  
    "createdAt": 1559248067.545,  
    "id": "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow",  
    "revisionNumber": 1  
  }  
}
```

For more information, see [Working with Flows](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [CreateFlowTemplate](#) in *AWS CLI Command Reference*.

create-system-instance

The following code example shows how to use create-system-instance.

AWS CLI

To create a system instance

The following create-system-instance example creates a system instance. The value of MySystemInstanceDefinition is the GraphQL that models the system instance.

```
aws iotthingsgraph create-system-instance -\
  -definition language=GRAPHQL,text="MySystemInstanceDefinition" \
  --target CLOUD \
  --flow-actions-role-arn myRoleARN
```

Output:

```
{
  "summary": {
    "id": "urn:tdm:us-west-2/123456789012/default:Deployment:Room218",
    "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/default/Room218",
    "status": "NOT_DEPLOYED",
    "target": "CLOUD",
    "createdAt": 1559249315.208,
    "updatedAt": 1559249315.208
  }
}
```

For more information, see [Working with Systems and Flow Configurations](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [CreateSystemInstance](#) in *AWS CLI Command Reference*.

create-system-template

The following code example shows how to use create-system-template.

AWS CLI

To create a system

The following `create-system-template` example creates a system. The value of `MySystemDefinition` is the GraphQL that models the system.

```
aws iotthingsgraph create-system-template \  
  --definition language=GRAPHQL,text="MySystemDefinition"
```

Output:

```
{  
  "summary": {  
    "createdAt": 1559249776.254,  
    "id": "urn:tdm:us-west-2/123456789012/default:System:MySystem",  
    "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:System/default/  
MySystem",  
    "revisionNumber": 1  
  }  
}
```

For more information, see [Creating Systems](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [CreateSystemTemplate](#) in *AWS CLI Command Reference*.

delete-flow-template

The following code example shows how to use `delete-flow-template`.

AWS CLI

To delete a flow

The following `delete-flow-template` example deletes a flow (workflow).

```
aws iotthingsgraph delete-flow-template \  
  --id "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow"
```

This command produces no output.

For more information, see [Lifecycle Management for AWS IoT Things Graph Entities, Flows, Systems, and Deployments](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [DeleteFlowTemplate](#) in *AWS CLI Command Reference*.

delete-namespace

The following code example shows how to use delete-namespace.

AWS CLI

To delete a namespace

The following delete-namespace example deletes a namespace.

```
aws iotthingsgraph delete-namespace
```

Output:

```
{
  "namespaceArn": "arn:aws:iotthingsgraph:us-west-2:123456789012",
  "namespaceName": "us-west-2/123456789012/default"
}
```

For more information, see [Lifecycle Management for AWS IoT Things Graph Entities, Flows, Systems, and Deployments](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [DeleteNamespace](#) in *AWS CLI Command Reference*.

delete-system-instance

The following code example shows how to use delete-system-instance.

AWS CLI

To delete a system instance

The following delete-system-instance example deletes a system instance.

```
aws iotthingsgraph delete-system-instance \
  --id "urn:tdm:us-west-2/123456789012/default:Deployment:Room218"
```

This command produces no output.

For more information, see [Lifecycle Management for AWS IoT Things Graph Entities, Flows, Systems, and Deployments](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [DeleteSystemInstance](#) in *AWS CLI Command Reference*.

delete-system-template

The following code example shows how to use `delete-system-template`.

AWS CLI

To delete a system

The following `delete-system-template` example deletes a system.

```
aws iotthingsgraph delete-system-template \  
  --id "urn:tdm:us-west-2/123456789012/default:System:MySystem"
```

This command produces no output.

For more information, see [Lifecycle Management for AWS IoT Things Graph Entities, Flows, Systems, and Deployments](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [DeleteSystemTemplate](#) in *AWS CLI Command Reference*.

deploy-system-instance

The following code example shows how to use `deploy-system-instance`.

AWS CLI

To deploy a system instance

The following `deploy-system-instance` example deploys a system instance.

```
aws iotthingsgraph deploy-system-instance \  
  --id "urn:tdm:us-west-2/123456789012/default:Deployment:Room218"
```

Output:

```
{  
  "summary": {  
    "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment:Room218",  
    "createdAt": 1559249776.254,  
    "id": "urn:tdm:us-west-2/123456789012/default:Deployment:Room218",
```



```
"status": "DEPLOYED_IN_TARGET",
"target": "CLOUD",
"updatedAt": 1559249776.254
}
}
```

For more information, see [Working with Systems and Flow Configurations](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [DeploySystemInstance](#) in *AWS CLI Command Reference*.

deprecate-flow-template

The following code example shows how to use `deprecate-flow-template`.

AWS CLI

To deprecate a flow

The following `deprecate-flow-template` example deprecates a flow (workflow).

```
aws iotthingsgraph deprecate-flow-template \
  --id "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow"
```

This command produces no output.

For more information, see [Lifecycle Management for AWS IoT Things Graph Entities, Flows, Systems, and Deployments](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [DeprecateFlowTemplate](#) in *AWS CLI Command Reference*.

deprecate-system-template

The following code example shows how to use `deprecate-system-template`.

AWS CLI

To deprecate a system

The following `deprecate-system-template` example deprecates a system.

```
aws iotthingsgraph deprecate-system-template \
```

```
--id "urn:tdm:us-west-2/123456789012/default:System:MySystem"
```

This command produces no output.

For more information, see [Lifecycle Management for AWS IoT Things Graph Entities, Flows, Systems, and Deployments](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [DeprecateSystemTemplate](#) in *AWS CLI Command Reference*.

describe-namespace

The following code example shows how to use describe-namespace.

AWS CLI

To get a description of your namespace

The following describe-namespace example gets a description of your namespace.

```
aws iotthingsgraph describe-namespace
```

Output:

```
{
  "namespaceName": "us-west-2/123456789012/default",
  "trackingNamespaceName": "aws",
  "trackingNamespaceVersion": 1,
  "namespaceVersion": 5
}
```

For more information, see [Namespaces](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [DescribeNamespace](#) in *AWS CLI Command Reference*.

dissociate-entity-from-thing

The following code example shows how to use dissociate-entity-from-thing.

AWS CLI

To dissociate a thing from a device

The following `dissociate-entity-from-thing` example dissociates a thing from a device.

```
aws iotthingsgraph dissociate-entity-from-thing \  
  --thing-name "MotionSensorName" \  
  --entity-type "DEVICE"
```

This command produces no output.

For more information, see [Creating and Uploading Models](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [DissociateEntityFromThing](#) in *AWS CLI Command Reference*.

get-entities

The following code example shows how to use `get-entities`.

AWS CLI

To get definitions for entities

The following `get-entities` example gets a definition for a device model.

```
aws iotthingsgraph get-entities \  
  --ids "urn:tdm:aws/examples:DeviceModel:MotionSensor"
```

Output:

```
{  
  "descriptions": [  
    {  
      "id": "urn:tdm:aws/examples:DeviceModel:MotionSensor",  
      "type": "DEVICE_MODEL",  
      "createdAt": 1559256190.599,  
      "definition": {  
        "language": "GRAPHQL",  
        "text": "##\n# Specification of motion sensor devices interface.\n##  
\n#type MotionSensor @deviceModel(id: \"urn:tdm:aws/examples:deviceModel:MotionSensor  
\",\n#  capability: \"urn:tdm:aws/examples:capability:MotionSensorCapability\")  
{ignore:void}"  
      }  
    }  
  ]  
}
```

```
]
}
```

For more information, see [Creating and Uploading Models](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [GetEntities](#) in *AWS CLI Command Reference*.

get-flow-template-revisions

The following code example shows how to use `get-flow-template-revisions`.

AWS CLI

To get revision information about a flow

The following `get-flow-template-revisions` example gets revision information about a flow (workflow).

```
aws iotthingsgraph get-flow-template-revisions \
  --id urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow
```

Output:

```
{
  "summaries": [
    {
      "id": "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow",
      "revisionNumber": 1,
      "createdAt": 1559247540.292
    }
  ]
}
```

For more information, see [Working with Flows](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [GetFlowTemplateRevisions](#) in *AWS CLI Command Reference*.

get-flow-template

The following code example shows how to use `get-flow-template`.

AWS CLI

To get a flow definition

The following `get-flow-template` example gets a definition for a flow (workflow).

```
aws iotthingsgraph get-flow-template \
  --id "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow"
```

Output:

```
{
  "description": {
    "summary": {
      "id": "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow",
      "revisionNumber": 1,
      "createdAt": 1559247540.292
    },
    "definition": {
      "language": "GRAPHQL",
      "text": "{\nquery MyFlow($camera: string!, $screen: string!)
@workflowType(id: \"urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow\")
@annotation(type: \"tgc:FlowEvent\", id: \"sledged790c1b2bcd949e09da0c9bfc077f79d
\", x: 1586, y: 653) @triggers(definition: \"{MotionSensor(description:
\\\\\"\\\\\") @position(x: 1045, y: 635.6666564941406) {\n  condition(expr:
\\\\\"devices[name == \\\\\"\\\\\"motionSensor\\\\\"\\\\\"].events[name == \\\\
\\\\\"StateChanged\\\\\"\\\\\"].lastEvent\\\\\")\n  action(expr: \\\\\"\\\\\")\n
\n}}\") {\n  variables {\n    cameraResult @property(id: \"urn:tdm:aws/
examples:property:CameraStateProperty\")\n  }\n  steps {\n    step(name: \"Camera
\", outEvent: [\"sledged790c1b2bcd949e09da0c9bfc077f79d\"] @position(x: 1377,
y: 638.6666564941406) {\n      DeviceActivity(deviceModel: \"urn:tdm:aws/
examples:deviceModel:Camera\", out: \"cameraResult\", deviceId: \"${camera}\")
{\n        capture\n      }\n    }\n    step(name: \"Screen\", inEvent:
[\"sledged790c1b2bcd949e09da0c9bfc077f79d\"] @position(x: 1675.6666870117188,
y: 637.9999847412109) {\n      DeviceActivity(deviceModel: \"urn:tdm:aws/
examples:deviceModel:Screen\", deviceId: \"${screen}\") {\n        display(imageUrl:
\"${cameraResult.lastClickedImage}\")\n      }\n    }\n  }\n}\n}\n}\n}"}
    },
    "validatedNamespaceVersion": 5
  }
}
```

For more information, see [Working with Flows](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [GetFlowTemplate](#) in *AWS CLI Command Reference*.

get-namespace-deletion-status

The following code example shows how to use `get-namespace-deletion-status`.

AWS CLI

To get the status of the namespace deletion task

The following `get-namespace-deletion-status` example gets the status of the namespace deletion task.

```
aws iotthingsgraph get-namespace-deletion-status
```

Output:

```
{
  "namespaceArn": "arn:aws:iotthingsgraph:us-west-2:123456789012",
  "namespaceName": "us-west-2/123456789012/default"
  "status": "SUCCEEDED "
}
```

For more information, see [Namespaces](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [GetNamespaceDeletionStatus](#) in *AWS CLI Command Reference*.

get-system-instance

The following code example shows how to use `get-system-instance`.

AWS CLI

To get a system instance

The following `get-system-instance` example gets a definition for a system instance.

```
aws iotthingsgraph get-system-instance \
  --id "urn:tdm:us-west-2/123456789012/default:Deployment:Room218"
```

Output:

```

{
  "description": {
    "summary": {
      "id": "urn:tdm:us-west-2/123456789012/default:Deployment:Room218",
      "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/
default/Room218",
      "status": "NOT_DEPLOYED",
      "target": "CLOUD",
      "createdAt": 1559249315.208,
      "updatedAt": 1559249315.208
    },
    "definition": {
      "language": "GRAPHQL",
      "text": "{\r\nquery Room218 @deployment(id: \"urn:tdm:us-
west-2/123456789012/default:Deployment:Room218\", systemId: \"urn:tdm:us-
west-2/123456789012/default:System:SecurityFlow\") {\r\n  motionSensor(deviceId:
\"MotionSensorName\")\r\n  screen(deviceId: \"ScreenName\")\r\n  camera(deviceId:
\"CameraName\") \r\n  triggers {MotionEventTrigger(description:
\"a trigger\") { \r\n    condition(expr: \"devices[name ==
'motionSensor'].events[name == 'StateChanged'].lastEvent\") \r\n    action(expr:
\"ThingsGraph.startFlow('SecurityFlow', bindings[name == 'camera'].deviceId,
bindings[name == 'screen'].deviceId))\r\n  }\r\n  }\r\n  }\r\n  }\r\n  }"
    },
    "metricsConfiguration": {
      "cloudMetricEnabled": false
    },
    "validatedNamespaceVersion": 5,
    "flowActionsRoleArn": "arn:aws:iam::123456789012:role/ThingsGraphRole"
  }
}

```

For more information, see [Working with Systems and Flow Configurations](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [GetSystemInstance](#) in *AWS CLI Command Reference*.

get-system-template-revisions

The following code example shows how to use `get-system-template-revisions`.

AWS CLI

To get revision information about a system

The following `get-system-template-revisions` example gets revision information about a system.

```
aws iotthingsgraph get-system-template-revisions \  
  --id "urn:tdm:us-west-2/123456789012/default:System:MySystem"
```

Output:

```
{  
  "summaries": [  
    {  
      "id": "urn:tdm:us-west-2/123456789012/default:System:MySystem",  
      "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:System/default/  
MySystem",  
      "revisionNumber": 1,  
      "createdAt": 1559247540.656  
    }  
  ]  
}
```

For more information, see [Working with Systems and Flow Configurations](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [GetSystemTemplateRevisions](#) in *AWS CLI Command Reference*.

get-system-template

The following code example shows how to use `get-system-template`.

AWS CLI

To get a system

The following `get-system-template` example gets a definition for a system.

```
aws iotthingsgraph get-system-template \  
  --id "urn:tdm:us-west-2/123456789012/default:System:MySystem"
```

Output:

```
{  
  "description": {
```



```

    "summary": {
      "id": "urn:tdm:us-west-2/123456789012/default:System:MySystem",
      "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:System/default/MyFlow",
      "revisionNumber": 1,
      "createdAt": 1559247540.656
    },
    "definition": {
      "language": "GRAPHQL",
      "text": "{\n  type MySystem @systemType(id: \"urn:tdm:us-west-2/123456789012/default:System:MySystem\", description: \"\") {\n    camera: Camera @thing(id: \"urn:tdm:aws/examples:deviceModel:Camera\")\n    screen: Screen @thing(id: \"urn:tdm:aws/examples:deviceModel:Screen\")\n    motionSensor: MotionSensor @thing(id: \"urn:tdm:aws/examples:deviceModel:MotionSensor\")\n    MyFlow: MyFlow @workflow(id: \"urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow\")\n  }\n}"
    },
    "validatedNamespaceVersion": 5
  }
}

```

For more information, see [Working with Systems and Flow Configurations](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [GetSystemTemplate](#) in *AWS CLI Command Reference*.

get-upload-status

The following code example shows how to use `get-upload-status`.

AWS CLI

To get the status of your entity upload

The following `get-upload-status` example gets the status of your entity upload operation. The value of `MyUploadId` is the ID value returned by the `upload-entity-definitions` operation.

```
aws iotthingsgraph get-upload-status \
  --upload-id "MyUploadId"
```

Output:

```
{
  "namespaceName": "us-west-2/123456789012/default",
  "namespaceVersion": 5,
  "uploadId": "f6294f1e-b109-4bbe-9073-f451a2dda2da",
  "uploadStatus": "SUCCEEDED"
}
```

For more information, see [Modeling Entities](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [GetUploadStatus](#) in *AWS CLI Command Reference*.

list-flow-execution-messages

The following code example shows how to use `list-flow-execution-messages`.

AWS CLI

To get information about events in a flow execution

The following `list-flow-execution-messages` example gets information about events in a flow execution.

```
aws iotthingsgraph list-flow-execution-messages \
  --flow-execution-id "urn:tdm:us-west-2/123456789012/
default:Workflow:SecurityFlow_2019-05-11T19:39:55.317Z_MotionSensor_69b151ad-
a611-42f5-ac21-fe537f9868ad"
```

Output:

```
{
  "messages": [
    {
      "eventType": "EXECUTION_STARTED",
      "messageId": "f6294f1e-b109-4bbe-9073-f451a2dda2da",
      "payload": "Flow execution started",
      "timestamp": 1559247540.656
    }
  ]
}
```

For more information, see [Working with Flows](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [ListFlowExecutionMessages](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list all tags for a resource

The following `list-tags-for-resource` example list all tags for an AWS IoT Things Graph resource.

```
aws iotthingsgraph list-tags-for-resource \
  --resource-arn "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/
  default/Room218"
```

Output:

```
{
  "tags": [
    {
      "key": "Type",
      "value": "Residential"
    }
  ]
}
```

For more information, see [Tagging Your AWS IoT Things Graph Resources](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

search-entities

The following code example shows how to use `search-entities`.

AWS CLI

To search for entities

The following `search-entities` example searches for all entities of type EVENT.

```
aws iotthingsgraph search-entities \
  --entity-types "EVENT"
```

Output:

```
{
  "descriptions": [
    {
      "id": "urn:tdm:aws/examples:Event:MotionSensorEvent",
      "type": "EVENT",
      "definition": {
        "language": "GRAPHQL",
        "text": "##\n# Description of events emitted by motion
sensor.\n##\n# type MotionSensorEvent @eventType(id: \"urn:tdm:aws/
examples:event:MotionSensorEvent\", \n          payload: \"urn:tdm:aws/
examples:property:MotionSensorStateProperty\") {ignore:void}"
      }
    },
    {
      "id": "urn:tdm:us-west-2/123456789012/
default:Event:CameraClickedEventV2",
      "type": "EVENT",
      "definition": {
        "language": "GRAPHQL",
        "text": "type CameraClickedEventV2 @eventType(id: \"urn:tdm:us-
west-2/123456789012/default:event:CameraClickedEventV2\", \r\npayload:
\"urn:tdm:aws:Property:Boolean\") {ignore:void}"
      }
    },
    {
      "id": "urn:tdm:us-west-2/123456789012/
default:Event:MotionSensorEventV2",
      "type": "EVENT",
      "definition": {
        "language": "GRAPHQL",
        "text": "# Event emitted by the motion sensor.\r\n# type
MotionSensorEventV2 @eventType(id: \"urn:tdm:us-west-2/123456789012/
default:event:MotionSensorEventV2\", \r\npayload: \"urn:tdm:us-west-2/123456789012/
default:property:MotionSensorStateProperty2\") {ignore:void}"
      }
    }
  ],
  "nextToken": "urn:tdm:us-west-2/123456789012/default:Event:MotionSensorEventV2"
```

```
}
```

For more information, see [AWS IoT Things Graph Data Model Reference](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [SearchEntities](#) in *AWS CLI Command Reference*.

search-flow-executions

The following code example shows how to use search-flow-executions.

AWS CLI

To search for flow executions

The following search-flow-executions example search for all executions of a flow in a specified system instance.

```
aws iotthingsgraph search-flow-executions \  
  --system-instance-id "urn:tdm:us-west-2/123456789012/default:Deployment:Room218"
```

Output:

```
{  
  "summaries": [  
    {  
      "createdAt": 1559247540.656,  
      "flowExecutionId": "f6294f1e-b109-4bbe-9073-f451a2dda2da",  
      "flowTemplateId": "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow",  
      "status": "RUNNING ",  
      "systemInstanceId": "urn:tdm:us-west-2/123456789012/  
default:System:MySystem",  
      "updatedAt": 1559247540.656  
    }  
  ]  
}
```

For more information, see [Working with Systems and Flow Configurations](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [SearchFlowExecutions](#) in *AWS CLI Command Reference*.

search-flow-templates

The following code example shows how to use search-flow-templates.

AWS CLI

To search for flows (or workflows)

The following search-flow-templates example searches for all flows (workflows) that contain the Camera device model.

```
aws iotthingsgraph search-flow-templates \  
  --filters name="DEVICE_MODEL_ID",value="urn:tdm:aws/examples:DeviceModel:Camera"
```

Output:

```
{  
  "summaries": [  
    {  
      "id": "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow",  
      "revisionNumber": 1,  
      "createdAt": 1559247540.292  
    },  
    {  
      "id": "urn:tdm:us-west-2/123456789012/default:Workflow:SecurityFlow",  
      "revisionNumber": 3,  
      "createdAt": 1548283099.27  
    }  
  ]  
}
```

For more information, see [Working with Flows](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [SearchFlowTemplates](#) in *AWS CLI Command Reference*.

search-system-instances

The following code example shows how to use search-system-instances.

AWS CLI

To search for system instances

The following `search-system-instances` example searches for all system instances that contain the specified system.

```
aws iotthingsgraph search-system-instances \  
  --filters name="SYSTEM_TEMPLATE_ID",value="urn:tdm:us-west-2/123456789012/  
default:System:SecurityFlow"
```

Output:

```
{  
  "summaries": [  
    {  
      "id": "urn:tdm:us-west-2/123456789012/  
default:Deployment:DeploymentForSample",  
      "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/  
default/DeploymentForSample",  
      "status": "NOT_DEPLOYED",  
      "target": "GREENGRASS",  
      "greengrassGroupName": "ThingsGraphGrnGr",  
      "createdAt": 1555716314.707,  
      "updatedAt": 1555716314.707  
    },  
    {  
      "id": "urn:tdm:us-west-2/123456789012/  
default:Deployment:MockDeployment",  
      "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/  
default/MockDeployment",  
      "status": "DELETED_IN_TARGET",  
      "target": "GREENGRASS",  
      "greengrassGroupName": "ThingsGraphGrnGr",  
      "createdAt": 1549416462.049,  
      "updatedAt": 1549416722.361,  
      "greengrassGroupId": "01d04b07-2a51-467f-9d03-0c90b3cdcaaf",  
      "greengrassGroupVersionId": "7365aed7-2d3e-4d13-aad8-75443d45eb05"  
    },  
    {  
      "id": "urn:tdm:us-west-2/123456789012/  
default:Deployment:MockDeployment2",  
      "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/  
default/MockDeployment2",  
      "status": "DEPLOYED_IN_TARGET",  
      "target": "GREENGRASS",  
      "greengrassGroupName": "ThingsGraphGrnGr",
```

```

        "createdAt": 1549572385.774,
        "updatedAt": 1549572418.408,
        "greengrassGroupId": "01d04b07-2a51-467f-9d03-0c90b3cdcaaf",
        "greengrassGroupVersionId": "bfa70ab3-2bf7-409c-a4d4-bc8328ae5b86"
    },
    {
        "id": "urn:tdm:us-west-2/123456789012/default:Deployment:Room215",
        "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/
default/Room215",
        "status": "NOT_DEPLOYED",
        "target": "GREENGRASS",
        "greengrassGroupName": "ThingsGraphGG",
        "createdAt": 1547056918.413,
        "updatedAt": 1547056918.413
    },
    {
        "id": "urn:tdm:us-west-2/123456789012/default:Deployment:Room218",
        "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/
default/Room218",
        "status": "NOT_DEPLOYED",
        "target": "CLOUD",
        "createdAt": 1559249315.208,
        "updatedAt": 1559249315.208
    }
]
}

```

For more information, see [Working with Systems and Flow Configurations](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [SearchSystemInstances](#) in *AWS CLI Command Reference*.

search-system-templates

The following code example shows how to use search-system-templates.

AWS CLI

To search for system

The following search-system-templates example searches for all systems that contain the specified flow.


```
aws iotthingsgraph search-system-templates \  
  --filters name="FLOW_TEMPLATE_ID",value="urn:tdm:us-west-2/123456789012/  
  default:Workflow:SecurityFlow"
```

Output:

```
{  
  "summaries": [  
    {  
      "id": "urn:tdm:us-west-2/123456789012/default:System:SecurityFlow",  
      "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:System/default/  
SecurityFlow",  
      "revisionNumber": 1,  
      "createdAt": 1548283099.433  
    }  
  ]  
}
```

For more information, see [Working with Flows](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [SearchSystemTemplates](#) in *AWS CLI Command Reference*.

search-things

The following code example shows how to use search-things.

AWS CLI

To search for things associated with devices and device models

The following search-things example searches for all things that are associated with the HCSR501MotionSensor device.

```
aws iotthingsgraph search-things \  
  --entity-id "urn:tdm:aws/examples:Device:HCSR501MotionSensor"
```

Output:

```
{  
  "things": [  

```

```
{
  "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/MotionSensor1",
  "thingName": "MotionSensor1"
},
{
  "thingArn": "arn:aws:iot:us-west-2:123456789012:thing/TG_MS",
  "thingName": "TG_MS"
}
]
```

For more information, see [Creating and Uploading Models](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [SearchThings](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To create a tag for a resource

The following tag-resource example creates a tag for the specified resource.

```
aws iotthingsgraph tag-resource \
  --resource-arn "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/
default/Room218" \
  --tags key="Type",value="Residential"
```

This command produces no output.

For more information, see [Tagging Your AWS IoT Things Graph Resources](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

undeploy-system-instance

The following code example shows how to use undeploy-system-instance.

AWS CLI

To undeploy a system instance from its target

The following `undeploy-system-instance` example removes a system instance from its target.

```
aws iotthingsgraph undeploy-system-instance \  
  --id "urn:tdm:us-west-2/123456789012/default:Deployment:Room215"
```

Output:

```
{  
  "summary": {  
    "id": "urn:tdm:us-west-2/123456789012/default:Deployment:Room215",  
    "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/default/  
Room215",  
    "status": "PENDING_DELETE",  
    "target": "GREENGRASS",  
    "greengrassGroupName": "ThingsGraphGrnGr",  
    "createdAt": 1553189694.255,  
    "updatedAt": 1559344549.601,  
    "greengrassGroupId": "01d04b07-2a51-467f-9d03-0c90b3cdcaaf",  
    "greengrassGroupVersionId": "731b371d-d644-4b67-ac64-3934e99b75d7"  
  }  
}
```

For more information, see [Lifecycle Management for AWS IoT Things Graph Entities, Flows, Systems, and Deployments](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [UndeploySystemInstance](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag for a resource

The following `untag-resource` example removes a tag for the specified resource.

```
aws iotthingsgraph untag-resource \  
  --resource-arn "arn:aws:iotthingsgraph:us-west-2:123456789012:Deployment/  
default/Room218" \  
  --tag-keys "Type"
```

This command produces no output.

For more information, see [Tagging Your AWS IoT Things Graph Resources](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-flow-template

The following code example shows how to use `update-flow-template`.

AWS CLI

To update a flow

The following `update-flow-template` example updates a flow (workflow). The value of `MyFlowDefinition` is the GraphQL that models the flow.

```
aws iotthingsgraph update-flow-template \  
  --id "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow" \  
  --definition language=GRAPHQL,text="MyFlowDefinition"
```

Output:

```
{  
  "summary": {  
    "createdAt": 1559248067.545,  
    "id": "urn:tdm:us-west-2/123456789012/default:Workflow:MyFlow",  
    "revisionNumber": 2  
  }  
}
```

For more information, see [Working with Flows](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [UpdateFlowTemplate](#) in *AWS CLI Command Reference*.

update-system-template

The following code example shows how to use `update-system-template`.

AWS CLI

To update a system

The following `update-system-template` example updates a system. The value of `MySystemDefinition` is the GraphQL that models the system.

```
aws iotthingsgraph update-system-template \  
  --id "urn:tdm:us-west-2/123456789012/default:System:MySystem" \  
  --definition language=GRAPHQL,text="MySystemDefinition"
```

Output:

```
{  
  "summary": {  
    "createdAt": 1559249776.254,  
    "id": "urn:tdm:us-west-2/123456789012/default:System:MySystem",  
    "arn": "arn:aws:iotthingsgraph:us-west-2:123456789012:System/default/  
MySystem",  
    "revisionNumber": 2  
  }  
}
```

For more information, see [Creating Systems](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [UpdateSystemTemplate](#) in *AWS CLI Command Reference*.

upload-entity-definitions

The following code example shows how to use `upload-entity-definitions`.

AWS CLI

To upload entity definitions

The following `upload-entity-definitions` example uploads entity definitions to your namespace. The value of `MyEntityDefinitions` is the GraphQL that models the entities.

```
aws iotthingsgraph upload-entity-definitions \  
  --namespace "urn:tdm:us-west-2/123456789012/default:Namespace:MyNamespace" \  
  --definitions "MyEntityDefinitions"
```

```
--document language=GRAPHQL,text="MyEntityDefinitions"
```

Output:

```
{  
  "uploadId": "f6294f1e-b109-4bbe-9073-f451a2dda2da"  
}
```

For more information, see [Modeling Entities](#) in the *AWS IoT Things Graph User Guide*.

- For API details, see [UploadEntityDefinitions](#) in *AWS CLI Command Reference*.

AWS IoT Wireless examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS IoT Wireless.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-aws-account-with-partner-account

The following code example shows how to use `associate-aws-account-with-partner-account`.

AWS CLI

To associate a partner account with your AWS account

The following `associate-aws-account-with-partner-account` example associates the following Sidewalk account credentials with your AWS account.

```
aws iotwireless associate-aws-account-with-partner-account \  
  --sidewalk  
  AmazonId="12345678901234",AppServerPrivateKey="a123b45c6d78e9f012a34cd5e6a7890b12c3d45e6f78a1b234c56d7e890a1234"
```

Output:

```
{  
  "Sidewalk": {  
    "AmazonId": "12345678901234",  
    "AppServerPrivateKey":  
    "a123b45c6d78e9f012a34cd5e6a7890b12c3d45e6f78a1b234c56d7e890a1234"  
  }  
}
```

For more information, see [Amazon Sidewalk Integration for AWS IoT Core](#) in the *AWS IoT Developers Guide*.

- For API details, see [AssociateAwsAccountWithPartnerAccount](#) in *AWS CLI Command Reference*.

associate-wireless-device-with-thing

The following code example shows how to use `associate-wireless-device-with-thing`.

AWS CLI

To associate a thing to a wireless device

The following `associate-wireless-device-with-thing` example associates a thing to your wireless device that has the specified ID.

```
aws iotwireless associate-wireless-device-with-thing \  
  --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d" \  
  --thing-arn "arn:aws:iot:us-east-1:123456789012:thing/MyIoTWirelessThing"
```

This command produces no output.

For more information, see [Add your gateways and wireless devices to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [AssociateWirelessDeviceWithThing](#) in *AWS CLI Command Reference*.

associate-wireless-gateway-with-certificate

The following code example shows how to use `associate-wireless-gateway-with-certificate`.

AWS CLI

To associate the certificate with the wireless gateway

The following `associate-wireless-gateway-with-certificate` associates a wireless gateway with a certificate.

```
aws iotwireless associate-wireless-gateway-with-certificate \
  --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d" \
  --iot-certificate-id
"a123b45c6d78e9f012a34cd5e6a7890b12c3d45e6f78a1b234c56d7e890a1234"
```

Output:

```
{
  "IotCertificateId":
  "a123b45c6d78e9f012a34cd5e6a7890b12c3d45e6f78a1b234c56d7e890a1234"
}
```

For more information, see [Add your gateways and wireless devices to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [AssociateWirelessGatewayWithCertificate](#) in *AWS CLI Command Reference*.

associate-wireless-gateway-with-thing

The following code example shows how to use `associate-wireless-gateway-with-thing`.

AWS CLI

To associate a thing to a wireless gateway

The following `associate-wireless-gateway-with-thing` example associates a thing to a wireless gateway.


```
aws iotwireless associate-wireless-gateway-with-thing \  
  --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d" \  
  --thing-arn "arn:aws:iot:us-east-1:123456789012:thing/MyIoTWirelessThing"
```

This command produces no output.

For more information, see [Add your gateways and wireless devices to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [AssociateWirelessGatewayWithThing](#) in *AWS CLI Command Reference*.

create-destination

The following code example shows how to use create-destination.

AWS CLI

To create an IoT wireless destination

The following create-destination example creates a destination for mapping a device message to an AWS IoT rule. Before you run this command, you must have created an IAM role that gives AWS IoT Core for LoRaWAN the permissions necessary to send data to the AWS IoT rule.

```
aws iotwireless create-destination \  
  --name IoTWirelessDestination \  
  --expression-type RuleName \  
  --expression IoTWirelessRule \  
  --role-arn arn:aws:iam::123456789012:role/IoTWirelessDestinationRole
```

Output:

```
{  
  "Arn": "arn:aws:iotwireless:us-east-1:123456789012:Destination/  
IoTWirelessDestination",  
  "Name": "IoTWirelessDestination"  
}
```

For more information, see [Add destinations to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [CreateDestination](#) in *AWS CLI Command Reference*.

create-device-profile

The following code example shows how to use `create-device-profile`.

AWS CLI

To create a new device profile

The following `create-device-profile` example creates a new IoT wireless device profile.

```
aws iotwireless create-device-profile
```

Output:

```
{
  "Arn": "arn:aws:iotwireless:us-east-1:123456789012:DeviceProfile/12345678-
a1b2-3c45-67d8-e90fa1b2c34d",
  "Id": "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
}
```

For more information, see [Add profiles to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [CreateDeviceProfile](#) in *AWS CLI Command Reference*.

create-service-profile

The following code example shows how to use `create-service-profile`.

AWS CLI

To create a new service profile

The following `create-service-profile` example creates a new IoT wireless service profile.

```
aws iotwireless create-service-profile
```

Output:

```
{
  "Arn": "arn:aws:iotwireless:us-east-1:123456789012:ServiceProfile/12345678-
a1b2-3c45-67d8-e90fa1b2c34d",
  "Id": "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
}
```

For more information, see [Add profiles to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [CreateServiceProfile](#) in *AWS CLI Command Reference*.

create-wireless-device

The following code example shows how to use `create-wireless-device`.

AWS CLI

To create an IoT wireless device

The following `create-wireless-device` example creates a wireless device resource of the type LoRaWAN.

```
aws iotwireless create-wireless-device \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "Description": "My LoRaWAN wireless device"
  "DestinationName": "IoTWirelessDestination"
  "LoRaWAN": {
    "DeviceProfileId": "ab0c23d3-b001-45ef-6a01-2bc3de4f5333",
    "ServiceProfileId": "fe98dc76-cd12-001e-2d34-5550432da100",
    "OtaaV1_1": {
      "AppKey": "3f4ca100e2fc675ea123f4eb12c4a012",
      "JoinEui": "b4c231a359bc2e3d",
      "NwkKey": "01c3f004a2d6efffe32c4eda14bcd2b4"
    },
    "DevEui": "ac12efc654d23fc2"
  },
  "Name": "SampleIoTWirelessThing"
```

```
"Type": LoRaWAN
}
```

Output:

```
{
  "Arn": "arn:aws:iotwireless:us-
east-1:123456789012:WirelessDevice/1ffd32c8-8130-4194-96df-622f072a315f",
  "Id": "1ffd32c8-8130-4194-96df-622f072a315f"
}
```

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [CreateWirelessDevice](#) in *AWS CLI Command Reference*.

create-wireless-gateway-task-definition

The following code example shows how to use `create-wireless-gateway-task-definition`.

AWS CLI

To create a wireless gateway task definition

The following `create-wireless-gateway-task-definition` automatically creates tasks using this task definition for all gateways with the specified current version.

```
aws iotwireless create-wireless-gateway-task-definition \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "AutoCreateTasks": true,
  "Name": "TestAutoUpdate",
  "Update": {
    "UpdateDataSource" : "s3://cupsalphagafirmwarebin/station",
    "UpdateDataRole" : "arn:aws:iam::001234567890:role/SDK_Test_Role",
    "LoRaWAN" : {
      "CurrentVersion" : {
        "PackageVersion" : "1.0.0",
```

```

        "Station" : "2.0.5",
        "Model" : "linux"
    },
    "UpdateVersion" :{
        "PackageVersion" : "1.0.1",
        "Station" : "2.0.5",
        "Model" : "minihub"
    }
}
}
}
}

```

Output:

```

{
  "Id": "b7d3baad-25c7-35e7-a4e1-1683a0d61da9"
}

```

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [CreateWirelessGatewayTaskDefinition](#) in *AWS CLI Command Reference*.

create-wireless-gateway-task

The following code example shows how to use `create-wireless-gateway-task`.

AWS CLI**To create the task for a wireless gateway**

The following `create-wireless-gateway-task` example creates a task for a wireless gateway.

```

aws iotwireless create-wireless-gateway-task \
  --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d" \
  --wireless-gateway-task-definition-id "aa000102-0304-b0cd-ef56-a1b23cde456a"

```

Output:

```

{

```

```
"WirelessGatewayTaskDefinitionId": "aa204003-0604-30fb-ac82-a4f95aaf450a",  
"Status": "Success"  
}
```

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [CreateWirelessGatewayTask](#) in *AWS CLI Command Reference*.

create-wireless-gateway

The following code example shows how to use `create-wireless-gateway`.

AWS CLI

To create a wireless gateway

The following `create-wireless-gateway` example creates a wireless LoRaWAN device gateway.

```
aws iotwireless create-wireless-gateway \  
  --lorawan GatewayEui="a1b2c3d4567890ab",RfRegion="US915" \  
  --name "myFirstLoRaWANGateway" \  
  --description "Using my first LoRaWAN gateway"
```

Output:

```
{  
  "Arn": "arn:aws:iotwireless:us-east-1:123456789012:WirelessGateway/12345678-  
a1b2-3c45-67d8-e90fa1b2c34d",  
  "Id": "12345678-a1b2-3c45-67d8-e90fa1b2c34d"  
}
```

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [CreateWirelessGateway](#) in *AWS CLI Command Reference*.

delete-destination

The following code example shows how to use `delete-destination`.

AWS CLI

To delete an IoT wireless destination

The following `delete-destination` example deletes the wireless destination resource with the name `IoTWirelessDestination` that you created.

```
aws iotwireless delete-destination \  
  --name "IoTWirelessDestination"
```

This command produces no output.

For more information, see [Add destinations to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [DeleteDestination](#) in *AWS CLI Command Reference*.

`delete-device-profile`

The following code example shows how to use `delete-device-profile`.

AWS CLI

To delete a device profile

The following `delete-device-profile` example deletes a device profile with the specified ID that you created.

```
aws iotwireless delete-device-profile \  
  --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

This command produces no output.

For more information, see [Add profiles to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [DeleteDeviceProfile](#) in *AWS CLI Command Reference*.

`delete-service-profile`

The following code example shows how to use `delete-service-profile`.

AWS CLI

To delete a service profile

The following `delete-service-profile` example deletes a service profile with the specified ID that you created.

```
aws iotwireless delete-service-profile \  
  --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

This command produces no output.

For more information, see [Add profiles to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [DeleteServiceProfile](#) in *AWS CLI Command Reference*.

delete-wireless-device

The following code example shows how to use `delete-wireless-device`.

AWS CLI

To delete a wireless device

The following `delete-wireless-device` example deletes a wireless device that has the specified ID.

```
aws iotwireless delete-wireless-device \  
  --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

This command produces no output.

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [DeleteWirelessDevice](#) in *AWS CLI Command Reference*.

delete-wireless-gateway-task-definition

The following code example shows how to use `delete-wireless-gateway-task-definition`.

AWS CLI

To delete a wireless gateway task definition

The following `delete-wireless-gateway-task-definition` example deletes the wireless gateway task definition that you created with the following ID.

```
aws iotwireless delete-wireless-gateway-task-definition \  
  --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

This command produces no output.

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [DeleteWirelessGatewayTaskDefinition](#) in *AWS CLI Command Reference*.

`delete-wireless-gateway-task`

The following code example shows how to use `delete-wireless-gateway-task`.

AWS CLI

To delete a wireless gateway task

The following `delete-wireless-gateway-task` example deletes the wireless gateway task that has the specified ID.

```
aws iotwireless delete-wireless-gateway-task \  
  --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

This command produces no output.

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [DeleteWirelessGatewayTask](#) in *AWS CLI Command Reference*.

`delete-wireless-gateway`

The following code example shows how to use `delete-wireless-gateway`.

AWS CLI

To delete a wireless gateway

The following `delete-wireless-gateway` example deletes a wireless gateway that has the specified ID.

```
aws iotwireless delete-wireless-gateway \  
  --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

This command produces no output.

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [DeleteWirelessGateway](#) in *AWS CLI Command Reference*.

disassociate-aws-account-from-partner-account

The following code example shows how to use `disassociate-aws-account-from-partner-account`.

AWS CLI

To disassociate the partner account from the AWS account

The following `disassociate-aws-account-from-partner-account` example disassociates a partner account from your currently associated AWS account.

```
aws iotwireless disassociate-aws-account-from-partner-account \  
  --partner-account-id "12345678901234" \  
  --partner-type "Sidewalk"
```

This command produces no output.

For more information, see [Add your gateways and wireless devices to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [DisassociateAwsAccountFromPartnerAccount](#) in *AWS CLI Command Reference*.

disassociate-wireless-device-from-thing

The following code example shows how to use `disassociate-wireless-device-from-thing`.

AWS CLI

To disassociate the thing from the wireless device

The following `disassociate-wireless-device-from-thing` example disassociates a wireless device from its currently associated thing.

```
aws iotwireless disassociate-wireless-device-from-thing \  
  --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

This command produces no output.

For more information, see [Add your gateways and wireless devices to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [DisassociateWirelessDeviceFromThing](#) in *AWS CLI Command Reference*.

disassociate-wireless-gateway-from-certificate

The following code example shows how to use `disassociate-wireless-gateway-from-certificate`.

AWS CLI

To disassociate the certificate from the wireless gateway

The following `disassociate-wireless-gateway-from-certificate` disassociates a wireless gateway from its currently associated certificate.

```
aws iotwireless disassociate-wireless-gateway-from-certificate \  
  --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

This command produces no output.

For more information, see [Add your gateways and wireless devices to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [DisassociateWirelessGatewayFromCertificate](#) in *AWS CLI Command Reference*.

disassociate-wireless-gateway-from-thing

The following code example shows how to use `disassociate-wireless-gateway-from-thing`.

AWS CLI

To disassociate the thing from the wireless gateway

The following `disassociate-wireless-gateway-from-thing` example disassociates a wireless gateway from its currently associated thing.

```
aws iotwireless disassociate-wireless-gateway-from-thing \  
  --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

This command produces no output.

For more information, see [Add your gateways and wireless devices to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [DisassociateWirelessGatewayFromThing](#) in *AWS CLI Command Reference*.

get-destination

The following code example shows how to use `get-destination`.

AWS CLI

To get information about an IoT wireless destination

The following `get-destination` example gets information about the destination resource with the name `IoTWirelessDestination` that you created.

```
aws iotwireless get-destination \  
  --name "IoTWirelessDestination"
```

Output:

```
{
  "Arn": "arn:aws:iotwireless:us-east-1:123456789012:Destination/
IoTWirelessDestination",
  "Name": "IoTWirelessDestination",
  "Expression": "IoTWirelessRule",
  "ExpressionType": "RuleName",
  "RoleArn": "arn:aws:iam::123456789012:role/IoTWirelessDestinationRole"
}
```

For more information, see [Add destinations to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetDestination](#) in *AWS CLI Command Reference*.

get-device-profile

The following code example shows how to use `get-device-profile`.

AWS CLI

To get information about a device profile

The following `get-device-profile` example gets information about the device profile with the specified ID that you created.

```
aws iotwireless get-device-profile \
  --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

Output:

```
{
  "Arn": "arn:aws:iotwireless:us-east-1:123456789012:DeviceProfile/12345678-
a1b2-3c45-67d8-e90fa1b2c34d",
  "Id": "12345678-a1b2-3c45-67d8-e90fa1b2c34d",
  "LoRaWAN": {
    "MacVersion": "1.0.3",
    "MaxDutyCycle": 10,
    "Supports32BitFCnt": false,
    "RegParamsRevision": "RP002-1.0.1",
    "SupportsJoin": true,
    "RfRegion": "US915",
```

```
"MaxEirp": 13,  
"SupportsClassB": false,  
"SupportsClassC": false  
}  
}
```

For more information, see [Add profiles to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetDeviceProfile](#) in *AWS CLI Command Reference*.

get-partner-account

The following code example shows how to use `get-partner-account`.

AWS CLI

To get the partner account information

The following `get-partner-account` example gets information about your Sidewalk account that has the following ID.

```
aws iotwireless get-partner-account \  
  --partner-account-id "12345678901234" \  
  --partner-type "Sidewalk"
```

Output:

```
{  
  "Sidewalk": {  
    "AmazonId": "12345678901234",  
    "Fingerprint":  
    "a123b45c6d78e9f012a34cd5e6a7890b12c3d45e6f78a1b234c56d7e890a1234"  
  },  
  "AccountLinked": false  
}
```

For more information, see [Amazon Sidewalk Integration for AWS IoT Core](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetPartnerAccount](#) in *AWS CLI Command Reference*.

get-service-endpoint

The following code example shows how to use `get-service-endpoint`.

AWS CLI

To get the service endpoint

The following `get-service-endpoint` example gets the account-specific endpoint for CUPS protocol.

```
aws iotwireless get-service-endpoint
```

Output:

```
{
  "ServiceType": "CUPS",
  "ServiceEndpoint": "https://A1RMKZ37ACAG0T.cups.lorawan.us-east-1.amazonaws.com:443",
  "ServerTrust": "-----BEGIN CERTIFICATE-----\n
MIIESTCCAzGgAwIBAgITBn+UV4WH6Kx33rJTMlu8mYtWDTANBgkqhkiG9w0BAQsF\n
ADA5MQswCQYDVQQGEwJVUzEPMA0GA1UEChMGQW1UEChMGQW1hem9uMRkwFwYDVQQDEExBBbWF6\n
b24gUm9vdCBDQSAxMB4XDTE1MTAyMjAwMDAwMFoXDTE1MTAxOTAwMDAwMFowRjEL\n
MAKGA1UEBhMVCVVMxMjAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAw\n
IDFCMQ8wDQYDVQQDEwZBbWF6b24wgwEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEK\n
AoIBAQCDCThZn3c68asg3Wuw6MLAd5tES6BIOsMzoKcG5b1PVo+sDORrMd4f2AbnZ\n
cMzPa43j4wNxp1ty6aUKk4T1qe9B0wKFjwK6zmxxLVYo7bHViXsP1J6q0MpFge5\n
b1DP+18x+B26A0piiQ0uPkfyDyeR4xQghfj66Yo19V+emU3nazfvpFA+R0z6WoVm\n
B5x+F2pV8xeKNR7u6azDdu5YVX1Tawp1mxRC1+WsAYmz6qP+z8ArDITC2FMVy2fw\n
0IjK0tEXc/VfmtTFch5+AfGYMGmqqvJ6LcXiAhqG5TI+Dr0RtM88k+8XUBCeQ8IG\n
KuANaL7TiItKZYxK1MMuTJtV9Ib1AgMBAAGjggE7MIIBNzASBgNVHRMBAf8ECDAG\n
AQH/AgEAMA4GA1UdDwEB/wQEAwIBhjAdBgNVHQ4EFgQUWaRmBlKge5WSPK0UByew\n
dFv5PdAwHwYDVR0jBBgwFoAUhBjMhTTsvAyU1C4IWZzHshB0CggwewYIKwYBBQUH\n
AQEEbzBtMC8GCCsGAQUFBzABhiNodHRwOi8vb2Nzc5yb290Y2ExLmFtYXpvbnRy\n
dXN0LmNvbTA6BgggrBgEFBQcwAoYuaHR0cDovL2NydC5yb290Y2ExLmFtYXpvbnRy\n
dXN0LmNvbS9yb290Y2ExLmN1cjA/BgNVHR8EODA2MDSGmQAwHi5odHRwOi8vY3Js\n
LnJvb3RjYtEuYW1hem9udHJ1c3QuY29tL3Jvb3RjYtEuY3JsMBMGA1UdIAQMMAow\n
CAYGZ4EMAQIBMA0GCSqGSIb3DQEBcWUAA4IBAQCfkr41u3nPo4FCH0TjY3NTOVI1\n
59Gt/a6ZiqyJEi+752+a1U5y6iAwYfmXss2lJwJFqMp2PphKg5625kXg8kP2CN5t\n
6G7bMQcT8C8xDZntYtd7WPD8UZiRKAJPBXa30/AbwuZe0GaFEQ8ugcYQgSn+IGBI\n
8/LwhBNTZTUVEWuCUUBVV18YtbAiPq3yXqMB480z+ctBWuZSkbvkNodPLamkB2g1\n
upRyzQ7qDn1X8nn8N8V7YJ6y68AtkHcNSRAnpTitxBKjtkPISLMVCx7i4hncxHZS\n
yLyKQXhw2W2Xs0qLeC1etA+jTGDK4UfLeC0SF7FSi8o5LL21L8IzApar2pR/\n
```

```
-----END CERTIFICATE-----\n"}
```

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetServiceEndpoint](#) in *AWS CLI Command Reference*.

get-service-profile

The following code example shows how to use `get-service-profile`.

AWS CLI

To get information about a service profile

The following `get-service-profile` example gets information about the service profile with the specified ID that you created.

```
aws iotwireless get-service-profile \
  --id "12345678-a1b2-3c45-67d8-e90fa1b2c34d"
```

Output:

```
{
  "Arn": "arn:aws:iotwireless:us-east-1:651419225604:ServiceProfile/538185bb-
d7e7-4b95-96a0-c51aa4a5b9a0",
  "Id": "12345678-a1b2-3c45-67d8-e90fa1b2c34d",
  "LoRaWAN": {
    "HrAllowed": false,
    "NwkGeoLoc": false,
    "DrMax": 15,
    "UlBucketSize": 4096,
    "PrAllowed": false,
    "ReportDevStatusBattery": false,
    "DrMin": 0,
    "DlRate": 60,
    "AddGwMetadata": false,
    "ReportDevStatusMargin": false,
    "MinGwDiversity": 1,
    "RaAllowed": false,
    "DlBucketSize": 4096,
    "DevStatusReqFreq": 24,
```



```
    "TargetPer": 5,  
    "UlRate": 60  
  }  
}
```

For more information, see [Add profiles to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetServiceProfile](#) in *AWS CLI Command Reference*.

get-wireless-device-statistics

The following code example shows how to use `get-wireless-device-statistics`.

AWS CLI

To get operating information about a wireless device

The following `get-wireless-device-statistics` example gets operating information about a wireless device.

```
aws iotwireless get-wireless-device-statistics \  
  --wireless-device-id "1ffd32c8-8130-4194-96df-622f072a315f"
```

Output:

```
{  
  "WirelessDeviceId": "1ffd32c8-8130-4194-96df-622f072a315f"  
}
```

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetWirelessDeviceStatistics](#) in *AWS CLI Command Reference*.

get-wireless-device

The following code example shows how to use `get-wireless-device`.

AWS CLI

To get information about the wireless device

The following `get-wireless-device` example lists the available widgets in your AWS account.

```
aws iotwireless get-wireless-device \  
  --identifier "1ffd32c8-8130-4194-96df-622f072a315f" \  
  --identifier-type WirelessDeviceID
```

Output:

```
{  
  "Name": "myLoRaWANDevice",  
  "ThingArn": "arn:aws:iot:us-east-1:123456789012:thing/44b87eb4-9bce-423d-  
b5fc-973f5ecc358b",  
  "DestinationName": "IoTWirelessDestination",  
  "Id": "1ffd32c8-8130-4194-96df-622f072a315f",  
  "ThingName": "44b87eb4-9bce-423d-b5fc-973f5ecc358b",  
  "Type": "LoRaWAN",  
  "LoRaWAN": {  
    "DeviceProfileId": "ab0c23d3-b001-45ef-6a01-2bc3de4f5333",  
    "ServiceProfileId": "fe98dc76-cd12-001e-2d34-5550432da100",  
    "OtaaV1_1": {  
      "AppKey": "3f4ca100e2fc675ea123f4eb12c4a012",  
      "JoinEui": "b4c231a359bc2e3d",  
      "NwkKey": "01c3f004a2d6efffe32c4eda14bcd2b4"  
    },  
    "DevEui": "ac12efc654d23fc2"  
  },  
  "Arn": "arn:aws:iotwireless:us-  
east-1:123456789012:WirelessDevice/1ffd32c8-8130-4194-96df-622f072a315f",  
  "Description": "My LoRaWAN wireless device"  
}
```

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetWirelessDevice](#) in *AWS CLI Command Reference*.

get-wireless-gateway-certificate

The following code example shows how to use `get-wireless-gateway-certificate`.

AWS CLI

To get the ID of a certificate associated with a wireless gateway

The following `get-wireless-gateway-certificate` example gets the certificate ID associated with a wireless gateway that has the specified ID.

```
aws iotwireless get-wireless-gateway-certificate \  
  --id "6c44ab31-8b4d-407a-bed3-19b6c7cda551"
```

Output:

```
{  
  "IotCertificateId":  
  "8ea4aeae3db34c78cce75d9abd830356869ead6972997e0603e5fd032c804b6f"  
}
```

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetWirelessGatewayCertificate](#) in *AWS CLI Command Reference*.

get-wireless-gateway-firmware-information

The following code example shows how to use `get-wireless-gateway-firmware-information`.

AWS CLI

To get firmware information about a wireless gateway

The following `get-wireless-gateway-firmware-information` example gets firmware version and other information about a wireless gateway.

```
aws iotwireless get-wireless-gateway-firmware-information \  
  --id "3039b406-5cc9-4307-925b-9948c63da25b"
```

Output:

```
{  
  "LoRaWAN" :{
```

```
    "CurrentVersion" :{
      "PackageVersion" : "1.0.0",
      "Station" : "2.0.5",
      "Model" : "linux"
    }
  }
}
```

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetWirelessGatewayFirmwareInformation](#) in *AWS CLI Command Reference*.

get-wireless-gateway-statistics

The following code example shows how to use `get-wireless-gateway-statistics`.

AWS CLI

To get operating information about a wireless gateway

The following `get-wireless-gateway-statistics` example gets operating information about a wireless gateway.

```
aws iotwireless get-wireless-gateway-statistics \
  --wireless-gateway-id "3039b406-5cc9-4307-925b-9948c63da25b"
```

Output:

```
{
  "WirelessGatewayId": "3039b406-5cc9-4307-925b-9948c63da25b"
}
```

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetWirelessGatewayStatistics](#) in *AWS CLI Command Reference*.

get-wireless-gateway-task-definition

The following code example shows how to use `get-wireless-gateway-task-definition`.

AWS CLI

To get information about a wireless gateway task definition

The following `get-wireless-gateway-task-definition` example gets information about the wireless task definition with the specified ID.

```
aws iotwireless get-wireless-gateway-task-definition \  
  --id "b7d3baad-25c7-35e7-a4e1-1683a0d61da9"
```

Output:

```
{  
  "AutoCreateTasks": true,  
  "Name": "TestAutoUpdate",  
  "Update": {  
    "UpdateDataSource" : "s3://cupsalphagafirmwarebin/station",  
    "UpdateDataRole" : "arn:aws:iam::001234567890:role/SDK_Test_Role",  
    "LoRaWAN" : {  
      "CurrentVersion" : {  
        "PackageVersion" : "1.0.0",  
        "Station" : "2.0.5",  
        "Model" : "linux"  
      },  
      "UpdateVersion" : {  
        "PackageVersion" : "1.0.1",  
        "Station" : "2.0.5",  
        "Model" : "minihub"  
      }  
    }  
  }  
}
```

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetWirelessGatewayTaskDefinition](#) in *AWS CLI Command Reference*.

`get-wireless-gateway-task`

The following code example shows how to use `get-wireless-gateway-task`.

AWS CLI

To get information about the wireless gateway task

The following `get-wireless-gateway-task` example gets information about the wireless gateway task with the specified ID.

```
aws iotwireless get-wireless-gateway-task \  
  --id "11693a46-6866-47c3-a031-c9a616e7644b"
```

Output:

```
{  
  "WirelessGatewayId": "6c44ab31-8b4d-407a-bed3-19b6c7cda551",  
  "WirelessGatewayTaskDefinitionId": "b7d3baad-25c7-35e7-a4e1-1683a0d61da9",  
  "Status": "Success"  
}
```

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetWirelessGatewayTask](#) in *AWS CLI Command Reference*.

get-wireless-gateway

The following code example shows how to use `get-wireless-gateway`.

AWS CLI

To get information about a wireless gateway

The following `get-wireless-gateway` example gets information about the wireless gateway `myFirstLoRaWANGateway`.

```
aws iotwireless get-wireless-gateway \  
  --identifier "12345678-a1b2-3c45-67d8-e90fa1b2c34d" \  
  --identifier-type WirelessGatewayId
```

Output:

```
{
```

```
"Description": "My first LoRaWAN gateway",
"ThingArn": "arn:aws:iot:us-east-1:123456789012:thing/a1b2c3d4-5678-90ab-
cdef-12ab345c67de",
"LoRaWAN": {
  "RfRegion": "US915",
  "GatewayEui": "a1b2c3d4567890ab"
},
"ThingName": "a1b2c3d4-5678-90ab-cdef-12ab345c67de",
"Id": "12345678-a1b2-3c45-67d8-e90fa1b2c34d",
"Arn": "arn:aws:iotwireless:us-
east-1:123456789012:WirelessGateway/6c44ab31-8b4d-407a-bed3-19b6c7cda551",
"Name": "myFirstLoRaWANGateway"
}
```

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [GetWirelessGateway](#) in *AWS CLI Command Reference*.

list-destinations

The following code example shows how to use `list-destinations`.

AWS CLI

To list the wireless destinations

The following `list-destinations` example lists the available destinations registered to your AWS account.

```
aws iotwireless list-destinations
```

Output:

```
{
  "DestinationList": [
    {
      "Arn": "arn:aws:iotwireless:us-east-1:123456789012:Destination/
IoTWirelessDestination",
      "Name": "IoTWirelessDestination",
      "Expression": "IoTWirelessRule",
```

```

        "Description": "Destination for messages processed using
IoTWirelessRule",
        "RoleArn": "arn:aws:iam::123456789012:role/IoTWirelessDestinationRole"
    },
    {
        "Arn": "arn:aws:iotwireless:us-east-1:123456789012:Destination/
IoTWirelessDestination2",
        "Name": "IoTWirelessDestination2",
        "Expression": "IoTWirelessRule2",
        "RoleArn": "arn:aws:iam::123456789012:role/IoTWirelessDestinationRole"
    }
]
}

```

For more information, see [Add destinations to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListDestinations](#) in *AWS CLI Command Reference*.

list-device-profiles

The following code example shows how to use `list-device-profiles`.

AWS CLI

To list the device profiles

The following `list-device-profiles` example lists the available device profiles registered to your AWS account.

```
aws iotwireless list-device-profiles
```

Output:

```

{
  "DeviceProfileList": [
    {
      "Id": "12345678-a1b2-3c45-67d8-e90fa1b2c34d",
      "Arn": "arn:aws:iotwireless:us-
east-1:123456789012:DeviceProfile/12345678-a1b2-3c45-67d8-e90fa1b2c34d"
    },
  ],
}

```



```
{
  "Id": "a1b2c3d4-5678-90ab-cdef-12ab345c67de",
  "Arn": "arn:aws:iotwireless:us-east-1:123456789012:DeviceProfile/
a1b2c3d4-5678-90ab-cdef-12ab345c67de"
}
]
```

For more information, see [Add profiles to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListDeviceProfiles](#) in *AWS CLI Command Reference*.

list-partner-accounts

The following code example shows how to use `list-partner-accounts`.

AWS CLI

To list the partner accounts

The following `list-partner-accounts` example lists the available partner accounts associated with your AWS account.

```
aws iotwireless list-partner-accounts
```

Output:

```
{
  "Sidewalk": [
    {
      "AmazonId": "78965678771228",
      "Fingerprint":
"bd96d8ef66dbfd2160eb60e156849e82ad7018b8b73c1ba0b4fc65c32498ee35"
    },
    {
      "AmazonId": "89656787651228",
      "Fingerprint":
"bc5e99e151c07be14be7e6603e4489c53f858b271213a36ebe3370777ba06e9b"
    }
  ]
}
```

```
}
```

For more information, see [Amazon Sidewalk Integration for AWS IoT Core](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListPartnerAccounts](#) in *AWS CLI Command Reference*.

list-service-profiles

The following code example shows how to use `list-service-profiles`.

AWS CLI

To list the service profiles

The following `list-service-profiles` example lists the available service profiles registered to your AWS account.

```
aws iotwireless list-service-profiles
```

Output:

```
{
  "ServiceProfileList": [
    {
      "Id": "12345678-a1b2-3c45-67d8-e90fa1b2c34d",
      "Arn": "arn:aws:iotwireless:us-east-1:123456789012:ServiceProfile/538185bb-d7e7-4b95-96a0-c51aa4a5b9a0"
    },
    {
      "Id": "a1b2c3d4-5678-90ab-cdef-12ab345c67de",
      "Arn": "arn:aws:iotwireless:us-east-1:123456789012:ServiceProfile/ea8bc823-5d13-472e-8d26-9550737d8100"
    }
  ]
}
```

For more information, see [Add profiles to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListServiceProfiles](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags assigned to the resource

The following `list-tags-for-resource` example lists the tags assigned to a wireless destination resource.

```
aws iotwireless list-tags-for-resource \  
  --resource-arn "arn:aws:iotwireless:us-east-1:123456789012:Destination/  
IoTWirelessDestination"
```

Output:

```
{  
  "Tags": [  
    {  
      "Value": "MyValue",  
      "Key": "MyTag"  
    }  
  ]  
}
```

For more information, see [Describe your AWS IoT Core for LoRaWAN resources](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

list-wireless-devices

The following code example shows how to use `list-wireless-devices`.

AWS CLI

To list the available wireless devices

The following `list-wireless-devices` example lists the available wireless devices registered to your AWS account.

```
aws iotwireless list-wireless-devices
```

Output:

```
{
  "WirelessDeviceList": [
    {
      "Name": "myLoRaWANDevice",
      "DestinationName": "IoTWirelessDestination",
      "Id": "1fffd32c8-8130-4194-96df-622f072a315f",
      "Type": "LoRaWAN",
      "LoRaWAN": {
        "DevEui": "ac12efc654d23fc2"
      },
      "Arn": "arn:aws:iotwireless:us-east-1:123456789012:WirelessDevice/1fffd32c8-8130-4194-96df-622f072a315f"
    }
  ]
}
```

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListWirelessDevices](#) in *AWS CLI Command Reference*.

list-wireless-gateway-task-definitions

The following code example shows how to use `list-wireless-gateway-task-definitions`.

AWS CLI**To list the wireless gateway task definitions**

The following `list-wireless-gateway-task-definitions` example lists the available wireless gateway task definitions registered to your AWS account.

```
aws iotwireless list-wireless-gateway-task-definitions
```

Output:

```
{
```

```
"TaskDefinitions": [  
  {  
    "Id": "b7d3baad-25c7-35e7-a4e1-1683a0d61da9",  
    "LoRaWAN" :  
      {  
        "CurrentVersion" :{  
          "PackageVersion" : "1.0.0",  
          "Station" : "2.0.5",  
          "Model" : "linux"  
        },  
        "UpdateVersion" :{  
          "PackageVersion" : "1.0.1",  
          "Station" : "2.0.5",  
          "Model" : "minihub"  
        }  
      }  
    }  
  ]  
}
```

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListWirelessGatewayTaskDefinitions](#) in *AWS CLI Command Reference*.

list-wireless-gateways

The following code example shows how to use `list-wireless-gateways`.

AWS CLI

To list the wireless gateways

The following `list-wireless-gateways` example lists the available wireless gateways in your AWS account.

```
aws iotwireless list-wireless-gateways
```

Output:

```
{  
  "WirelessGatewayList": [  
    {  
      "Id": "b7d3baad-25c7-35e7-a4e1-1683a0d61da9",  
      "LoRaWAN": {  
        "CurrentVersion": {  
          "PackageVersion": "1.0.0",  
          "Station": "2.0.5",  
          "Model": "linux"  
        },  
        "UpdateVersion": {  
          "PackageVersion": "1.0.1",  
          "Station": "2.0.5",  
          "Model": "minihub"  
        }  
      }  
    }  
  ]  
}
```

```

    {
      "Description": "My first LoRaWAN gateway",
      "LoRaWAN": {
        "RfRegion": "US915",
        "GatewayEui": "dac632ebc01d23e4"
      },
      "Id": "3039b406-5cc9-4307-925b-9948c63da25b",
      "Arn": "arn:aws:iotwireless:us-
east-1:123456789012:WirelessGateway/3039b406-5cc9-4307-925b-9948c63da25b",
      "Name": "myFirstLoRaWANGateway"
    },
    {
      "Description": "My second LoRaWAN gateway",
      "LoRaWAN": {
        "RfRegion": "US915",
        "GatewayEui": "cda123fffe92ecd2"
      },
      "Id": "3285bdc7-5a12-4991-84ed-dadca65e342e",
      "Arn": "arn:aws:iotwireless:us-
east-1:123456789012:WirelessGateway/3285bdc7-5a12-4991-84ed-dadca65e342e",
      "Name": "mySecondLoRaWANGateway"
    }
  ]
}

```

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [ListWirelessGateways](#) in *AWS CLI Command Reference*.

send-data-to-wireless-device

The following code example shows how to use `send-data-to-wireless-device`.

AWS CLI

To send data to the wireless device

The following `send-data-to-wireless-device` example sends a decrypted application data frame to the wireless device.

```

aws iotwireless send-data-to-wireless-device \
  --id "11aa5eae-2f56-4b8e-a023-b28d98494e49" \

```

```
--transmit-mode "1" \  
--payload-data "SGVsbG8gVG8gRGV2c2lt" \  
--wireless-metadata LoRaWAN={FPort=1}
```

Output:

```
{  
  MessageId: "6011dd36-0043d6eb-0072-0008"  
}
```

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [SendDataToWirelessDevice](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To specify a tag key and value for a resource

The following `tag-resource` example tags the wireless destination `IoTWirelessDestination` with the key `MyTag` and value `MyValue`.

```
aws iotwireless tag-resource \  
  --resource-arn "arn:aws:iotwireless:us-east-1:651419225604:Destination/  
IoTWirelessDestination" \  
  --tags Key="MyTag",Value="MyValue"
```

This command produces no output.

For more information, see [Describe your AWS IoT Core for LoRaWAN resources](#) in the *AWS IoT Developers Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

test-wireless-device

The following code example shows how to use `test-wireless-device`.

AWS CLI

To test the wireless device

The following `test-wireless-device` example sends uplink data of Hello to a device with specified ID.

```
aws iotwireless test-wireless-device \  
  --id "11aa5eae-2f56-4b8e-a023-b28d98494e49"
```

Output:

```
{  
  Result: "Test succeeded. one message is sent with payload: hello"  
}
```

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [TestWirelessDevice](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove one or more tags from a resource

The following `untag-resource` example removes the tag `MyTag` and its value from the wireless destination `IoTWirelessDestination`.

```
aws iotwireless untag-resource \  
  --resource-arn "arn:aws:iotwireless:us-east-1:123456789012:Destination/  
IoTWirelessDestination" \  
  --tag-keys "MyTag"
```

This command produces no output.

For more information, see [Describe your AWS IoT Core for LoRaWAN resources](#) in the *AWS IoT Developers Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-destination

The following code example shows how to use `update-destination`.

AWS CLI

To update the properties of a destination

The following `update-destination` example updates the `description` property of a wireless destination.

```
aws iotwireless update-destination \  
  --name "IoTWirelessDestination" \  
  --description "Destination for messages processed using IoTWirelessRule"
```

This command produces no output.

For more information, see [Add destinations to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [UpdateDestination](#) in *AWS CLI Command Reference*.

update-partner-account

The following code example shows how to use `update-partner-account`.

AWS CLI

To update the properties of a partner account

The following `update-partner-account` updates the `AppServerPrivateKey` for the account that has the specified ID.

```
aws iotwireless update-partner-account \  
  --partner-account-id "78965678771228" \  
  --partner-type "Sidewalk" \  
  --sidewalk  
  AppServerPrivateKey="f798ab4899346a88599180fee9e14fa1ada7b6df989425b7c6d2146dd6c815bb"
```

This command produces no output.

For more information, see [Amazon Sidewalk Integration for AWS IoT Core](#) in the *AWS IoT Developers Guide*.

- For API details, see [UpdatePartnerAccount](#) in *AWS CLI Command Reference*.

update-wireless-device

The following code example shows how to use `update-wireless-device`.

AWS CLI

To update the properties of a wireless device

The following `update-wireless-device` example updates the properties of a wireless device registered to your AWS account.

```
aws iotwireless update-wireless-device \  
  --id "1fffd32c8-8130-4194-96df-622f072a315f" \  
  --destination-name IoTWirelessDestination2 \  
  --description "Using my first LoRaWAN device"
```

This command produces no output.

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [UpdateWirelessDevice](#) in *AWS CLI Command Reference*.

update-wireless-gateway

The following code example shows how to use `update-wireless-gateway`.

AWS CLI

To update the wireless gateway

The following `update-wireless-gateway` example updates the description of your wireless gateway.

```
aws iotwireless update-wireless-gateway \  
  --id "3285bdc7-5a12-4991-84ed-dadca65e342e" \  
  --description "Using my LoRaWAN gateway"
```

This command produces no output.

For more information, see [Connecting devices and gateways to AWS IoT Core for LoRaWAN](#) in the *AWS IoT Developers Guide*.

- For API details, see [UpdateWirelessGateway](#) in *AWS CLI Command Reference*.

Amazon IVS examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon IVS.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

batch-get-channel

The following code example shows how to use `batch-get-channel`.

AWS CLI

To get channel configuration information about multiple channels

The following `batch-get-channel` example lists information about the specified channels.

```
aws ivs batch-get-channel \  
  --arns arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh \  
        arn:aws:ivs:us-west-2:123456789012:channel/efghEFGHijkl
```

Output:

```
{
  "channels": [
    {
      "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
      "authorized": false,
      "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
      "insecureIngest": false,
      "latencyMode": "LOW",
      "name": "channel-1",
      "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/video/v1/us-west-2.123456789012.channel-1.abcdEFGH.m3u8",
      "preset": "",
      "playbackRestrictionPolicyArn": "",
      "recordingConfigurationArn": "arn:aws:ivs:us-west-2:123456789012:recording-configuration/ABCD12cdEFgh",
      "srt": {
        "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
        "passphrase":
"AB1C2defGHijklMNop3PqQRstUvwxyzABCDefghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
      },
      "tags": {},
      "type": "STANDARD"
    },
    {
      "arn": "arn:aws:ivs:us-west-2:123456789012:channel/efghEFGHijkl",
      "authorized": false,
      "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
      "insecureIngest": true,
      "latencyMode": "LOW",
      "name": "channel-2",
      "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/video/v1/us-west-2.123456789012.channel-2.abcdEFGH.m3u8",
      "preset": "",
      "playbackRestrictionPolicyArn": "arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/ABCdef34ghIJ",
      "recordingConfigurationArn": "",
      "srt": {
        "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
        "passphrase":
"BA1C2defGHijklMNop3PqQRstUvwxyzABCDefghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
      },
      "tags": {}
    }
  ]
}
```

```

        "type": "STANDARD"
      }
    ]
  }

```

For more information, see [Create a Channel](#) in the *IVS Low-Latency User Guide*.

- For API details, see [BatchGetChannel](#) in *AWS CLI Command Reference*.

batch-get-stream-key

The following code example shows how to use `batch-get-stream-key`.

AWS CLI

To get information about multiple stream keys

The following `batch-get-stream-key` example gets information about the specified stream keys.

```

aws ivs batch-get-stream-key \
  --arns arn:aws:ivs:us-west-2:123456789012:stream-key/skSKABCDefgh \
  arn:aws:ivs:us-west-2:123456789012:stream-key/skSKIJKLmnop

```

Output:

```

{
  "streamKeys": [
    {
      "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/skSKABCDefgh",
      "value": "sk_us-west-2_abcdABCDefgh_567890abcdef",
      "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
      "tags": {}
    },
    {
      "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/skSKIJKLmnop",
      "value": "sk_us-west-2_abcdABCDefgh_567890ghijkl",
      "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
      "tags": {}
    }
  ]
}

```

For more information, see [Create a Channel](#) in the *IVS Low-Latency User Guide*.

- For API details, see [BatchGetStreamKey](#) in *AWS CLI Command Reference*.

batch-start-viewer-session-revocation

The following code example shows how to use `batch-start-viewer-session-revocation`.

AWS CLI

To revoke viewer sessions for multiple channel-ARN and viewer-ID pairs

The following `batch-start-viewer-session-revocation` example performs session revocation on multiple channel-ARN and viewer-ID pairs simultaneously. The request may complete normally but return values in the `errors` field if the caller does not have permission to revoke specified session.

```
aws ivs batch-start-viewer-session-revocation \
  --viewer-sessions '[{"channelArn":"arn:aws:ivs:us-west-2:123456789012:channel/
abcdABCDefgh1","viewerId":"abcdefg1","viewerSessionVersionsLessThanOrEqualTo":1234567890},
\
  {"channelArn":"arn:aws:ivs:us-west-2:123456789012:channel/
abcdABCDefgh2","viewerId":"abcdefg2","viewerSessionVersionsLessThanOrEqualTo":1234567890}]'
```

Output:

```
{
  "errors": [
    {
      "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/
abcdABCDefgh1",
      "viewerId": "abcdefg1",
      "code": "403",
      "message": "not authorized",
    },
    {
      "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/
abcdABCDefgh2",
      "viewerId": "abcdefg2",
      "code": "403",
      "message": "not authorized",
    }
  ]
}
```

```
]
}
```

For more information, see [Setting Up Private Channels](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [BatchStartViewerSessionRevocation](#) in *AWS CLI Command Reference*.

create-channel

The following code example shows how to use `create-channel`.

AWS CLI

Example 1: To create a channel with no recording

The following `create-channel` example creates a new channel and an associated stream key to start streaming.

```
aws ivs create-channel \
  --name "test-channel" \
  --no-insecure-ingest
```

Output:

```
{
  "channel": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
    "authorized": false,
    "name": "test-channel",
    "latencyMode": "LOW",
    "playbackRestrictionPolicyArn": "",
    "recordingConfigurationArn": "",
    "srt": {
      "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
      "passphrase":
"AB1C2defGHijkLMNo3PqQRstUvwxyzaBCDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
    },
    "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
    "insecureIngest": false,
    "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
```

```

    "preset": "",
    "tags": {},
    "type": "STANDARD"
  },
  "streamKey": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/g1H2I3j4k5L6",
    "value": "sk_us-west-2_abcdABCDefgh_567890abcdef",
    "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
    "tags": {}
  }
}

```

For more information, see [Create a Channel](#) in the *IVS Low-Latency User Guide*.

Example 2: To create a channel with recording enabled, using the RecordingConfiguration resource specified by its ARN

The following `create-channel` example creates a new channel and an associated stream key to start streaming, and sets up recording for the channel.

```

aws ivs create-channel \
  --name test-channel-with-recording \
  --insecure-ingest \
  --recording-configuration-arn "arn:aws:ivs:us-west-2:123456789012:recording-
configuration/ABCD12cdEFgh"

```

Output:

```

{
  "channel": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
    "name": "test-channel-with-recording",
    "latencyMode": "LOW",
    "type": "STANDARD",
    "playbackRestrictionPolicyArn": "",
    "recordingConfigurationArn": "arn:aws:ivs:us-west-2:123456789012:recording-
configuration/ABCD12cdEFgh",
    "srt": {
      "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
      "passphrase":
"BA1C2defGHijklMN03PqQRstUvwxyzaBCDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
    },
  },
}

```



```

    "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
    "insecureIngest": true,
    "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
    "preset": "",
    "authorized": false,
    "tags": {},
    "type": "STANDARD"
  },
  "streamKey": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/abcdABCDefgh",
    "value": "sk_us-west-2_abcdABCDefgh_567890abcdef",
    "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
    "tags": {}
  }
}

```

For more information, see [Record to Amazon S3](#) in the *IVS Low-Latency User Guide*.

Example 3: To create a channel with a playback restriction policy specified by its ARN

The following `create-channel` example creates a new channel and an associated stream key to start streaming, and sets up a playback restriction policy for the channel.

```

aws ivs create-channel \
  --name test-channel-with-playback-restriction-policy \
  --insecure-ingest \
  --playback-restriction-policy-arn "arn:aws:ivs:us-west-2:123456789012:playback-
restriction-policy/ABcdef34ghIJ"

```

Output:

```

{
  "channel": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
    "name": "test-channel-with-playback-restriction-policy",
    "latencyMode": "LOW",
    "type": "STANDARD",
    "playbackRestrictionPolicyArn": "arn:aws:ivs:us-
west-2:123456789012:playback-restriction-policy/ABcdef34ghIJ",
    "recordingConfigurationArn": "",
    "srt": {

```

```

        "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
        "passphrase":
"AB1C2edfGHijkLMNo3PqQRstUvwxyzaBCDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
    },
    "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
    "insecureIngest": true,
    "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
    "preset": "",
    "authorized": false,
    "tags": {},
    "type": "STANDARD"
},
"streamKey": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/abcdABCDefgh",
    "value": "sk_us-west-2_abcdABCDefgh_567890abcdef",
    "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
    "tags": {}
}
}

```

For more information, see [Undesired Content and Viewers](#) in the *IVS Low-Latency User Guide*.

- For API details, see [CreateChannel](#) in *AWS CLI Command Reference*.

create-playback-restriction-policy

The following code example shows how to use create-playback-restriction-policy.

AWS CLI

To create a playback restriction policy

The following create-playback-restriction-policy example creates a new playback restriction policy.

```

aws ivs create-playback-restriction-policy \
  --name "test-playback-restriction-policy" \
  --enable-strict-origin-enforcement \
  --tags "key1=value1, key2=value2" \
  --allowed-countries US MX \
  --allowed-origins https://www.website1.com https://www.website2.com

```

Output:

```
{
  "playbackRestrictionPolicy": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/
ABCdef34ghIJ",
    "allowedCountries": [
      "US",
      "MX"
    ],
    "allowedOrigins": [
      "https://www.website1.com",
      "https://www.website2.com"
    ],
    "enableStrictOriginEnforcement": true,
    "name": "test-playback-restriction-policy",
    "tags": {
      "key1": "value1",
      "key2": "value2"
    }
  }
}
```

For more information, see [Undesired Content and Viewers](#) in the *IVS Low-Latency User Guide*.

- For API details, see [CreatePlaybackRestrictionPolicy](#) in *AWS CLI Command Reference*.

create-recording-configuration

The following code example shows how to use create-recording-configuration.

AWS CLI**To create a RecordingConfiguration resource**

The following create-recording-configuration example creates a RecordingConfiguration resource to enable recording to Amazon S3.

```
aws ivs create-recording-configuration \
  --name "test-recording-config" \
  --recording-reconnect-window-seconds 60 \
  --tags "key1=value1, key2=value2" \
```

```

--rendition-configuration renditionSelection="CUSTOM",renditions="HD" \
--thumbnail-configuration
recordingMode="INTERVAL",targetIntervalSeconds=1,storage="LATEST",resolution="LOWEST_RESOLUTION" \
--destination-configuration s3={bucketName=demo-recording-bucket}

```

Output:

```

{
  "recordingConfiguration": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:recording-configuration/
ABcdef34ghIJ",
    "name": "test-recording-config",
    "destinationConfiguration": {
      "s3": {
        "bucketName": "demo-recording-bucket"
      }
    },
    "state": "CREATING",
    "tags": {
      "key1": "value1",
      "key2": "value2"
    },
    "thumbnailConfiguration": {
      "recordingMode": "INTERVAL",
      "targetIntervalSeconds": 1,
      "resolution": "LOWEST_RESOLUTION",
      "storage": [
        "LATEST"
      ]
    },
    "recordingReconnectWindowSeconds": 60,
    "renditionConfiguration": {
      "renditionSelection": "CUSTOM",
      "renditions": [
        "HD"
      ]
    }
  }
}

```

For more information, see [Record to Amazon S3](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [CreateRecordingConfiguration](#) in *AWS CLI Command Reference*.

create-stream-key

The following code example shows how to use `create-stream-key`.

AWS CLI

To create a stream key

The following `create-stream-key` example creates a stream key for a specified ARN (Amazon Resource Name).

```
aws ivs create-stream-key \  
  --channel-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh
```

Output:

```
{  
  "streamKey": {  
    "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/abcdABCDefgh",  
    "value": "sk_us-west-2_abcdABCDefgh_567890abcdef",  
    "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",  
    "tags": {}  
  }  
}
```

For more information, see [Create a Channel](#) in the *IVS Low-Latency User Guide*.

- For API details, see [CreateStreamKey](#) in *AWS CLI Command Reference*.

delete-channel

The following code example shows how to use `delete-channel`.

AWS CLI

To delete a channel and its associated stream keys

The following `delete-channel` example deletes the channel with the specified ARN (Amazon Resource Name).

```
aws ivs delete-channel \  
  --arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh
```

This command produces no output.

For more information, see [Create a Channel](#) in the *IVS Low-Latency User Guide*.

- For API details, see [DeleteChannel](#) in *AWS CLI Command Reference*.

delete-playback-key-pair

The following code example shows how to use `delete-playback-key-pair`.

AWS CLI

To delete a specified playback key pair

The following `delete-playback-key-pair` example returns the fingerprint of the specified key pair.

```
aws ivs delete-playback-key-pair \  
  --arn arn:aws:ivs:us-west-2:123456789012:playback-key/abcd1234efgh
```

This command produces no output.

For more information, see [Setting Up Private Channels](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [DeletePlaybackKeyPair](#) in *AWS CLI Command Reference*.

delete-playback-restriction-policy

The following code example shows how to use `delete-playback-restriction-policy`.

AWS CLI

To delete a playback restriction policy

The following `delete-playback-restriction-policy` example deletes the playback restriction policy with the specified policy ARN (Amazon Resource Name).

```
aws ivs delete-playback-restriction-policy \  
  --arn arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/abcd1234efgh
```

```
--arn "arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/ABCdef34ghIJ"
```

This command produces no output.

For more information, see [Undesired Content and Viewers](#) in the *IVS Low-Latency User Guide*.

- For API details, see [DeletePlaybackRestrictionPolicy](#) in *AWS CLI Command Reference*.

delete-recording-configuration

The following code example shows how to use delete-recording-configuration.

AWS CLI

To delete the RecordingConfiguration resource specified by its ARN

The following delete-recording-configuration example deletes the RecordingConfiguration resource with the specified ARN.

```
aws ivs delete-recording-configuration \  
  --arn "arn:aws:ivs:us-west-2:123456789012:recording-configuration/ABCdef34ghIJ"
```

This command produces no output.

For more information, see [Record to Amazon S3](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [DeleteRecordingConfiguration](#) in *AWS CLI Command Reference*.

delete-stream-key

The following code example shows how to use delete-stream-key.

AWS CLI

To delete a stream key

The following delete-stream-key example deletes the stream key for a specified ARN (Amazon Resource Name), so it can no longer be used to stream.

```
aws ivs delete-stream-key \  
  --arn "arn:aws:ivs:us-west-2:123456789012:stream-key/ABCdef34ghIJ"
```

```
--arn arn:aws:ivs:us-west-2:123456789012:stream-key/g1H2I3j4k5L6
```

This command produces no output.

For more information, see [Create a Channel](#) in the *IVS Low-Latency User Guide*.

- For API details, see [DeleteStreamKey](#) in *AWS CLI Command Reference*.

get-channel

The following code example shows how to use `get-channel`.

AWS CLI

To get a channel's configuration information

The following `get-channel` example gets the channel configuration for a specified channel ARN (Amazon Resource Name).

```
aws ivs get-channel \  
  --arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh
```

Output:

```
{  
  "channel": {  
    "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",  
    "name": "channel-1",  
    "latencyMode": "LOW",  
    "type": "STANDARD",  
    "playbackRestrictionPolicyArn": "",  
    "preset": "",  
    "recordingConfigurationArn": "arn:aws:ivs:us-west-2:123456789012:recording-  
configuration/ABCD12cdEFgh",  
    "srt": {  
      "endpoint": "a1b2c3d4e5f6.srt.live-video.net",  
      "passphrase":  
"AB1C2defGHijkLMNo3PqQRstUvwxyzaBCDEfghh4ijk1MN5opqrStuVWxyzAbCDEfghIJ"  
    },  
    "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",  
    "insecureIngest": false,  
    "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/  
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
```



```
    "tags": {}
  }
}
```

For more information, see [Create a Channel](#) in the *IVS Low-Latency User Guide*.

- For API details, see [GetChannel](#) in *AWS CLI Command Reference*.

get-playback-key-pair

The following code example shows how to use `get-playback-key-pair`.

AWS CLI

To get a specified playback key pair

The following `get-playback-key-pair` example returns the fingerprint of the specified key pair.

```
aws ivs get-playback-key-pair \
  --arn arn:aws:ivs:us-west-2:123456789012:playback-key/abcd1234efgh
```

Output:

```
{
  "keyPair": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:playback-key/abcd1234efgh",
    "name": "my-playback-key",
    "fingerprint": "0a:1b:2c:ab:cd:ef:34:56:70:b1:b2:71:01:2a:a3:72",
    "tags": {}
  }
}
```

For more information, see [Setting Up Private Channels](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [GetPlaybackKeyPair](#) in *AWS CLI Command Reference*.

get-playback-restriction-policy

The following code example shows how to use `get-playback-restriction-policy`.

AWS CLI

To get a playback restriction policy's configuration information

The following `get-playback-restriction-policy` example gets the playback restriction policy configuration with the specified policy ARN (Amazon Resource Name).

```
aws ivs get-playback-restriction-policy \
  --arn "arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/
  ABcdef34ghIJ"
```

Output:

```
{
  "playbackRestrictionPolicy": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/
  ABcdef34ghIJ",
    "allowedCountries": [
      "US",
      "MX"
    ],
    "allowedOrigins": [
      "https://www.website1.com",
      "https://www.website2.com"
    ],
    "enableStrictOriginEnforcement": true,
    "name": "test-playback-restriction-policy",
    "tags": {
      "key1": "value1",
      "key2": "value2"
    }
  }
}
```

For more information, see [Undesired Content and Viewers](#) in the *IVS Low-Latency User Guide*.

- For API details, see [GetPlaybackRestrictionPolicy](#) in *AWS CLI Command Reference*.

get-recording-configuration

The following code example shows how to use `get-recording-configuration`.

AWS CLI

To get information about a RecordingConfiguration resource

The following `get-recording-configuration` example gets information about the RecordingConfiguration resource for the specified ARN.

```
aws ivs get-recording-configuration \  
  --arn "arn:aws:ivs:us-west-2:123456789012:recording-configuration/ABcdef34ghIJ"
```

Output:

```
{  
  "recordingConfiguration": {  
    "arn": "arn:aws:ivs:us-west-2:123456789012:recording-configuration/  
ABcdef34ghIJ",  
    "destinationConfiguration": {  
      "s3": {  
        "bucketName": "demo-recording-bucket"  
      }  
    },  
    "name": "test-recording-config",  
    "recordingReconnectWindowSeconds": 60,  
    "state": "ACTIVE",  
    "tags": {  
      "key1" : "value1",  
      "key2" : "value2"  
    },  
    "thumbnailConfiguration": {  
      "recordingMode": "INTERVAL",  
      "targetIntervalSeconds": 1,  
      "resolution": "LOWEST_RESOLUTION",  
      "storage": [  
        "LATEST"  
      ]  
    },  
    "renditionConfiguration": {  
      "renditionSelection": "CUSTOM",  
      "renditions": [  
        "HD"  
      ]  
    }  
  }  
}
```

```
}
```

For more information, see [Record to Amazon S3](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [GetRecordingConfiguration](#) in *AWS CLI Command Reference*.

get-stream-key

The following code example shows how to use `get-stream-key`.

AWS CLI

To get information about a stream

The following `get-stream-key` example gets information about the specified stream key.

```
aws ivs get-stream-key \  
  --arn arn:aws:ivs:us-west-2:123456789012:stream-key/skSKABCDefgh --region=us-  
west-2
```

Output:

```
{  
  "streamKey": {  
    "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/skSKABCDefgh",  
    "value": "sk_us-west-2_abcdABCDefgh_567890abcdef",  
    "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",  
    "tags": {}  
  }  
}
```

For more information, see [Create a Channel](#) in the *IVS Low-Latency User Guide*.

- For API details, see [GetStreamKey](#) in *AWS CLI Command Reference*.

get-stream-session

The following code example shows how to use `get-stream-session`.

AWS CLI

To get metadata for a specified stream

The following `get-stream-session` example gets the metadata configuration for the specified channel ARN (Amazon Resource Name) and the specified stream; if `streamId` is not provided, the most recent stream for the channel is selected.

```
aws ivs get-stream-session \  
  --channel-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh \  
  --stream-id "mystream"
```

Output:

```
{  
  "streamSession": {  
    "streamId": "mystream1",  
    "startTime": "2023-06-26T19:09:28+00:00",  
    "channel": {  
      "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",  
      "name": "mychannel",  
      "latencyMode": "LOW",  
      "type": "STANDARD",  
      "recordingConfigurationArn": "arn:aws:ivs:us-  
west-2:123456789012:recording-configuration/ABCdef34ghIJ",  
      "srt": {  
        "endpoint": "a1b2c3d4e5f6.srt.live-video.net",  
        "passphrase":  
"AB1C2defGHijkLMNo3PqQRstUvwxyzaBCDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"  
      },  
      "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",  
      "playbackUrl": "url-string",  
      "authorized": false,  
      "insecureIngest": false,  
      "preset": ""  
    },  
    "ingestConfiguration": {  
      "video": {  
        "avcProfile": "Baseline",  
        "avcLevel": "4.2",  
        "codec": "avc1.42C02A",  
        "encoder": "Lavf58.45.100",  
        "targetBitrate": 8789062,  
        "targetFramerate": 60,  
        "videoHeight": 1080,  
        "videoWidth": 1920  
      },  
    },  
  },  
}
```

```
    "audio": {
      "codec": "mp4a.40.2",
      "targetBitrate": 46875,
      "sampleRate": 8000,
      "channels": 2
    }
  },
  "recordingConfiguration": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:recording-configuration/
ABcdef34ghIJ",
    "name": "test-recording-config",
    "destinationConfiguration": {
      "s3": {
        "bucketName": "demo-recording-bucket"
      }
    },
    "state": "ACTIVE",
    "tags": {
      "key1": "value1",
      "key2": "value2"
    },
    "thumbnailConfiguration": {
      "recordingMode": "INTERVAL",
      "targetIntervalSeconds": 1,
      "resolution": "LOWEST_RESOLUTION",
      "storage": [
        "LATEST"
      ]
    },
    "recordingReconnectWindowSeconds": 60,
    "renditionConfiguration": {
      "renditionSelection": "CUSTOM",
      "renditions": [
        "HD"
      ]
    }
  },
  "truncatedEvents": [
    {
      "name": "Recording Start",
      "type": "IVS Recording State Change",
      "eventTime": "2023-06-26T19:09:35+00:00"
    },
    {
```

```

        "name": "Stream Start",
        "type": "IVS Stream State Change",
        "eventTime": "2023-06-26T19:09:34+00:00"
      },
      {
        "name": "Session Created",
        "type": "IVS Stream State Change",
        "eventTime": "2023-06-26T19:09:28+00:00"
      }
    ]
  }
}

```

For more information, see [Create a Channel](#) in the *IVS Low-Latency User Guide*.

- For API details, see [GetStreamSession](#) in *AWS CLI Command Reference*.

get-stream

The following code example shows how to use `get-stream`.

AWS CLI

To get information about a stream

The following `get-stream` example gets information about the stream for the specified channel.

```

aws ivs get-stream \
  --channel-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh

```

Output:

```

{
  "stream": {
    "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
    "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
    "startTime": "2020-05-05T21:55:38Z",
    "state": "LIVE",
    "health": "HEALTHY",
    "streamId": "st-ABCDEFghij01234KLMN5678",
  }
}

```

```
    "viewerCount": 1
  }
}
```

For more information, see [Create a Channel](#) in the *IVS Low-Latency User Guide*.

- For API details, see [GetStream](#) in *AWS CLI Command Reference*.

import-playback-key-pair

The following code example shows how to use `import-playback-key-pair`.

AWS CLI

To import the public portion of a new key pair

The following `import-playback-key-pair` example imports the specified public key (specified as a string in PEM format) and returns the arn and fingerprint of the new key pair.

```
aws ivs import-playback-key-pair \
  --name "my-playback-key" \
  --public-key-material "G1lbnQxOTA3BgNVBAMMMFdoeSBhcmUgeW91IGR1..."
```

Output:

```
{
  "keyPair": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:playback-key/abcd1234efgh",
    "name": "my-playback-key",
    "fingerprint": "0a:1b:2c:ab:cd:ef:34:56:70:b1:b2:71:01:2a:a3:72",
    "tags": {}
  }
}
```

For more information, see [Setting Up Private Channels](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [ImportPlaybackKeyPair](#) in *AWS CLI Command Reference*.

list-channels

The following code example shows how to use `list-channels`.

AWS CLI

Example 1: To get summary information about all channels

The following `list-channels` example lists all channels for your AWS account.

```
aws ivs list-channels
```

Output:

```
{
  "channels": [
    {
      "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
      "name": "channel-1",
      "latencyMode": "LOW",
      "authorized": false,
      "insecureIngest": false,
      "preset": "",
      "playbackRestrictionPolicyArn": "",
      "recordingConfigurationArn": "arn:aws:ivs:us-
west-2:123456789012:recording-configuration/ABCD12cdEFgh",
      "tags": {},
      "type": "STANDARD"
    },
    {
      "arn": "arn:aws:ivs:us-west-2:123456789012:channel/efghEFGHijkl",
      "name": "channel-2",
      "latencyMode": "LOW",
      "authorized": false,
      "preset": "",
      "playbackRestrictionPolicyArn": "arn:aws:ivs:us-
west-2:123456789012:playback-restriction-policy/ABcdef34ghIJ",
      "recordingConfigurationArn": "",
      "tags": {},
      "type": "STANDARD"
    }
  ]
}
```

For more information, see [Create a Channel](#) in the *IVS Low-Latency User Guide*.

Example 2: To get summary information about all channels, filtered by the specified RecordingConfiguration ARN

The following `list-channels` example lists all channels for your AWS account, that are associated with the specified RecordingConfiguration ARN.

```
aws ivs list-channels \  
  --filter-by-recording-configuration-arn "arn:aws:ivs:us-  
west-2:123456789012:recording-configuration/ABCD12cdEFgh"
```

Output:

```
{  
  "channels": [  
    {  
      "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",  
      "name": "channel-1",  
      "latencyMode": "LOW",  
      "authorized": false,  
      "insecureIngest": false,  
      "preset": "",  
      "playbackRestrictionPolicyArn": "",  
      "recordingConfigurationArn": "arn:aws:ivs:us-  
west-2:123456789012:recording-configuration/ABCD12cdEFgh",  
      "tags": {},  
      "type": "STANDARD"  
    }  
  ]  
}
```

For more information, see [Record to Amazon S3](#) in the *IVS Low-Latency User Guide*.

Example 3: To get summary information about all channels, filtered by the specified PlaybackRestrictionPolicy ARN

The following `list-channels` example lists all channels for your AWS account, that are associated with the specified PlaybackRestrictionPolicy ARN.

```
aws ivs list-channels \  
  --filter-by-playback-restriction-policy-arn "arn:aws:ivs:us-  
west-2:123456789012:playback-restriction-policy/ABCdef34ghIJ"
```

Output:

```
{
  "channels": [
    {
      "arn": "arn:aws:ivs:us-west-2:123456789012:channel/efghEFGHijkl",
      "name": "channel-2",
      "latencyMode": "LOW",
      "authorized": false,
      "preset": "",
      "playbackRestrictionPolicyArn": "arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/ABCdef34ghIJ",
      "recordingConfigurationArn": "",
      "tags": {},
      "type": "STANDARD"
    }
  ]
}
```

For more information, see [Undesired Content and Viewers](#) in the *IVS Low-Latency User Guide*.

- For API details, see [ListChannels](#) in *AWS CLI Command Reference*.

list-playback-key-pairs

The following code example shows how to use `list-playback-key-pairs`.

AWS CLI**To get summary information about all playback key pairs**

The following `list-playback-key-pairs` example returns information about all key pairs.

```
aws ivs list-playback-key-pairs
```

Output:

```
{
  "keyPairs": [
    {
      "arn": "arn:aws:ivs:us-west-2:123456789012:playback-key/abcd1234efgh",
      "name": "test-key-0",
      "tags": {}
    }
  ]
}
```

```
    },
    {
      "arn": "arn:aws:ivs:us-west-2:123456789012:playback-key/ijkl5678mnop",
      "name": "test-key-1",
      "tags": {}
    }
  ]
}
```

For more information, see [Setting Up Private Channels](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [ListPlaybackKeyPairs](#) in *AWS CLI Command Reference*.

list-playback-restriction-policies

The following code example shows how to use `list-playback-restriction-policies`.

AWS CLI

To get summary information about all playback restriction policies

The following `list-playback-restriction-policies` example lists all playback restriction policies for your AWS account.

```
aws ivs list-playback-restriction-policies
```

Output:

```
{
  "playbackRestrictionPolicies": [
    {
      "arn": "arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/
ABcdef34ghIJ",
      "allowedCountries": [
        "US",
        "MX"
      ],
      "allowedOrigins": [
        "https://www.website1.com",
        "https://www.website2.com"
      ],
    }
  ]
}
```

```
        "enableStrictOriginEnforcement": true,
        "name": "test-playback-restriction-policy",
        "tags": {
            "key1": "value1",
            "key2": "value2"
        }
    }
]
```

For more information, see [Undesired Content and Viewers](#) in the *IVS Low-Latency User Guide*.

- For API details, see [ListPlaybackRestrictionPolicies](#) in *AWS CLI Command Reference*.

list-recording-configurations

The following code example shows how to use `list-recording-configurations`.

AWS CLI

To list all the RecordingConfiguration resources created in this account

The following `list-recording-configurations` example gets information about all RecordingConfiguration resources in your account.

```
aws ivs list-recording-configurations
```

Output:

```
{
  "recordingConfigurations": [
    {
      "arn": "arn:aws:ivs:us-west-2:123456789012:recording-configuration/
ABCdef34ghIJ",
      "name": "test-recording-config-1",
      "destinationConfiguration": {
        "s3": {
          "bucketName": "demo-recording-bucket-1"
        }
      },
      "state": "ACTIVE",
      "tags": {}
    },
  ],
}
```

```

    {
      "arn": "arn:aws:ivs:us-west-2:123456789012:recording-configuration/
CD12abcdGHIJ",
      "name": "test-recording-config-2",
      "destinationConfiguration": {
        "s3": {
          "bucketName": "demo-recording-bucket-2"
        }
      },
      "state": "ACTIVE",
      "tags": {}
    }
  ]
}

```

For more information, see [Record to Amazon S3](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [ListRecordingConfigurations](#) in *AWS CLI Command Reference*.

list-stream-keys

The following code example shows how to use `list-stream-keys`.

AWS CLI

To get a list of stream keys

The following `list-stream-keys` example lists all stream keys for a specified ARN (Amazon Resource Name).

```

aws ivs list-stream-keys \
  --channel-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh

```

Output:

```

{
  "streamKeys": [
    {
      "arn": "arn:aws:ivs:us-west-2:123456789012:stream-key/abcdABCDefgh",
      "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
      "tags": {}
    }
  ]
}

```

```

    }
  ]
}

```

For more information, see [Create a Channel](#) in the *IVS Low-Latency User Guide*.

- For API details, see [ListStreamKeys](#) in *AWS CLI Command Reference*.

list-stream-sessions

The following code example shows how to use `list-stream-sessions`.

AWS CLI

To get a summary of current and previous streams for a specified channel in the current AWS region

The following `list-stream-sessions` example reports summary information for streams for a specified channel ARN (Amazon Resource Name).

```

aws ivs list-stream-sessions \
  --channel-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh \
  --max-results 25 \
  --next-token ""

```

Output:

```

{
  "nextToken": "set-2",
  "streamSessions": [
    {
      "startTime": 1641578182,
      "endTime": 1641579982,
      "hasErrorEvent": false,
      "streamId": "mystream"
    }
    ...
  ]
}

```

For more information, see [Create a Channel](#) in the *IVS Low-Latency User Guide*.

- For API details, see [ListStreamSessions](#) in *AWS CLI Command Reference*.

list-streams

The following code example shows how to use `list-streams`.

AWS CLI

To get a list of live streams and their state

The following `list-streams` example lists all live streams for your AWS account.

```
aws ivs list-streams
```

Output:

```
{
  "streams": [
    {
      "channelArn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
      "state": "LIVE",
      "health": "HEALTHY",
      "streamId": "st-ABCDefghij01234KLMN5678",
      "viewerCount": 1
    }
  ]
}
```

For more information, see [Create a Channel](#) in the *IVS Low-Latency User Guide*.

- For API details, see [ListStreams](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list all tags for an AWS resource (for example: channel, stream key)

The following `list-tags-for-resource` example lists all tags for a specified resource ARN (Amazon Resource Name).

```
aws ivs list-tags-for-resource \
  --resource-arn arn:aws:ivs:us-west-2:12345689012:channel/abcdABCDefgh
```


Output:

```
{
  "tags":
  {
    "key1": "value1",
    "key2": "value2"
  }
}
```

For more information, see [Tagging](#) in the *Amazon Interactive Video Service API Reference*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

put-metadata

The following code example shows how to use `put-metadata`.

AWS CLI**To insert metadata into the active stream for a specified channel**

The following `put-metadata` example inserts the given metadata into the stream for the specified channel.

```
aws ivs put-metadata \
  --channel-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh \
  --metadata '{"my": "metadata"}'
```

This command produces no output.

For more information, see [Create a Channel](#) in the *IVS Low-Latency User Guide*.

- For API details, see [PutMetadata](#) in *AWS CLI Command Reference*.

start-viewer-session-revocation

The following code example shows how to use `start-viewer-session-revocation`.

AWS CLI**To revoke a viewer session for a given multiple channel-ARN and viewer-ID pair**

The following `start-viewer-session-revocation` example starts the process of revoking the viewer session associated with a specified channel ARN and viewer ID, up to and including the specified session version number. If the version is not provided, it defaults to 0.

```
aws ivs batch-start-viewer-session-revocation \  
  --channel-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh \  
  --viewer-id abcdefg \  
  --viewer-session-versions-less-than-or-equal-to 1234567890
```

This command produces no output.

For more information, see [Setting Up Private Channels](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [StartViewerSessionRevocation](#) in *AWS CLI Command Reference*.

stop-stream

The following code example shows how to use `stop-stream`.

AWS CLI

To stop a specified stream

The following `stop-stream` example stops the stream on the specified channel.

```
aws ivs stop-stream \  
  --channel-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh
```

This command produces no output.

For more information, see [Create a Channel](#) in the *IVS Low-Latency User Guide*.

- For API details, see [StopStream](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add or update tags for an AWS resource (for example: channel, stream key)

The following `tag-resource` example adds or updates tags for a specified resource ARN (Amazon Resource Name).

```
aws ivs tag-resource \  
  --resource-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh \  
  --tags "tagkey1=tagvalue1, tagkey2=tagvalue2"
```

This command produces no output.

For more information, see [Tagging](#) in the *Amazon Interactive Video Service API Reference*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags for an AWS resource (for example: channel, stream key)

The following `untag-resource` example removes the specified tags for a specified resource ARN (Amazon Resource Name).

```
aws ivs untag-resource \  
  --resource-arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh \  
  --tag-keys "tagkey1, tagkey2"
```

This command produces no output.

For more information, see [Tagging](#) in the *Amazon Interactive Video Service API Reference*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-channel

The following code example shows how to use `update-channel`.

AWS CLI

Example 1: To update a channel's configuration information

The following `update-channel` example updates the channel configuration for a specified channel ARN to change the channel name. This does not affect an ongoing stream of this channel; you must stop and restart the stream for the changes to take effect.

```
aws ivs update-channel \  
  --arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh \  
  --name "channel-1" \  
  --insecure-ingest
```

Output:

```
{  
  "channel": {  
    "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",  
    "name": "channel-1",  
    "latencyMode": "LOW",  
    "type": "STANDARD",  
    "playbackRestrictionPolicyArn": "",  
    "recordingConfigurationArn": "",  
    "srt": {  
      "endpoint": "a1b2c3d4e5f6.srt.live-video.net",  
      "passphrase":  
"AB1C2defGHijklMNop3PqQRstUvwxyzABCDefghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"  
    },  
    "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",  
    "insecureIngest": true,  
    "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/  
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",  
    "preset": "",  
    "authorized": false,  
    "tags": {}  
  }  
}
```

For more information, see [Create a Channel](#) in the *IVS Low-Latency User Guide*.

Example 2: To update a channel's configuration to enable recording

The following `update-channel` example updates the channel configuration for a specified channel ARN to enable recording. This does not affect an ongoing stream of this channel; you must stop and restart the stream for the changes to take effect.

```
aws ivs update-channel \  
  --arn arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh \  
  --name "channel-1" \  
  --insecure-ingest
```

```
--arn "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh" \
--no-insecure-ingest \
--recording-configuration-arn "arn:aws:ivs:us-west-2:123456789012:recording-
configuration/ABCD12cdEFgh"
```

Output:

```
{
  "channel": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
    "name": "test-channel-with-recording",
    "latencyMode": "LOW",
    "type": "STANDARD",
    "playbackRestrictionPolicyArn": "",
    "recordingConfigurationArn": "arn:aws:ivs:us-west-2:123456789012:recording-
configuration/ABCD12cdEFgh",
    "srt": {
      "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
      "passphrase":
"BA1C2defGHijkLMNo3PqQRstUvwxyzaBCDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
    },
    "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
    "insecureIngest": false,
    "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
    "preset": "",
    "authorized": false,
    "tags": {}
  }
}
```

For more information, see [Record to Amazon S3](#) in the *IVS Low-Latency User Guide*.

Example 3: To update a channel's configuration to disable recording

The following `update-channel` example updates the channel configuration for a specified channel ARN to disable recording. This does not affect an ongoing stream of this channel; you must stop and restart the stream for the changes to take effect.

```
aws ivs update-channel \
  --arn "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh" \
  --recording-configuration-arn ""
```

Output:

```
{
  "channel": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
    "name": "test-channel-with-recording",
    "latencyMode": "LOW",
    "type": "STANDARD",
    "playbackRestrictionPolicyArn": "",
    "recordingConfigurationArn": "",
    "srt": {
      "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
      "passphrase":
"AB1C2edfGHijklMN03PqQRstUvwxyzABCDEFghh4ijklMN5opqrStuVWXYZAbCDEfghIJ"
    },
    "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
    "insecureIngest": false,
    "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
    "preset": "",
    "authorized": false,
    "tags": {}
  }
}
```

For more information, see [Record to Amazon S3](#) in the *IVS Low-Latency User Guide*.

Example 4: To update a channel's configuration to enable playback restriction

The following `update-channel` example updates the channel configuration for a specified channel ARN to apply a playback restriction policy. This does not affect an ongoing stream of this channel; you must stop and restart the stream for the changes to take effect.

```
aws ivs update-channel \
  --arn "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh" \
  --no-insecure-ingest \
  --playback-restriction-policy-arn "arn:aws:ivs:us-west-2:123456789012:playback-
restriction-policy/ABcdef34ghIJ"
```

Output:

```
{
```

```

"channel": {
  "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
  "name": "test-channel-with-playback-restriction-policy",
  "latencyMode": "LOW",
  "type": "STANDARD",
  "playbackRestrictionPolicyArn": "arn:aws:ivs:us-
west-2:123456789012:playback-restriction-policy/ABCdef34ghIJ",
  "recordingConfigurationArn": "",
  "srt": {
    "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
    "passphrase":
"AB1C2defGHijklMN03PqQRstUvwxyzaCBDEfghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
  },
  "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
  "insecureIngest": false,
  "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
  "preset": "",
  "authorized": false,
  "tags": {}
}
}

```

For more information, see [Undesired Content and Viewers](#) in the *IVS Low-Latency User Guide*.

Example 5: To update a channel's configuration to disable playback restriction

The following `update-channel` example updates the channel configuration for a specified channel ARN to disable playback restriction. This does not affect an ongoing stream of this channel; you must stop and restart the stream for the changes to take effect.

```

aws ivs update-channel \
  --arn "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh" \
  --playback-restriction-policy-arn ""

```

Output:

```

{
  "channel": {
    "arn": "arn:aws:ivs:us-west-2:123456789012:channel/abcdABCDefgh",
    "name": "test-channel-with-playback-restriction-policy",
    "latencyMode": "LOW",
    "type": "STANDARD",

```

```

    "playbackRestrictionPolicyArn": "",
    "recordingConfigurationArn": "",
    "srt": {
        "endpoint": "a1b2c3d4e5f6.srt.live-video.net",
        "passphrase":
"AB1C2defGHijklMNop3PqQRstUvwxyzABCDeFghh4ijklMN5opqrStuVWxyzAbCDEfghIJ"
    },
    "ingestEndpoint": "a1b2c3d4e5f6.global-contribute.live-video.net",
    "insecureIngest": false,
    "playbackUrl": "https://a1b2c3d4e5f6.us-west-2.playback.live-video.net/api/
video/v1/us-west-2.123456789012.channel.abcdEFGH.m3u8",
    "preset": "",
    "authorized": false,
    "tags": {}
}
}

```

For more information, see [Undesired Content and Viewers](#) in the *IVS Low-Latency User Guide*.

- For API details, see [UpdateChannel](#) in *AWS CLI Command Reference*.

update-playback-restriction-policy

The following code example shows how to use `update-playback-restriction-policy`.

AWS CLI

To update a playback restriction policy

The following `update-playback-restriction-policy` example updates the playback restriction policy with the specified policy ARN to disable strict origin enforcement. This does not affect an ongoing stream of the associated channel; you must stop and restart the stream for the changes to take effect.

```

aws ivs update-playback-restriction-policy \
  --arn "arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/
ABCdef34ghIJ" \
  --no-enable-strict-origin-enforcement

```

Output:

```
{
```



```
"playbackRestrictionPolicy": {
  "arn": "arn:aws:ivs:us-west-2:123456789012:playback-restriction-policy/
ABcdef34ghIJ",
  "allowedCountries": [
    "US",
    "MX"
  ],
  "allowedOrigins": [
    "https://www.website1.com",
    "https://www.website2.com"
  ],
  "enableStrictOriginEnforcement": false,
  "name": "test-playback-restriction-policy",
  "tags": {
    "key1": "value1",
    "key2": "value2"
  }
}
```

For more information, see [Undesired Content and Viewers](#) in the *IVS Low-Latency User Guide*.

- For API details, see [UpdatePlaybackRestrictionPolicy](#) in *AWS CLI Command Reference*.

Amazon IVS Chat examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon IVS Chat.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-chat-token

The following code example shows how to use `create-chat-token`.

AWS CLI

To create a chat token

The following `create-chat-token` example creates an encrypted chat token that is used to establish an individual WebSocket connection to a room. The token is valid for one minute, and a connection (session) established with the token is valid for the specified duration.

```
aws ivschat create-chat-token \
  --roomIdIdentifier "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6", \
  --userId "11231234" \
  --capabilities "SEND_MESSAGE", \
  --sessionDurationInMinutes 30
```

Output:

```
{
  "token": "ACEGmnoq#1rstu2...BDFH3vxwy!4hlm!#5",
  "sessionExpirationTime": "2022-03-16T04:44:09+00:00"
  "state": "CREATING",
  "tokenExpirationTime": "2022-03-16T03:45:09+00:00"
}
```

For more information, see [Step 3: Authenticate and Authorize Chat Clients](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [CreateChatToken](#) in *AWS CLI Command Reference*.

create-logging-configuration

The following code example shows how to use `create-logging-configuration`.

AWS CLI

To create a chat LoggingConfiguration resource

The following `create-logging-configuration` example creates a `LoggingConfiguration` resource that allows clients to store and record sent messages.

```
aws ivschat create-logging-configuration \  
  --destination-configuration s3={bucketName=demo-logging-bucket} \  
  --name "test-logging-config" \  
  --tags "key1=value1, key2=value2"
```

Output:

```
{  
  "arn": "arn:aws:ivschat:us-west-2:123456789012:logging-configuration/  
ABcdef34ghIJ",  
  "createTime": "2022-09-14T17:48:00.653000+00:00",  
  "destinationConfiguration": {  
    "s3": {  
      "bucketName": "demo-logging-bucket"  
    }  
  },  
  "id": "ABcdef34ghIJ",  
  "name": "test-logging-config",  
  "state": "ACTIVE",  
  "tags": { "key1" : "value1", "key2" : "value2" },  
  "updateTime": "2022-09-14T17:48:01.104000+00:00"  
}
```

For more information, see [Getting Started with Amazon IVS Chat](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [CreateLoggingConfiguration](#) in *AWS CLI Command Reference*.

create-room

The following code example shows how to use `create-room`.

AWS CLI

To create a room

The following `create-room` example creates a new room.

```
aws ivschat create-room \  
  --name "test-room" \  
  --tags "key1=value1, key2=value2"
```

```
--name "test-room-1" \  
--logging-configuration-identifiers "arn:aws:ivschat:us-  
west-2:123456789012:logging-configuration/ABCdef34ghIJ" \  
--maximum-message-length 256 \  
--maximum-message-rate-per-second 5
```

Output:

```
{  
  "arn": "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6",  
  "id": "g1H2I3j4k5L6",  
  "createTime": "2022-03-16T04:44:09+00:00",  
  "loggingConfigurationIdentifiers": ["arn:aws:ivschat:us-  
west-2:123456789012:logging-configuration/ABCdef34ghIJ"],  
  "maximumMessageLength": 256,  
  "maximumMessageRatePerSecond": 5,  
  "name": "test-room-1",  
  "tags": {}  
  "updateTime": "2022-03-16T07:22:09+00:00"  
}
```

For more information, see [Step 2: Create a Chat Room](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [CreateRoom](#) in *AWS CLI Command Reference*.

delete-logging-configuration

The following code example shows how to use delete-logging-configuration.

AWS CLI

To delete a chat LoggingConfiguration resource

The following delete-logging-configuration example deletes the LoggingConfiguration resource for the specified ARN.

```
aws ivschat delete-logging-configuration \  
  --identifier "arn:aws:ivschat:us-west-2:123456789012:logging-configuration/  
ABCdef34ghIJ"
```

This command produces no output.

For more information, see [Getting Started with Amazon IVS Chat](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [DeleteLoggingConfiguration](#) in *AWS CLI Command Reference*.

delete-message

The following code example shows how to use `delete-message`.

AWS CLI

To delete messages from a specified room

The following `delete-message` example sends an event to the specified room, which directs clients to delete the specified message: that is, unrender it from view and delete it from the client's chat history.

```
aws ivschat delete-message \  
  --roomIdentifier "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6" \  
  --id "ABC123def456" \  
  --reason "Message contains profanity"
```

Output:

```
{  
  "id": "12345689012"  
}
```

For more information, see [Getting Started with Amazon IVS Chat](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [DeleteMessage](#) in *AWS CLI Command Reference*.

delete-room

The following code example shows how to use `delete-room`.

AWS CLI

To delete a room

The following `delete-room` example deletes the specified room. Connected clients are disconnected. On success it returns HTTP 204 with an empty response body.

```
aws ivschat delete-room \  
  --identifier "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6"
```

This command produces no output.

For more information, see [Getting Started with Amazon IVS Chat](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [DeleteRoom](#) in *AWS CLI Command Reference*.

disconnect-user

The following code example shows how to use `disconnect-user`.

AWS CLI

To disconnect a user from a room

The following `disconnect-user` example disconnects all connections for the specified user from the specified room. On success it returns HTTP 200 with an empty response body.

```
aws ivschat disconnect-user \  
  --roomId "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6" \  
  --userId "ABC123def456" \  
  --reason "Violated terms of service"
```

This command produces no output.

For more information, see [Getting Started with Amazon IVS Chat](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [DisconnectUser](#) in *AWS CLI Command Reference*.

get-logging-configuration

The following code example shows how to use `get-logging-configuration`.

AWS CLI

To get information about a LoggingConfiguration resource

The following `get-logging-configuration` example gets information about the LoggingConfiguration resource for the specified ARN.

```
aws ivschat get-logging-configuration \  
  --identifier "arn:aws:ivschat:us-west-2:123456789012:logging-configuration/  
ABcdef34ghIJ"
```

Output:

```
{  
  "arn": "arn:aws:ivschat:us-west-2:123456789012:logging-configuration/  
ABcdef34ghIJ",  
  "createTime": "2022-09-14T17:48:00.653000+00:00",  
  "destinationConfiguration": {  
    "s3": {  
      "bucketName": "demo-logging-bucket"  
    }  
  },  
  "id": "ABcdef34ghIJ",  
  "name": "test-logging-config",  
  "state": "ACTIVE",  
  "tags": { "key1" : "value1", "key2" : "value2" },  
  "updateTime": "2022-09-14T17:48:01.104000+00:00"  
}
```

For more information, see [Getting Started with Amazon IVS Chat](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [GetLoggingConfiguration](#) in *AWS CLI Command Reference*.

get-room

The following code example shows how to use `get-room`.

AWS CLI

To get the specified room

The following `get-room` example gets information about the specified room.

```
aws ivschat get-room \  
  --identifier "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6"
```

Output:

```
{  
  "arn": "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6",  
  "createTime": "2022-03-16T04:44:09+00:00",  
  "id": "g1H2I3j4k5L6",  
  "loggingConfigurationIdentifiers": ["arn:aws:ivschat:us-  
west-2:123456789012:logging-configuration/ABCdef34ghIJ"],  
  "maximumMessageLength": 256,  
  "maximumMessageRatePerSecond": 5,  
  "name": "test-room-1",  
  "tags": {},  
  "updateTime": "2022-03-16T07:22:09+00:00"  
}
```

For more information, see [Getting Started with Amazon IVS Chat](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [GetRoom](#) in *AWS CLI Command Reference*.

list-logging-configurations

The following code example shows how to use `list-logging-configurations`.

AWS CLI

To get summary information about all logging configurations for the user in the AWS region where the API request is processed

The following `list-logging-configurations` example lists information about all `LoggingConfiguration` resources for the user in the AWS region where the API request is processed.

```
aws ivschat list-logging-configurations \  
  --max-results 2 \  
  --next-token ""
```

Output:


```
{
  "nextToken": "set-2",
  "loggingConfigurations": [
    {
      "arn": "arn:aws:ivschat:us-west-2:123456789012:logging-configuration/
ABcdef34ghIJ",
      "createTime": "2022-09-14T17:48:00.653000+00:00",
      "destinationConfiguration": {
        "s3": {
          "bucketName": "demo-logging-bucket"
        }
      },
      "id": "ABcdef34ghIJ",
      "name": "test-logging-config",
      "state": "ACTIVE",
      "tags": { "key1" : "value1", "key2" : "value2" },
      "updateTime": "2022-09-14T17:48:01.104000+00:00"
    }
    ...
  ]
}
```

For more information, see [Getting Started with Amazon IVS Chat](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [ListLoggingConfigurations](#) in *AWS CLI Command Reference*.

list-rooms

The following code example shows how to use `list-rooms`.

AWS CLI

To get summary information about all your rooms in the current region

The following `list-rooms` example gets summary information about all the rooms in the AWS region where the request is processed. Results are sorted in descending order of `updateTime`.

```
aws ivschat list-rooms \
  --logging-configuration-identifier "arn:aws:ivschat:us-
west-2:123456789012:logging-configuration/ABcdef34ghIJ" \
  --max-results 10 \
```

```
--next-token ""
```

Output:

```
{
  "nextToken": "page3",
  "rooms": [
    {
      "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6",
      "createTime": "2022-03-16T04:44:09+00:00",
      "id": "g1H2I3j4k5L6",
      "loggingConfigurationIdentifiers": ["arn:aws:ivschat:us-
west-2:123456789012:logging-configuration/ABCdef34ghIJ"],
      "name": "test-room-1",
      "tags": {},
      "updateTime": "2022-03-16T07:22:09+00:00"
    }
  ]
}
```

For more information, see [Getting Started with Amazon IVS Chat](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [ListRooms](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI**To list all tags for an AWS resource (for example: Room)**

The following `list-tags-for-resource` example lists all tags for a specified resource ARN (Amazon Resource Name).

```
aws ivschat list-tags-for-resource \
  --resource-arn arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6
```

Output:

```
{
```

```
"tags":
{
  "key1": "value1",
  "key2": "value2"
}
}
```

For more information, see [Tagging](#) in the *Amazon Interactive Video Service API Reference*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

send-event

The following code example shows how to use send-event.

AWS CLI

To send an event to a room

The following send-event example sends the given event to the specified room.

```
aws ivschat send-event \
  --roomIdIdentifier "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6" \
  --eventName "SystemMessage" \
  --attributes \
    "msgType"="user-notification", \
    "msgText"="This chat room will close in 15 minutes."
```

Output:

```
{
  "id": "12345689012"
}
```

For more information, see [Getting Started with Amazon IVS Chat](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [SendEvent](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To add or update tags for an AWS resource (for example: Room)

The following `tag-resource` example adds or updates tags for a specified resource ARN (Amazon Resource Name). On success it returns HTTP 200 with an empty response body.

```
aws ivschat tag-resource \  
  --resource-arn arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6 \  
  --tags "tagkey1=tagkeyvalue1, tagkey2=tagkeyvalue2"
```

This command produces no output.

For more information, see [Tagging](#) in the *Amazon Interactive Video Service API Reference*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags for an AWS resource (for example: Room)

The following `untag-resource` example removes the specified tags for a specified resource ARN (Amazon Resource Name). On success it returns HTTP 200 with an empty response body.

```
aws ivschat untag-resource \  
  --resource-arn arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6 \  
  --tag-keys "tagkey1, tagkey2"
```

This command produces no output.

For more information, see [Tagging](#) in the *Amazon Interactive Video Service API Reference*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-logging-configuration

The following code example shows how to use `update-logging-configuration`.

AWS CLI

To update a room's logging configuration

The following `update-logging-configuration` example updates a `LoggingConfiguration` resource with the given data.

```
aws ivschat update-logging-configuration \  
  --destination-configuration s3={bucketName=demo-logging-bucket} \  
  --identifier "arn:aws:ivschat:us-west-2:123456789012:logging-configuration/  
ABcdef34ghIJ" \  
  --name "test-logging-config"
```

Output:

```
{  
  "arn": "arn:aws:ivschat:us-west-2:123456789012:logging-configuration/  
ABcdef34ghIJ",  
  "createTime": "2022-09-14T17:48:00.653000+00:00",  
  "destinationConfiguration": {  
    "s3": {  
      "bucketName": "demo-logging-bucket"  
    }  
  },  
  "id": "ABcdef34ghIJ",  
  "name": "test-logging-config",  
  "state": "ACTIVE",  
  "tags": { "key1" : "value1", "key2" : "value2" },  
  "updateTime": "2022-09-14T17:48:01.104000+00:00"  
}
```

For more information, see [Getting Started with Amazon IVS Chat](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [UpdateLoggingConfiguration](#) in *AWS CLI Command Reference*.

update-room

The following code example shows how to use `update-room`.

AWS CLI

To update a room's configuration

The following `update-room` example updates the specified room's configuration with the given data.

```
aws ivschat update-room \  
  --identifier "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6" \  
  --logging-configuration-identifiers "arn:aws:ivschat:us-  
west-2:123456789012:logging-configuration/ABCdef34ghIJ" \  
  --name "chat-room-a" \  
  --maximum-message-length 256 \  
  --maximum-message-rate-per-second 5
```

Output:

```
{  
  "arn": "arn:aws:ivschat:us-west-2:12345689012:room/g1H2I3j4k5L6",  
  "createTime": "2022-03-16T04:44:09+00:00",  
  "id": "g1H2I3j4k5L6",  
  "loggingConfigurationIdentifiers": ["arn:aws:ivschat:us-  
west-2:123456789012:logging-configuration/ABCdef34ghIJ"],  
  "maximumMessageLength": 256,  
  "maximumMessageRatePerSecond": 5,  
  "name": "chat-room-a",  
  "tags": {},  
  "updateTime": "2022-03-16T07:22:09+00:00"  
}
```

For more information, see [Getting Started with Amazon IVS Chat](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [UpdateRoom](#) in *AWS CLI Command Reference*.

Amazon IVS Real-Time Streaming examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon IVS Real-Time Streaming.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-encoder-configuration

The following code example shows how to use create-encoder-configuration.

AWS CLI

To create a composition encoder configuration

The following create-encoder-configuration example creates a composition encoder configuration with the specified properties.

```
aws ivs-realtime create-encoder-configuration \  
  --name test-ec --video bitrate=3500000,framerate=30.0,height=1080,width=1920
```

Output:

```
{  
  "encoderConfiguration": {  
    "arn": "arn:aws:ivs:ap-northeast-1:123456789012:encoder-configuration/  
ABabCDcdEFef",  
    "name": "test-ec",  
    "tags": {},  
    "video": {  
      "bitrate": 3500000,  
      "framerate": 30,  
      "height": 1080,  
      "width": 1920  
    }  
  }  
}
```

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [CreateEncoderConfiguration](#) in *AWS CLI Command Reference*.

create-participant-token

The following code example shows how to use create-participant-token.

AWS CLI

To create a stage participant token

The following create-participant-token example creates a participant token for the specified stage.

```
aws ivs-realtime create-participant-token \
  --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
  --user-id bob
```

Output:

```
{
  "participantToken": {
    "expirationTime": "2023-03-07T09:47:43+00:00",
    "participantId": "ABCDEFghij01234KLMN6789",
    "token": "abcd1234defg5678"
  }
}
```

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [CreateParticipantToken](#) in *AWS CLI Command Reference*.

create-stage

The following code example shows how to use create-stage.

AWS CLI

To create a stage

The following create-stage example creates a stage and stage participant token for a specified user.


```
aws ivs-realtime create-stage \  
  --name stage1 \  
  --participant-token-configurations userId=alice
```

Output:

```
{  
  "participantTokens": [  
    {  
      "participantId": "ABCDEFghij01234KLMN5678",  
      "token": "a1b2c3d4567890ab",  
      "userId": "alice"  
    }  
  ],  
  "stage": {  
    "activeSessionId": "st-a1b2c3d4e5f6g",  
    "arn": "arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh",  
    "name": "stage1",  
    "tags": {}  
  }  
}
```

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [CreateStage](#) in *AWS CLI Command Reference*.

create-storage-configuration

The following code example shows how to use `create-storage-configuration`.

AWS CLI**To create a composition storage configuration**

The following `create-storage-configuration` example creates a composition storage configuration with the specified properties.

```
aws ivs-realtime create-storage-configuration \  
  --name "test-sc" --s3 "bucketName=test-bucket-name"
```

Output:

```
{
  "storageConfiguration": {
    "arn": "arn:aws:ivs:ap-northeast-1:123456789012:storage-configuration/
ABabCDcdEFef",
    "name": "test-sc",
    "s3": {
      "bucketName": "test-bucket-name"
    },
    "tags": {}
  }
}
```

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [CreateStorageConfiguration](#) in *AWS CLI Command Reference*.

delete-encoder-configuration

The following code example shows how to use `delete-encoder-configuration`.

AWS CLI

To delete a composition encoder configuration

The following `delete-encoder-configuration` deletes the composition encoder configuration specified by the given ARN (Amazon Resource Name).

```
aws ivs-realtime delete-encoder-configuration \
  --arn "arn:aws:ivs:ap-northeast-1:123456789012:encoder-configuration/
ABabCDcdEFef"
```

This command produces no output.

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [DeleteEncoderConfiguration](#) in *AWS CLI Command Reference*.

delete-stage

The following code example shows how to use `delete-stage`.

AWS CLI

To delete a stage

The following `delete-stage` example deletes the specified stage.

```
aws ivs-realtime delete-stage \  
  --arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh
```

This command produces no output.

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [DeleteStage](#) in *AWS CLI Command Reference*.

`delete-storage-configuration`

The following code example shows how to use `delete-storage-configuration`.

AWS CLI

To delete a composition storage configuration

The following `delete-storage-configuration` deletes the composition storage configuration specified by the given ARN (Amazon Resource Name).

```
aws ivs-realtime delete-storage-configuration \  
  --arn "arn:aws:ivs:ap-northeast-1:123456789012:storage-configuration/  
  ABabCDcdEFef"
```

This command produces no output.

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [DeleteStorageConfiguration](#) in *AWS CLI Command Reference*.

`disconnect-participant`

The following code example shows how to use `disconnect-participant`.

AWS CLI

To disconnect a stage participant

The following `disconnect-participant` example disconnects the specified participant from the specified stage.

```
aws ivs-realtime disconnect-participant \  
  --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \  
  --participant-id ABCDEfghij01234KLMN5678
```

This command produces no output.

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [DisconnectParticipant](#) in *AWS CLI Command Reference*.

get-composition

The following code example shows how to use `get-composition`.

AWS CLI

Example 1: To get a composition with default layout settings

The following `get-composition` example gets the composition for the ARN (Amazon Resource Name) specified.

```
aws ivs-realtime get-composition \  
  --arn "arn:aws:ivs:ap-northeast-1:123456789012:composition/abcdABCDefgh"
```

Output:

```
{  
  "composition": {  
    "arn": "arn:aws:ivs:ap-northeast-1:123456789012:composition/abcdABCDefgh",  
    "destinations": [  
      {  
        "configuration": {  
          "channel": {  
            "channelArn": "arn:aws:ivs:ap-northeast-1:123456789012:channel/abcABCdefDEg",
```

```

        "encoderConfigurationArn": "arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
    },
    "name": ""
},
"id": "AabBCcdDEefF",
"startTime": "2023-10-16T23:26:00+00:00",
"state": "ACTIVE"
},
{
    "configuration": {
        "name": "",
        "s3": {
            "encoderConfigurationArns": [
                "arn:aws:ivs:arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
            ],
            "recordingConfiguration": {
                "format": "HLS"
            },
            "storageConfigurationArn": "arn:arn:aws:ivs:ap-
northeast-1:123456789012:storage-configuration/FefABabCDcdE"
        }
    },
    "detail": {
        "s3": {
            "recordingPrefix": "aBcDeFgHhGfE/AbCdEfGhHgFe/GHFabcgefABC/
composite"
        }
    },
    "id": "GHFabcgefABC",
    "startTime": "2023-10-16T23:26:00+00:00",
    "state": "STARTING"
}
],
"layout": {
    "grid": {
        "featuredParticipantAttribute": ""
        "gridGap": 2,
        "omitStoppedVideo": false,
        "videoAspectRatio": "VIDEO",
        "videoFillMode": ""
    }
},
"stageArn": "arn:aws:ivs:ap-northeast-1:123456789012:stage/defgABCDabcd",

```

```
    "startTime": "2023-10-16T23:24:00+00:00",
    "state": "ACTIVE",
    "tags": {}
  }
}
```

For more information, see [Composite Recording \(Real-Time Streaming\)](#) in the *Amazon Interactive Video Service User Guide*.

Example 2: To get a composition with PiP layout

The following get-composition example gets the composition for the ARN (Amazon Resource Name) specified, which uses PiP layout.

```
aws ivs-realtime get-composition \
  --arn "arn:aws:ivs:ap-northeast-1:123456789012:composition/wxyzWXYZpqrs"
```

Output:

```
{
  "composition": {
    "arn": "arn:aws:ivs:ap-northeast-1:123456789012:composition/wxyzWXYZpqrs",
    "destinations": [
      {
        "configuration": {
          "channel": {
            "channelArn": "arn:aws:ivs:ap-northeast-1:123456789012:channel/abcABCdefDEg",
            "encoderConfigurationArn": "arn:aws:ivs:ap-northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
          },
          "name": ""
        },
        "id": "AabBCcdDEefF",
        "startTime": "2023-10-16T23:26:00+00:00",
        "state": "ACTIVE"
      },
      {
        "configuration": {
          "name": "",
          "s3": {
            "encoderConfigurationArns": [
```

```

        "arn:aws:ivs:arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
    ],
    "recordingConfiguration": {
        "format": "HLS"
    },
    "storageConfigurationArn": "arn:arn:aws:ivs:ap-
northeast-1:123456789012:storage-configuration/FefABabCDcdE"
    }
},
"detail": {
    "s3": {
        "recordingPrefix": "aBcDeFgHhGfE/AbCdEfGhHgFe/GHFabcgefABC/
composite"
    }
},
"id": "GHFabcgefABC",
"startTime": "2023-10-16T23:26:00+00:00",
"state": "STARTING"
}
],
"layout": {
    "pip": {
        "featuredParticipantAttribute": "abcdefg",
        "gridGap": 0,
        "omitStoppedVideo": false,
        "pipBehavior": "STATIC",
        "pipOffset": 0,
        "pipParticipantAttribute": "",
        "pipPosition": "BOTTOM_RIGHT",
        "videoFillMode": "COVER"
    }
},
"stageArn": "arn:aws:ivs:ap-northeast-1:123456789012:stage/defgABCDabcd",
"startTime": "2023-10-16T23:24:00+00:00",
"state": "ACTIVE",
"tags": {}
}
}

```

For more information, see [Composite Recording \(Real-Time Streaming\)](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [GetComposition](#) in *AWS CLI Command Reference*.

get-encoder-configuration

The following code example shows how to use `get-encoder-configuration`.

AWS CLI

To get a composition encoder configuration

The following `get-encoder-configuration` example gets the composition encoder configuration specified by the given ARN (Amazon Resource Name).

```
aws ivs-realtime get-encoder-configuration \  
  --arn "arn:aws:ivs:ap-northeast-1:123456789012:encoder-configuration/  
abcdABCDefgh"
```

Output:

```
{  
  "encoderConfiguration": {  
    "arn": "arn:aws:ivs:ap-northeast-1:123456789012:encoder-configuration/  
abcdABCDefgh",  
    "name": "test-ec",  
    "tags": {},  
    "video": {  
      "bitrate": 3500000,  
      "framerate": 30,  
      "height": 1080,  
      "width": 1920  
    }  
  }  
}
```

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [GetEncoderConfiguration](#) in *AWS CLI Command Reference*.

get-participant

The following code example shows how to use `get-participant`.

AWS CLI

To get a stage participant

The following `get-participant` example gets the stage participant for a specified participant ID and session ID in the specified stage ARN (Amazon Resource Name).

```
aws ivs-realtime get-participant \  
  --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \  
  --session-id st-a1b2c3d4e5f6g \  
  --participant-id abCDEf12GHIj
```

Output:

```
{  
  "participant": {  
    "browserName", "Google Chrome",  
    "browserVersion", "116",  
    "firstJoinTime": "2023-04-26T20:30:34+00:00",  
    "ispName", "Comcast",  
    "osName", "Microsoft Windows 10 Pro",  
    "osVersion", "10.0.19044",  
    "participantId": "abCDEf12GHIj",  
    "published": true,  
    "sdkVersion", "",  
    "state": "DISCONNECTED",  
    "userId": ""  
  }  
}
```

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [GetParticipant](#) in *AWS CLI Command Reference*.

get-stage-session

The following code example shows how to use `get-stage-session`.

AWS CLI

To get a stage session

The following `get-stage-session` example gets the stage session for a specified session ID of a specified stage ARN (Amazon Resource Name).

```
aws ivs-realtime get-stage-session \  
  --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \  
  --session-id st-a1b2c3d4e5f6g
```

Output:

```
{  
  "stageSession": {  
    "endTime": "2023-04-26T20:36:29+00:00",  
    "sessionId": "st-a1b2c3d4e5f6g",  
    "startTime": "2023-04-26T20:30:29.602000+00:00"  
  }  
}
```

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [GetStageSession](#) in *AWS CLI Command Reference*.

get-stage

The following code example shows how to use `get-stage`.

AWS CLI

To get a stage's configuration information

The following `get-stage` example gets the stage configuration for a specified stage ARN (Amazon Resource Name).

```
aws ivs-realtime get-stage \  
  --arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh
```

Output:

```
{  
  "stage": {  
    "activeSessionId": "st-a1b2c3d4e5f6g",  
    "arn": "arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh",
```

```
    "name": "test",
    "tags": {}
  }
}
```

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [GetStage](#) in *AWS CLI Command Reference*.

get-storage-configuration

The following code example shows how to use `get-storage-configuration`.

AWS CLI

To get a composition storage configuration

The following `get-storage-configuration` example gets the composition storage configuration specified by the given ARN (Amazon Resource Name).

```
aws ivs-realtime get-storage-configuration \
  --name arn "arn:aws:ivs:ap-northeast-1:123456789012:storage-configuration/
abcdABCDefgh"
```

Output:

```
{
  "storageConfiguration": {
    "arn": "arn:aws:ivs:ap-northeast-1:123456789012:storage-configuration/
abcdABCDefgh",
    "name": "test-sc",
    "s3": {
      "bucketName": "test-bucket-name"
    },
    "tags": {}
  }
}
```

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [GetStorageConfiguration](#) in *AWS CLI Command Reference*.

list-compositions

The following code example shows how to use `list-compositions`.

AWS CLI

To get a list of compositions

The following `list-compositions` lists all compositions for your AWS account, in the AWS region where the API request is processed.

```
aws ivs-realtime list-compositions
```

Output:

```
{
  "compositions": [
    {
      "arn": "arn:aws:ivs:ap-northeast-1:123456789012:composition/
abcdABCDefgh",
      "destinations": [
        {
          "id": "AabBCcdDEefF",
          "startTime": "2023-10-16T23:25:23+00:00",
          "state": "ACTIVE"
        }
      ],
      "stageArn": "arn:aws:ivs:ap-northeast-1:123456789012:stage/
defgABCDabcd",
      "startTime": "2023-10-16T23:25:21+00:00",
      "state": "ACTIVE",
      "tags": {}
    },
    {
      "arn": "arn:aws:ivs:ap-northeast-1:123456789012:composition/
ABcdabCDefgh",
      "destinations": [
        {
          "endTime": "2023-10-16T23:25:00.786512+00:00",
          "id": "aABbcCDdeEFf",
          "startTime": "2023-10-16T23:24:01+00:00",
          "state": "STOPPED"
        }
      ],
    }
  ]
}
```

```

        {
            "endTime": "2023-10-16T23:25:00.786512+00:00",
            "id": "deEFfaABbcCD",
            "startTime": "2023-10-16T23:24:01+00:00",
            "state": "STOPPED"
        }
    ],
    "endTime": "2023-10-16T23:25:00+00:00",
    "stageArn": "arn:aws:ivs:ap-northeast-1:123456789012:stage/
efghabcdABCD",
    "startTime": "2023-10-16T23:24:00+00:00",
    "state": "STOPPED",
    "tags": {}
}
]
}

```

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [ListCompositions](#) in *AWS CLI Command Reference*.

list-encoder-configurations

The following code example shows how to use `list-encoder-configurations`.

AWS CLI

To list composition encoder configurations

The following `list-encoder-configurations` lists all composition encoder configurations for your AWS account, in the AWS region where the API request is processed.

```
aws ivs-realtime list-encoder-configurations
```

Output:

```

{
  "encoderConfigurations": [
    {
      "arn": "arn:aws:ivs:ap-northeast-1:123456789012:encoder-configuration/
abcdABCDefgh",
      "name": "test-ec-1",

```

```

        "tags": {}
    },
    {
        "arn": "arn:aws:ivs:ap-northeast-1:123456789012:encoder-configuration/
ABCefgEFGabc",
        "name": "test-ec-2",
        "tags": {}
    }
]
}

```

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [ListEncoderConfigurations](#) in *AWS CLI Command Reference*.

list-participant-events

The following code example shows how to use `list-participant-events`.

AWS CLI

To get a list of stage participant events

The following `list-participant-events` example lists all participant events for a specified participant ID and session ID of a specified stage ARN (Amazon Resource Name).

```

aws ivs-realtime list-participant-events \
  --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
  --session-id st-a1b2c3d4e5f6g \
  --participant-id abCDEf12GHIj

```

Output:

```

{
  "events": [
    {
      "eventTime": "2023-04-26T20:36:28+00:00",
      "name": "LEFT",
      "participantId": "abCDEf12GHIj"
    },
    {
      "eventTime": "2023-04-26T20:36:28+00:00",

```

```

        "name": "PUBLISH_STOPPED",
        "participantId": "abCDEf12GHIj"
    },
    {
        "eventTime": "2023-04-26T20:30:34+00:00",
        "name": "JOINED",
        "participantId": "abCDEf12GHIj"
    },
    {
        "eventTime": "2023-04-26T20:30:34+00:00",
        "name": "PUBLISH_STARTED",
        "participantId": "abCDEf12GHIj"
    }
]
}

```

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [ListParticipantEvents](#) in *AWS CLI Command Reference*.

list-participants

The following code example shows how to use `list-participants`.

AWS CLI

To get a list of stage participants

The following `list-participants` example lists all participants for a specified session ID of a specified stage ARN (Amazon Resource Name).

```

aws ivs-realtime list-participants \
  --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \
  --session-id st-a1b2c3d4e5f6g

```

Output:

```

{
  "participants": [
    {
      "firstJoinTime": "2023-04-26T20:30:34+00:00",
      "participantId": "abCDEf12GHIj"
    }
  ]
}

```

```
        "published": true,  
        "state": "DISCONNECTED",  
        "userId": ""  
    }  
]  
}
```

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [ListParticipants](#) in *AWS CLI Command Reference*.

list-stage-sessions

The following code example shows how to use `list-stage-sessions`.

AWS CLI

To get a list of stage sessions

The following `list-stage-sessions` example lists all sessions for a specified stage ARN (Amazon Resource Name).

```
aws ivs-realtime list-stage-sessions \  
  --stage-arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh
```

Output:

```
{  
  "stageSessions": [  
    {  
      "endTime": "2023-04-26T20:36:29+00:00",  
      "sessionId": "st-a1b2c3d4e5f6g",  
      "startTime": "2023-04-26T20:30:29.602000+00:00"  
    }  
  ]  
}
```

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [ListStageSessions](#) in *AWS CLI Command Reference*.

list-stages

The following code example shows how to use `list-stages`.

AWS CLI

To get summary information about all stages

The following `list-stages` example lists all stages for your AWS account, in the AWS region where the API request is processed.

```
aws ivs-realtime list-stages
```

Output:

```
{
  "stages": [
    {
      "activeSessionId": "st-a1b2c3d4e5f6g",
      "arn": "arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh",
      "name": "stage1",
      "tags": {}
    },
    {
      "activeSessionId": "st-a123bcd456efg",
      "arn": "arn:aws:ivs:us-west-2:123456789012:stage/abcd1234ABCD",
      "name": "stage2",
      "tags": {}
    },
    {
      "activeSessionId": "st-abcDEF1234ghi",
      "arn": "arn:aws:ivs:us-west-2:123456789012:stage/ABCD1234efgh",
      "name": "stage3",
      "tags": {}
    }
  ]
}
```

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [ListStages](#) in *AWS CLI Command Reference*.

list-storage-configurations

The following code example shows how to use `list-storage-configurations`.

AWS CLI

To list composition storage configurations

The following `list-storage-configurations` lists all composition storage configurations for your AWS account, in the AWS region where the API request is processed.

```
aws ivs-realtime list-storage-configurations
```

Output:

```
{
  "storageConfigurations": [
    {
      "arn": "arn:aws:ivs:ap-northeast-1:123456789012:storage-configuration/abcdABCDefgh",
      "name": "test-sc-1",
      "s3": {
        "bucketName": "test-bucket-1-name"
      },
      "tags": {}
    },
    {
      "arn": "arn:aws:ivs:ap-northeast-1:123456789012:storage-configuration/ABCefgEFGabc",
      "name": "test-sc-2",
      "s3": {
        "bucketName": "test-bucket-2-name"
      },
      "tags": {}
    }
  ]
}
```

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [ListStorageConfigurations](#) in *AWS CLI Command Reference*.

start-composition

The following code example shows how to use start-composition.

AWS CLI

Example 1: To start a composition with default layout settings

The following start-composition example starts a composition for the specified stage to be streamed to the specified locations.

```
aws ivs-realtime start-composition \
  --stage-arn arn:aws:ivs:ap-northeast-1:123456789012:stage/defgABCdabcd \
  --destinations '[{"channel": {"channelArn": "arn:aws:ivs:ap-
northeast-1:123456789012:channel/abcABCdefDEg", \
  "encoderConfigurationArn": "arn:aws:ivs:ap-northeast-1:123456789012:encoder-
configuration/ABabCDcdEFef"}}, \
  {"s3":{"encoderConfigurationArns":["arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"], \
  "storageConfigurationArn":"arn:aws:ivs:ap-northeast-1:123456789012:storage-
configuration/FefABabCDcdE"}}]'
```

Output:

```
{
  "composition": {
    "arn": "arn:aws:ivs:ap-northeast-1:123456789012:composition/abcdABCDefgh",
    "destinations": [
      {
        "configuration": {
          "channel": {
            "channelArn": "arn:aws:ivs:ap-
northeast-1:123456789012:channel/abcABCdefDEg",
            "encoderConfigurationArn": "arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
          },
          "name": ""
        },
        "id": "AabBCcdDEefF",
        "state": "STARTING"
      },
      {
        "configuration": {
```

```

        "name": "",
        "s3": {
            "encoderConfigurationArns": [
                "arn:aws:ivs:arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
            ],
            "recordingConfiguration": {
                "format": "HLS"
            },
            "storageConfigurationArn": "arn:arn:aws:ivs:ap-
northeast-1:123456789012:storage-configuration/FefABabCDcdE"
        }
    },
    "detail": {
        "s3": {
            "recordingPrefix": "aBcDeFgHhGfE/AbCdEfGhHgFe/GHfabcgefABC/
composite"
        }
    },
    "id": "GHfabcgefABC",
    "state": "STARTING"
}
],
"layout": {
    "grid": {
        "featuredParticipantAttribute": "",
        "gridGap": 2,
        "omitStoppedVideo": false,
        "videoAspectRatio": "VIDEO",
        "videoFillMode": ""
    }
},
"stageArn": "arn:aws:ivs:ap-northeast-1:123456789012:stage/defgABCDabcd",
"startTime": "2023-10-16T23:24:00+00:00",
"state": "STARTING",
"tags": {}
}
}

```

For more information, see [Composite Recording \(Real-Time Streaming\)](#) in the *Amazon Interactive Video Service User Guide*.

Example 2: To start a composition with PiP layout

The following `start-composition` example starts a composition for the specified stage to be streamed to the specified locations using PiP layout.

```
aws ivs-realtime start-composition \
  --stage-arn arn:aws:ivs:ap-northeast-1:123456789012:stage/defgABCdabcd \
  --destinations '[{"channel": {"channelArn": "arn:aws:ivs:ap-
northeast-1:123456789012:channel/abcABCdefDEg"}, \
  "encoderConfigurationArn": "arn:aws:ivs:ap-northeast-1:123456789012:encoder-
configuration/ABabCDcdEFef"}], \
  {"s3":{"encoderConfigurationArns":["arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"], \
  "storageConfigurationArn":"arn:aws:ivs:ap-northeast-1:123456789012:storage-
configuration/FefABabCDcdE"}}]' \
  --layout pip='{featuredParticipantAttribute="abcdefg}"'
```

Output:

```
{
  "composition": {
    "arn": "arn:aws:ivs:ap-northeast-1:123456789012:composition/wxyzWXYZpqrs",
    "destinations": [
      {
        "configuration": {
          "channel": {
            "channelArn": "arn:aws:ivs:ap-
northeast-1:123456789012:channel/abcABCdefDEg",
            "encoderConfigurationArn": "arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
          },
          "name": ""
        },
        "id": "AabBCcdDEefF",
        "state": "STARTING"
      },
      {
        "configuration": {
          "name": "",
          "s3": {
            "encoderConfigurationArns": [
              "arn:aws:ivs:arn:aws:ivs:ap-
northeast-1:123456789012:encoder-configuration/ABabCDcdEFef"
            ],
            "recordingConfiguration": {
```

```

        "format": "HLS"
      },
      "storageConfigurationArn": "arn:arn:aws:ivs:ap-
northeast-1:123456789012:storage-configuration/FefABabCDcdE"
    }
  },
  "detail": {
    "s3": {
      "recordingPrefix": "aBcDeFgHhGfE/AbCdEfGhHgFe/GHFabcgefABC/
composite"
    }
  },
  "id": "GHFabcgefABC",
  "state": "STARTING"
}
],
"layout": {
  "pip": {
    "featuredParticipantAttribute": "abcdefg",
    "gridGap": 0,
    "omitStoppedVideo": false,
    "pipBehavior": "STATIC",
    "pipOffset": 0,
    "pipParticipantAttribute": "",
    "pipPosition": "BOTTOM_RIGHT",
    "videoFillMode": "COVER"
  }
},
"stageArn": "arn:aws:ivs:ap-northeast-1:123456789012:stage/defgABCDabcd",
"startTime": "2023-10-16T23:24:00+00:00",
"state": "STARTING",
"tags": {}
}
}

```

For more information, see [Composite Recording \(Real-Time Streaming\)](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [StartComposition](#) in *AWS CLI Command Reference*.

stop-composition

The following code example shows how to use stop-composition.

AWS CLI

To stop a composition

The following `stop-composition` stops the composition specified by the given ARN (Amazon Resource Name).

```
aws ivs-realtime stop-composition \  
  --arn "arn:aws:ivs:ap-northeast-1:123456789012:composition/abcdABCDefgh"
```

This command produces no output.

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [StopComposition](#) in *AWS CLI Command Reference*.

update-stage

The following code example shows how to use `update-stage`.

AWS CLI

To update a stage's configuration

The following `update-stage` example updates a stage for a specified stage ARN to update the stage name.

```
aws ivs-realtime update-stage \  
  --arn arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh \  
  --name stage1a
```

Output:

```
{  
  "stage": {  
    "arn": "arn:aws:ivs:us-west-2:123456789012:stage/abcdABCDefgh",  
    "name": "stage1a"  
  }  
}
```

For more information, see [Enabling Multiple Hosts on an Amazon IVS Stream](#) in the *Amazon Interactive Video Service User Guide*.

- For API details, see [UpdateStage](#) in *AWS CLI Command Reference*.

Amazon Kendra examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon Kendra.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-data-source

The following code example shows how to use `create-data-source`.

AWS CLI

To create an Amazon Kendra data source connector

The following `create-data-source` creates and configures an Amazon Kendra data source connector. You can use `describe-data-source` to view the status of a data source connector, and read any error messages if the status shows a data source connector "FAILED" to completely create.

```
aws kendra create-data-source \
```



```

--name "example data source 1" \
--description "Example data source 1 for example index 1 contains the first set
of example documents" \
--tags '{"Key": "test resources", "Value": "kendra"}, {"Key": "test resources",
"Value": "aws"}' \
--role-arn "arn:aws:iam::my-account-id:role/
KendraRoleForS3TemplateConfigDataSource" \
--index-id exampleindex1 \
--language-code "es" \
--schedule "0 0 18 ? * TUE,MON,WED,THU,FRI,SAT *" \
--configuration '{"TemplateConfiguration": {"Template": file://
s3schemaconfig.json}}' \
--type "TEMPLATE" \
--custom-document-enrichment-configuration '{"PostExtractionHookConfiguration":
{"LambdaArn": "arn:aws:iam::my-account-id:function/my-function-ocr-docs",
"S3Bucket": "s3://my-s3-bucket/scanned-image-text-example-docs"}, "RoleArn":
"arn:aws:iam:my-account-id:role/KendraRoleForCDE"}' \
--vpc-configuration '{"SecurityGroupIds": ["sg-1234567890abcdef0"], "SubnetIds":
["subnet-1c234", "subnet-2b134"]}'

```

Output:

```

{
  "Id": "exampledatasource1"
}

```

For more information, see [Getting started with an Amazon Kendra index and data source connector](#) in the *Amazon Kendra Developer Guide*.

- For API details, see [CreateDataSource](#) in *AWS CLI Command Reference*.

create-index

The following code example shows how to use `create-index`.

AWS CLI

To create an Amazon Kendra index

The following `create-index` creates and configures an Amazon Kendra index. You can use `describe-index` to view the status of an index, and read any error messages if the status shows an index "FAILED" to completely create.

```
aws kendra create-index \  
  --name "example index 1" \  
  --description "Example index 1 contains the first set of example documents" \  
  --tags '{"Key": "test resources", "Value": "kendra"}, {"Key": "test resources",  
"Value": "aws"}' \  
  --role-arn "arn:aws:iam::my-account-id:role/KendraRoleForExampleIndex" \  
  --edition "DEVELOPER_EDITION" \  
  --server-side-encryption-configuration '{"KmsKeyId": "my-kms-key-id"}' \  
  --user-context-policy "USER_TOKEN" \  
  --user-token-configurations '{"JsonTokenTypeConfiguration":  
{"GroupAttributeField": "groupNameField", "UserNameAttributeField":  
"userNameField"}}'
```

Output:

```
{  
  "Id": index1  
}
```

For more information, see [Getting started with an Amazon Kendra index and data source connector](#) in the *Amazon Kendra Developer Guide*.

- For API details, see [CreateIndex](#) in *AWS CLI Command Reference*.

describe-data-source

The following code example shows how to use describe-data-source.

AWS CLI

To get information about an Amazon Kendra data source connector

The following describe-data-source gets information about an Amazon Kendra data source connector. You can view the configuration of a data source connector, and read any error messages if the status shows a data source connector "FAILED" to completely create.

```
aws kendra describe-data-source \  
  --id exampledatasource1 \  
  --index-id exampleindex1
```

Output:

```
{
  "Configuration": {
    "TemplateConfiguration": {
      "Template": {
        "connectionConfiguration": {
          "repositoryEndpointMetadata": {
            "BucketName": "my-bucket"
          }
        },
        "repositoryConfigurations": {
          "document": {
            "fieldMappings": [
              {
                "indexFieldName": "_document_title",
                "indexFieldType": "STRING",
                "dataSourceFieldName": "title"
              },
              {
                "indexFieldName": "_last_updated_at",
                "indexFieldType": "DATE",
                "dataSourceFieldName": "modified_date"
              }
            ]
          }
        },
        "additionalProperties": {
          "inclusionPatterns": [
            "*.txt",
            "*.doc",
            "*.docx"
          ],
          "exclusionPatterns": [
            "*.json"
          ],
          "inclusionPrefixes": [
            "PublicExampleDocsFolder"
          ],
          "exclusionPrefixes": [
            "PrivateDocsFolder/private"
          ],
          "aclConfigurationFilePath": "ExampleDocsFolder/AclConfig.json",
          "metadataFilesPrefix": "metadata"
        }
      }
    }
  }
}
```

```

        "syncMode": "FULL_CRAWL",
        "type": "S3",
        "version": "1.0.0"
    }
},
"CreatedAt": 2024-02-25T13:30:10+00:00,
"CustomDocumentEnrichmentConfiguration": {
    "PostExtractionHookConfiguration": {
        "LambdaArn": "arn:aws:iam::my-account-id:function/my-function-ocr-docs",
        "S3Bucket": "s3://my-s3-bucket/scanned-image-text-example-docs/function"
    },
    "RoleArn": "arn:aws:iam:my-account-id:role/KendraRoleForCDE"
}
>Description": "Example data source 1 for example index 1 contains the first set
of example documents",
"Id": exampledatasource1,
"IndexId": exampleindex1,
"LanguageCode": "en",
"Name": "example data source 1",
"RoleArn": "arn:aws:iam::my-account-id:role/
KendraRoleForS3TemplateConfigDataSource",
"Schedule": "0 0 18 ? * TUE,MON,WED,THU,FRI,SAT *",
>Status": "ACTIVE",
>Type": "TEMPLATE",
>UpdatedAt": 1709163615,
>VpcConfiguration": {
    "SecurityGroupIds": ["sg-1234567890abcdef0"],
    "SubnetIds": ["subnet-1c234", "subnet-2b134"]
}
}

```

For more information, see [Getting started with an Amazon Kendra index and data source connector](#) in the *Amazon Kendra Developer Guide*.

- For API details, see [DescribeDataSource](#) in *AWS CLI Command Reference*.

describe-index

The following code example shows how to use `describe-index`.

AWS CLI

To get information about an Amazon Kendra index

The following `describe-index` gets information about an Amazon Kendra index. You can view the configuration of an index, and read any error messages if the status shows an index "FAILED" to completely create.

```
aws kendra describe-index \  
  --id exampleindex1
```

Output:

```
{  
  "CapacityUnits": {  
    "QueryCapacityUnits": 0,  
    "StorageCapacityUnits": 0  
  },  
  "CreatedAt": 2024-02-25T12:30:10+00:00,  
  "Description": "Example index 1 contains the first set of example documents",  
  "DocumentMetadataConfigurations": [  
    {  
      "Name": "_document_title",  
      "Relevance": {  
        "Importance": 8  
      },  
      "Search": {  
        "Displayable": true,  
        "Facetable": false,  
        "Searchable": true,  
        "Sortable": false  
      },  
      "Type": "STRING_VALUE"  
    },  
    {  
      "Name": "_document_body",  
      "Relevance": {  
        "Importance": 5  
      },  
      "Search": {  
        "Displayable": true,  
        "Facetable": false,  
        "Searchable": true,  
        "Sortable": false  
      }  
    }  
  ]  
}
```

```
        "Sortable": false
      },
      "Type": "STRING_VALUE"
    },
    {
      "Name": "_last_updated_at",
      "Relevance": {
        "Importance": 6,
        "Duration": "2628000s",
        "Freshness": true
      },
      "Search": {
        "Displayable": true,
        "Facetable": false,
        "Searchable": true,
        "Sortable": true
      },
      "Type": "DATE_VALUE"
    },
    {
      "Name": "department_custom_field",
      "Relevance": {
        "Importance": 7,
        "ValueImportanceMap": {
          "Human Resources" : 4,
          "Marketing and Sales" : 2,
          "Research and innvoation" : 3,
          "Admin" : 1
        }
      },
      "Search": {
        "Displayable": true,
        "Facetable": true,
        "Searchable": true,
        "Sortable": true
      },
      "Type": "STRING_VALUE"
    }
  ],
  "Edition": "DEVELOPER_EDITION",
  "Id": "index1",
  "IndexStatistics": {
    "FaqStatistics": {
      "IndexedQuestionAnswersCount": 10
    }
  }
}
```

```

    },
    "TextDocumentStatistics": {
      "IndexedTextBytes": 1073741824,
      "IndexedTextDocumentsCount": 1200
    }
  },
  "Name": "example index 1",
  "RoleArn": "arn:aws:iam::my-account-id:role/KendraRoleForExampleIndex",
  "ServerSideEncryptionConfiguration": {
    "KmsKeyId": "my-kms-key-id"
  },
  "Status": "ACTIVE",
  "UpdatedAt": 1709163615,
  "UserContextPolicy": "USER_TOKEN",
  "UserTokenConfigurations": [
    {
      "JsonTokenTypeConfiguration": {
        "GroupAttributeField": "groupNameField",
        "UserNameAttributeField": "userNameField"
      }
    }
  ]
}

```

For more information, see [Getting started with an Amazon Kendra index and data source connector](#) in the *Amazon Kendra Developer Guide*.

- For API details, see [DescribeIndex](#) in *AWS CLI Command Reference*.

update-data-source

The following code example shows how to use `update-data-source`.

AWS CLI

To update an Amazon Kendra data source connector

The following `update-data-source` updates the configuration of an Amazon Kendra data source connector. If the action is successful, the service either sends back no output, the HTTP status code 200, or the AWS CLI return code 0. You can use `describe-data-source` to view the configuration and status of a data source connector.

```
aws kendra update-data-source \
```

```

--id exampledatasource1 \
--index-id exampleindex1 \
--name "new name for example data source 1" \
--description "new description for example data source 1" \
--role-arn arn:aws:iam::my-account-id:role/KendraNewRoleForExampleDataSource \
--configuration '{"TemplateConfiguration": {"Template": file://
s3schemanewconfig.json}}' \
--custom-document-enrichment-configuration '{"PostExtractionHookConfiguration":
{"LambdaArn": "arn:aws:iam::my-account-id:function/my-function-ocr-docs",
"S3Bucket": "s3://my-s3-bucket/scanned-image-text-example-docs"}, "RoleArn":
"arn:aws:iam:my-account-id:role/KendraNewRoleForCDE"}' \
--language-code "es" \
--schedule "0 0 18 ? * MON,WED,FRI *" \
--vpc-configuration '{"SecurityGroupIds": ["sg-1234567890abcdef0"], "SubnetIds":
["subnet-1c234", "subnet-2b134"]}'

```

This command produces no output.

For more information, see [Getting started with an Amazon Kendra index and data source connector](#) in the *Amazon Kendra Developer Guide*.

- For API details, see [UpdateDataSource](#) in *AWS CLI Command Reference*.

update-index

The following code example shows how to use update-index.

AWS CLI

To update an Amazon Kendra index

The following update-index updates the configuration of an Amazon Kendra index. If the action is successful, the service either sends back no output, the HTTP status code 200, or the AWS CLI return code 0. You can use describe-index to view the configuration and status of an index.

```

aws kendra update-index \
--id enterpriseindex1 \
--name "new name for Enterprise Edition index 1" \
--description "new description for Enterprise Edition index 1" \
--role-arn arn:aws:iam::my-account-id:role/KendraNewRoleForEnterpriseIndex \
--capacity-units '{"QueryCapacityUnits": 2, "StorageCapacityUnits": 1}' \

```



```
--document-metadata-configuration-updates '{"Name": "_document_title",
"Relevance": {"Importance": 6}}, {"Name": "_last_updated_at", "Relevance":
{"Importance": 8}}' \
--user-context-policy "USER_TOKEN" \
--user-token-configurations '{"JsonTokenTypeConfiguration":
{"GroupAttributeField": "groupNameField", "UserNameAttributeField":
"userNameField"}}'
```

This command produces no output.

For more information, see [Getting started with an Amazon Kendra index and data source connector](#) in the *Amazon Kendra Developer Guide*.

- For API details, see [UpdateIndex](#) in *AWS CLI Command Reference*.

Kinesis examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Kinesis.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-tags-to-stream

The following code example shows how to use `add-tags-to-stream`.

AWS CLI

To add tags to a data stream

The following `add-tags-to-stream` example assigns a tag with the key `samplekey` and value `example` to the specified stream.

```
aws kinesis add-tags-to-stream \  
  --stream-name samplestream \  
  --tags samplekey=example
```

This command produces no output.

For more information, see [Tagging Your Streams](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [AddTagsToStream](#) in *AWS CLI Command Reference*.

create-stream

The following code example shows how to use `create-stream`.

AWS CLI

To create a data stream

The following `create-stream` example creates a data stream named `samplestream` with 3 shards.

```
aws kinesis create-stream \  
  --stream-name samplestream \  
  --shard-count 3
```

This command produces no output.

For more information, see [Creating a Stream](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [CreateStream](#) in *AWS CLI Command Reference*.

decrease-stream-retention-period

The following code example shows how to use `decrease-stream-retention-period`.

AWS CLI

To decrease data stream retention period

The following `decrease-stream-retention-period` example decreases the retention period (the length of time data records are accessible after they are added to the stream) of a stream named `samplestream` to 48 hours.

```
aws kinesis decrease-stream-retention-period \  
  --stream-name samplestream \  
  --retention-period-hours 48
```

This command produces no output.

For more information, see [Changing the Data Retention Period](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [DecreaseStreamRetentionPeriod](#) in *AWS CLI Command Reference*.

`delete-stream`

The following code example shows how to use `delete-stream`.

AWS CLI

To delete a data stream

The following `delete-stream` example deletes the specified data stream.

```
aws kinesis delete-stream \  
  --stream-name samplestream
```

This command produces no output.

For more information, see [Deleting a Stream](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [DeleteStream](#) in *AWS CLI Command Reference*.

`deregister-stream-consumer`

The following code example shows how to use `deregister-stream-consumer`.

AWS CLI

To deregister a data stream consumer

The following `deregister-stream-consumer` example deregisters the specified consumer from the specified data stream.

```
aws kinesis deregister-stream-consumer \  
  --stream-arn arn:aws:kinesis:us-west-2:123456789012:stream/samplestream \  
  --consumer-name KinesisConsumerApplication
```

This command produces no output.

For more information, see [Developing Consumers with Enhanced Fan-Out Using the Kinesis Data Streams API](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [DeregisterStreamConsumer](#) in *AWS CLI Command Reference*.

describe-limits

The following code example shows how to use `describe-limits`.

AWS CLI

To describe shard limits

The following `describe-limits` example displays the shard limits and usage for the current AWS account.

```
aws kinesis describe-limits
```

Output:

```
{  
  "ShardLimit": 500,  
  "OpenShardCount": 29  
}
```

For more information, see [Resharding a Stream](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [DescribeLimits](#) in *AWS CLI Command Reference*.

describe-stream-consumer

The following code example shows how to use `describe-stream-consumer`.

AWS CLI

To describe a data stream consumer

The following `describe-stream-consumer` example returns the description of the specified consumer, registered with the specified data stream.

```
aws kinesis describe-stream-consumer \  
  --stream-arn arn:aws:kinesis:us-west-2:012345678912:stream/samplestream \  
  --consumer-name KinesisConsumerApplication
```

Output:

```
{  
  "ConsumerDescription": {  
    "ConsumerName": "KinesisConsumerApplication",  
    "ConsumerARN": "arn:aws:kinesis:us-west-2:123456789012:stream/samplestream/  
consumer/KinesisConsumerApplication:1572383852",  
    "ConsumerStatus": "ACTIVE",  
    "ConsumerCreationTimestamp": 1572383852.0,  
    "StreamARN": "arn:aws:kinesis:us-west-2:123456789012:stream/samplestream"  
  }  
}
```

For more information, see [Reading Data from Amazon Kinesis Data Streams](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [DescribeStreamConsumer](#) in *AWS CLI Command Reference*.

describe-stream-summary

The following code example shows how to use `describe-stream-summary`.

AWS CLI

To describe a data stream summary

The following `describe-stream-summary` example provides a summarized description (without the shard list) of the specified data stream.

```
aws kinesis describe-stream-summary \  
  --stream-name samplestream
```

Output:

```
{  
  "StreamDescriptionSummary": {  
    "StreamName": "samplestream",  
    "StreamARN": "arn:aws:kinesis:us-west-2:123456789012:stream/samplestream",  
    "StreamStatus": "ACTIVE",  
    "RetentionPeriodHours": 48,  
    "StreamCreationTimestamp": 1572297168.0,  
    "EnhancedMonitoring": [  
      {  
        "ShardLevelMetrics": []  
      }  
    ],  
    "EncryptionType": "NONE",  
    "OpenShardCount": 3,  
    "ConsumerCount": 0  
  }  
}
```

For more information, see [Creating and Managing Streams](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [DescribeStreamSummary](#) in *AWS CLI Command Reference*.

describe-stream

The following code example shows how to use `describe-stream`.

AWS CLI

To describe a data stream

The following `describe-stream` example returns the details of the specified data stream.

```
aws kinesis describe-stream \  
  --stream-name samplestream
```

```
--stream-name samplestream
```

Output:

```
{
  "StreamDescription": {
    "Shards": [
      {
        "ShardId": "shardId-000000000000",
        "HashKeyRange": {
          "StartingHashKey": "0",
          "EndingHashKey": "113427455640312821154458202477256070484"
        },
        "SequenceNumberRange": {
          "StartingSequenceNumber":
"49600871682957036442365024926191073437251060580128653314"
        }
      },
      {
        "ShardId": "shardId-000000000001",
        "HashKeyRange": {
          "StartingHashKey": "113427455640312821154458202477256070485",
          "EndingHashKey": "226854911280625642308916404954512140969"
        },
        "SequenceNumberRange": {
          "StartingSequenceNumber":
"49600871682979337187563555549332609155523708941634633746"
        }
      },
      {
        "ShardId": "shardId-000000000002",
        "HashKeyRange": {
          "StartingHashKey": "226854911280625642308916404954512140970",
          "EndingHashKey": "340282366920938463463374607431768211455"
        },
        "SequenceNumberRange": {
          "StartingSequenceNumber":
"49600871683001637932762086172474144873796357303140614178"
        }
      }
    ],
    "StreamARN": "arn:aws:kinesis:us-west-2:123456789012:stream/samplestream",
    "StreamName": "samplestream",
  }
}
```

```
    "StreamStatus": "ACTIVE",
    "RetentionPeriodHours": 24,
    "EnhancedMonitoring": [
      {
        "ShardLevelMetrics": []
      }
    ],
    "EncryptionType": "NONE",
    "KeyId": null,
    "StreamCreationTimestamp": 1572297168.0
  }
}
```

For more information, see [Creating and Managing Streams](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [DescribeStream](#) in *AWS CLI Command Reference*.

disable-enhanced-monitoring

The following code example shows how to use `disable-enhanced-monitoring`.

AWS CLI

To disable enhanced monitoring for shard-level metrics

The following `disable-enhanced-monitoring` example disables enhanced Kinesis data stream monitoring for shard-level metrics.

```
aws kinesis disable-enhanced-monitoring \
  --stream-name samplestream --shard-level-metrics ALL
```

Output:

```
{
  "StreamName": "samplestream",
  "CurrentShardLevelMetrics": [
    "IncomingBytes",
    "OutgoingRecords",
    "IteratorAgeMilliseconds",
    "IncomingRecords",
    "ReadProvisionedThroughputExceeded",
```



```

        "WriteProvisionedThroughputExceeded",
        "OutgoingBytes"
    ],
    "DesiredShardLevelMetrics": []
}

```

For more information, see [Monitoring Streams in Amazon Kinesis Data Streams](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [DisableEnhancedMonitoring](#) in *AWS CLI Command Reference*.

enable-enhanced-monitoring

The following code example shows how to use `enable-enhanced-monitoring`.

AWS CLI

To enable enhanced monitoring for shard-level metrics

The following `enable-enhanced-monitoring` example enables enhanced Kinesis data stream monitoring for shard-level metrics.

```

aws kinesis enable-enhanced-monitoring \
  --stream-name samplestream \
  --shard-level-metrics ALL

```

Output:

```

{
  "StreamName": "samplestream",
  "CurrentShardLevelMetrics": [],
  "DesiredShardLevelMetrics": [
    "IncomingBytes",
    "OutgoingRecords",
    "IteratorAgeMilliseconds",
    "IncomingRecords",
    "ReadProvisionedThroughputExceeded",
    "WriteProvisionedThroughputExceeded",
    "OutgoingBytes"
  ]
}

```

For more information, see [Monitoring Streams in Amazon Kinesis Data Streams](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [EnableEnhancedMonitoring](#) in *AWS CLI Command Reference*.

get-records

The following code example shows how to use `get-records`.

AWS CLI

To obtain records from a shard

The following `get-records` example gets data records from a Kinesis data stream's shard using the specified shard iterator.

```
aws kinesis get-records \  
  --shard-iterator AAAAAAAAAAF7/0mWD7IuHj1yGv/  
TKuNgx2ukD5xipCY4cy4gU96orWwZwcSXh3K9tAmGYe0ZyLZrvzze0FVf9iN99hUPw/w/  
b0YWYeefNvnf1DYt5XpDJghLKr3DzgzknTmMymDP3R+3wRKeuEw6/kdxY2yKJH0veaiekaVc4N2VwK/  
GvaGP2Hh9Fg7N++q0Adg6fIDQPt4p8RpavDbk+A4sL9SWGE1
```

Output:

```
{  
  "Records": [],  
  "MillisBehindLatest": 80742000  
}
```

For more information, see [Developing Consumers Using the Kinesis Data Streams API with the AWS SDK for Java](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [GetRecords](#) in *AWS CLI Command Reference*.

get-shard-iterator

The following code example shows how to use `get-shard-iterator`.

AWS CLI

To obtain a shard iterator

The following `get-shard-iterator` example uses the `AT_SEQUENCE_NUMBER` shard iterator type and generates a shard iterator to start reading data records exactly from the position denoted by the specified sequence number.

```
aws kinesis get-shard-iterator \  
  --stream-name samplestream \  
  --shard-id shardId-000000000001 \  
  --shard-iterator-type LATEST
```

Output:

```
{  
  "ShardIterator": "AAAAAAAAAAFEvJjIYI+3jw/4aqqH9FifJ+n48XWTh/  
IFIsbILP6o5eDueD39NXNBfpZ10WL5K6ADXk8w+5H+Qhd9cFA9k268CPXCz/kebq1TGYI7Vy  
+1UkA9BuN3xvATxMBGxRY3zYK05gqgvaIRn9408SqeEqwhigwZxNWxID3Ej7YYYcxQi8Q/fIrCjGAY/  
n2r5Z9G864YpWdfN9upNNQAR/ii0WKs"  
}
```

For more information, see [Developing Consumers Using the Kinesis Data Streams API with the AWS SDK for Java](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [GetShardIterator](#) in *AWS CLI Command Reference*.

increase-stream-retention-period

The following code example shows how to use `increase-stream-retention-period`.

AWS CLI

To increase data stream retention period

The following `increase-stream-retention-period` example increases the retention period (the length of time data records are accessible after they are added to the stream) of the specified stream to 168 hours.

```
aws kinesis increase-stream-retention-period \  
  --stream-name samplestream \  
  --retention-period-hours 168
```

This command produces no output.

For more information, see [Changing the Data Retention Period](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [IncreaseStreamRetentionPeriod](#) in *AWS CLI Command Reference*.

list-shards

The following code example shows how to use `list-shards`.

AWS CLI

To list shards in a data stream

The following `list-shards` example lists all shards in the specified stream starting with the shard whose ID immediately follows the specified `exclusive-start-shard-id` of `shardId-000000000000`.

```
aws kinesis list-shards \  
  --stream-name samplestream \  
  --exclusive-start-shard-id shardId-000000000000
```

Output:

```
{  
  "Shards": [  
    {  
      "ShardId": "shardId-000000000001",  
      "HashKeyRange": {  
        "StartingHashKey": "113427455640312821154458202477256070485",  
        "EndingHashKey": "226854911280625642308916404954512140969"  
      },  
      "SequenceNumberRange": {  
        "StartingSequenceNumber":  
"4960087168297933718756355549332609155523708941634633746"  
      }  
    },  
    {  
      "ShardId": "shardId-000000000002",  
      "HashKeyRange": {  
        "StartingHashKey": "226854911280625642308916404954512140970",  
        "EndingHashKey": "340282366920938463463374607431768211455"  
      },  
    }  
  ]  
}
```

```
        "SequenceNumberRange": {
            "StartingSequenceNumber":
"49600871683001637932762086172474144873796357303140614178"
        }
    ]
}
```

For more information, see [Listing Shards](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [ListShards](#) in *AWS CLI Command Reference*.

list-streams

The following code example shows how to use `list-streams`.

AWS CLI

To list data streams

The following `list-streams` example lists all active data streams in the current account and region.

```
aws kinesis list-streams
```

Output:

```
{
  "StreamNames": [
    "samplestream",
    "samplestream1"
  ]
}
```

For more information, see [Listing Streams](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [ListStreams](#) in *AWS CLI Command Reference*.

list-tags-for-stream

The following code example shows how to use `list-tags-for-stream`.

AWS CLI

To list tags for a data stream

The following `list-tags-for-stream` example lists the tags attached to the specified data stream.

```
aws kinesis list-tags-for-stream \  
  --stream-name samplestream
```

Output:

```
{  
  "Tags": [  
    {  
      "Key": "samplekey",  
      "Value": "example"  
    }  
  ],  
  "HasMoreTags": false  
}
```

For more information, see [Tagging Your Streams](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [ListTagsForStream](#) in *AWS CLI Command Reference*.

merge-shards

The following code example shows how to use `merge-shards`.

AWS CLI

To merge shards

The following `merge-shards` example merges two adjacent shards with IDs of `shardId-000000000000` and `shardId-000000000001` in the specified data stream and combines them into a single shard.

```
aws kinesis merge-shards \  
  --stream-name samplestream \  
  --shard-to-merge shardId-000000000000 \  
  --shard-to-merge shardId-000000000001
```

```
--adjacent-shard-to-merge shardId-000000000001
```

This command produces no output.

For more information, see [Merging Two Shards](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [MergeShards](#) in *AWS CLI Command Reference*.

put-record

The following code example shows how to use `put-record`.

AWS CLI

To write a record into a data stream

The following `put-record` example writes a single data record into the specified data stream using the specified partition key.

```
aws kinesis put-record \  
  --stream-name samplestream \  
  --data sampledatarecord \  
  --partition-key samplepartitionkey
```

Output:

```
{  
  "ShardId": "shardId-000000000009",  
  "SequenceNumber": "49600902273357540915989931256901506243878407835297513618",  
  "EncryptionType": "KMS"  
}
```

For more information, see [Developing Producers Using the Amazon Kinesis Data Streams API with the AWS SDK for Java](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [PutRecord](#) in *AWS CLI Command Reference*.

put-records

The following code example shows how to use `put-records`.

AWS CLI

To write multiple records into a data stream

The following `put-records` example writes a data record using the specified partition key and another data record using a different partition key in a single call.

```
aws kinesis put-records \  
  --stream-name samplestream \  
  --records Data=blob1,PartitionKey=partitionkey1  
           Data=blob2,PartitionKey=partitionkey2
```

Output:

```
{  
  "FailedRecordCount": 0,  
  "Records": [  
    {  
      "SequenceNumber":  
"49600883331171471519674795588238531498465399900093808706",  
      "ShardId": "shardId-000000000004"  
    },  
    {  
      "SequenceNumber":  
"49600902273357540915989931256902715169698037101720764562",  
      "ShardId": "shardId-000000000009"  
    }  
  ],  
  "EncryptionType": "KMS"  
}
```

For more information, see [Developing Producers Using the Amazon Kinesis Data Streams API with the AWS SDK for Java](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [PutRecords](#) in *AWS CLI Command Reference*.

register-stream-consumer

The following code example shows how to use `register-stream-consumer`.

AWS CLI

To register a data stream consumer

The following `register-stream-consumer` example registers a consumer called `KinesisConsumerApplication` with the specified data stream.

```
aws kinesis register-stream-consumer \  
  --stream-arn arn:aws:kinesis:us-west-2:012345678912:stream/samplestream \  
  --consumer-name KinesisConsumerApplication
```

Output:

```
{  
  "Consumer": {  
    "ConsumerName": "KinesisConsumerApplication",  
    "ConsumerARN": "arn:aws:kinesis:us-west-2:123456789012:stream/samplestream/  
consumer/KinesisConsumerApplication:1572383852",  
    "ConsumerStatus": "CREATING",  
    "ConsumerCreationTimestamp": 1572383852.0  
  }  
}
```

For more information, see [Developing Consumers with Enhanced Fan-Out Using the Kinesis Data Streams API](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [RegisterStreamConsumer](#) in *AWS CLI Command Reference*.

remove-tags-from-stream

The following code example shows how to use `remove-tags-from-stream`.

AWS CLI

To remove tags from a data stream

The following `remove-tags-from-stream` example removes the tag with the specified key from the specified data stream.

```
aws kinesis remove-tags-from-stream \  
  --stream-name samplestream \  
  --tag-keys samplekey
```

This command produces no output.

For more information, see [Tagging Your Streams](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [RemoveTagsFromStream](#) in *AWS CLI Command Reference*.

split-shard

The following code example shows how to use `split-shard`.

AWS CLI

To split shards

The following `split-shard` example splits the specified shard into two new shards using a new starting hash key of 10.

```
aws kinesis split-shard \  
  --stream-name samplestream \  
  --shard-to-split shardId-000000000000 \  
  --new-starting-hash-key 10
```

This command produces no output.

For more information, see [Splitting a Shard](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [SplitShard](#) in *AWS CLI Command Reference*.

start-stream-encryption

The following code example shows how to use `start-stream-encryption`.

AWS CLI

To enable data stream encryption

The following `start-stream-encryption` example enables server-side encryption for the specified stream, using the specified AWS KMS key.

```
aws kinesis start-stream-encryption \  
  --encryption-type KMS \  
  --stream-name samplestream
```

```
--key-id arn:aws:kms:us-west-2:012345678912:key/a3c4a7cd-728b-45dd-  
b334-4d3eb496e452 \  
--stream-name samplestream
```

This command produces no output.

For more information, see [Data Protection in Amazon Kinesis Data Streams](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [StartStreamEncryption](#) in *AWS CLI Command Reference*.

stop-stream-encryption

The following code example shows how to use stop-stream-encryption.

AWS CLI

To disable data stream encryption

The following stop-stream-encryption example disables server-side encryption for the specified stream, using the specified AWS KMS key.

```
aws kinesis start-stream-encryption \  
  --encryption-type KMS \  
  --key-id arn:aws:kms:us-west-2:012345678912:key/a3c4a7cd-728b-45dd-  
b334-4d3eb496e452 \  
  --stream-name samplestream
```

This command produces no output.

For more information, see [Data Protection in Amazon Kinesis Data Streams](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [StopStreamEncryption](#) in *AWS CLI Command Reference*.

update-shard-count

The following code example shows how to use update-shard-count.

AWS CLI

To update the shard count in a data stream

The following `update-shard-count` example updates the shard count of the specified data stream to 6. This example uses uniform scaling, which creates shards of equal size.

```
aws kinesis update-shard-count \  
  --stream-name samplestream \  
  --scaling-type UNIFORM_SCALING \  
  --target-shard-count 6
```

Output:

```
{  
  "StreamName": "samplestream",  
  "CurrentShardCount": 3,  
  "TargetShardCount": 6  
}
```

For more information, see [Resharding a Stream](#) in the *Amazon Kinesis Data Streams Developer Guide*.

- For API details, see [UpdateShardCount](#) in *AWS CLI Command Reference*.

AWS KMS examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS KMS.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

cancel-key-deletion

The following code example shows how to use `cancel-key-deletion`.

AWS CLI

To cancel the scheduled deletion of a customer managed KMS key

The following `cancel-key-deletion` example cancels the scheduled deletion of a customer managed KMS key.

```
aws kms cancel-key-deletion \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Output:

```
{  
  "KeyId": "arn:aws:kms:us-  
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"  
}
```

When the `cancel-key-deletion` command succeeds, the scheduled deletion is canceled. However, the key state of the KMS key is `Disabled`, so you can't use the KMS key in cryptographic operations. To restore its functionality, use the `enable-key` command .

For more information, see [Scheduling and canceling key deletion](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [CancelKeyDeletion](#) in *AWS CLI Command Reference*.

connect-custom-key-store

The following code example shows how to use `connect-custom-key-store`.

AWS CLI

To connect a custom key store

The following `connect-custom-key-store` example reconnects the specified custom key store. You can use a command like this one to connect a custom key store for the first time or to reconnect a key store that was disconnected.

You can use this command to connect an AWS CloudHSM key store or an external key store.

```
aws kms connect-custom-key-store \  
  --custom-key-store-id cks-1234567890abcdef0
```

This command does not return any output. To verify that the command was effective, use the `describe-custom-key-stores` command.

For information about connecting an AWS CloudHSM key store, see [Connecting and disconnecting an AWS CloudHSM key store](#) in the *AWS Key Management Service Developer Guide*.

For information about connecting an external key store, see [Connecting and disconnecting an external key store](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [ConnectCustomKeyStore](#) in *AWS CLI Command Reference*.

create-alias

The following code example shows how to use `create-alias`.

AWS CLI

To create an alias for a KMS key

The following `create-alias` command creates an alias named `example-alias` for the KMS key identified by key ID `1234abcd-12ab-34cd-56ef-1234567890ab`.

Alias names must begin with `alias/`. Do not use alias names that begin with `alias/aws`; these are reserved for use by AWS.

```
aws kms create-alias \  
  --alias-name alias/example-alias \  
  --target-key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

This command doesn't return any output. To see the new alias, use the `list-aliases` command.

For more information, see [Using aliases](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [CreateAlias](#) in *AWS CLI Command Reference*.

create-custom-key-store

The following code example shows how to use `create-custom-key-store`.

AWS CLI

Example 1: To create an AWS CloudHSM key store

The following `create-custom-key-store` example creates an AWS CloudHSM key store backed by an AWS CloudHSM cluster using the required parameters. You can also add the `custom-key-store-type` parameter with the default value: `AWS_CLOUDHSM`.

To specify the file input for the `trust-anchor-certificate` command in the AWS CLI, the `file://` prefix is required.

```
aws kms create-custom-key-store \  
  --custom-key-store-name ExampleCloudHSMKeyStore \  
  --cloud-hsm-cluster-id cluster-1a23b4cdefg \  
  --key-store-password kmsPswd \  
  --trust-anchor-certificate file://customerCA.crt
```

Output:

```
{  
  "CustomKeyId": cks-1234567890abcdef0  
}
```

For more information, see [Creating an AWS CloudHSM key store](#) in the *AWS Key Management Service Developer Guide*.

Example 2: To create an external key store with public endpoint connectivity

The following `create-custom-key-store` example creates an external key store (XKS) that communicates with AWS KMS over the internet.

In this example, the `XksProxyUriPath` uses an optional prefix of `example-prefix`.

NOTE: If you use AWS CLI version 1.0, run the following command before specifying a parameter with an HTTP or HTTPS value, such as the `XksProxyUriEndpoint` parameter.

```
aws configure set cli_follow_urlparam false
```

Otherwise, AWS CLI version 1.0 replaces the parameter value with the content found at that URI address.

```
aws kms create-custom-key-store \  
  --custom-key-store-name ExamplePublicEndpointXKS \  
  --custom-key-store-type EXTERNAL_KEY_STORE \  
  --xks-proxy-connectivity PUBLIC_ENDPOINT \  
  --xks-proxy-uri-endpoint "https://myproxy.xks.example.com" \  
  --xks-proxy-uri-path "/example-prefix/kms/xks/v1" \  
  --xks-proxy-authentication-credential "AccessKeyId=ABCDE12345670EXAMPLE,  
RawSecretAccessKey=DXjSUawne12fr6SKC7G25CNxTyWKE5PF9XX6H/u9pSo="
```

Output:

```
{  
  "CustomKeyId": cks-2234567890abcdef0  
}
```

For more information, see [Creating an external key store](#) in the *AWS Key Management Service Developer Guide*.

Example 3: To create an external key store with VPC endpoint service connectivity

The following `create-custom-key-store` example creates an external key store (XKS) that uses an Amazon VPC endpoint service to communicate with AWS KMS.

NOTE: If you use AWS CLI version 1.0, run the following command before specifying a parameter with an HTTP or HTTPS value, such as the `XksProxyUriEndpoint` parameter.

```
aws configure set cli_follow_urlparam false
```

Otherwise, AWS CLI version 1.0 replaces the parameter value with the content found at that URI address.

```
aws kms create-custom-key-store \  
  --custom-key-store-name ExamplePublicEndpointXKS \  
  --custom-key-store-type EXTERNAL_KEY_STORE \  
  --xks-proxy-connectivity VPC_ENDPOINT \  
  --xks-proxy-uri-endpoint "https://myproxy.xks.example.com" \  
  --xks-proxy-uri-path "/example-prefix/kms/xks/v1" \  
  --xks-proxy-authentication-credential "AccessKeyId=ABCDE12345670EXAMPLE,  
RawSecretAccessKey=DXjSUawne12fr6SKC7G25CNxTyWKE5PF9XX6H/u9pSo="
```



```

--custom-key-store-name ExampleVPCEndpointXKS \
--custom-key-store-type EXTERNAL_KEY_STORE \
--xks-proxy-connectivity VPC_ENDPOINT_SERVICE \
--xks-proxy-uri-endpoint "https://myproxy-private.xks.example.com" \
--xks-proxy-uri-path "/kms/xks/v1" \
--xks-proxy-vpc-endpoint-service-name "com.amazonaws.vpce.us-east-1.vpce-svc-
example1" \
--xks-proxy-authentication-credential "AccessKeyId=ABCDE12345670EXAMPLE,
RawSecretAccessKey=DXjSUawne12fr6SKC7G25CNxTyWKE5PF9XX6H/u9pSo="

```

Output:

```

{
  "CustomKeyId": cks-3234567890abcdef0
}

```

For more information, see [Creating an external key store](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [CreateCustomKeyStore](#) in *AWS CLI Command Reference*.

create-grant

The following code example shows how to use `create-grant`.

AWS CLI

To create a grant

The following `create-grant` example creates a grant that allows the `exampleUser` user to use the `decrypt` command on the `1234abcd-12ab-34cd-56ef-1234567890ab` example KMS key. The retiring principal is the `adminRole` role. The grant uses the `EncryptionContextSubset` grant constraint to allow this permission only when the encryption context in the `decrypt` request includes the `"Department": "IT"` key-value pair.

```

aws kms create-grant \
--key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
--grantee-principal arn:aws:iam::123456789012:user/exampleUser \
--operations Decrypt \
--constraints EncryptionContextSubset={Department=IT} \
--retiring-principal arn:aws:iam::123456789012:role/adminRole

```

Output:

```
{
  "GrantId": "1a2b3c4d2f5e69f440bae30eaec9570bb1fb7358824f9ddfa1aa5a0dab1a59b2",
  "GrantToken": "<grant token here>"
}
```

To view detailed information about the grant, use the `list-grants` command.

For more information, see [Grants in AWS KMS](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [CreateGrant](#) in *AWS CLI Command Reference*.

create-key

The following code example shows how to use `create-key`.

AWS CLI**Example 1: To create a customer managed KMS key in AWS KMS**

The following `create-key` example creates a symmetric encryption KMS key.

To create the basic KMS key, a symmetric encryption key, you do not need to specify any parameters. The default values for those parameters create a symmetric encryption key.

Because this command doesn't specify a key policy, the KMS key gets the [default key policy](#) for programmatically created KMS keys. To view the key policy, use the `get-key-policy` command. To change the key policy, use the `put-key-policy` command.

```
aws kms create-key
```

The `create-key` command returns the key metadata, including the key ID and ARN of the new KMS key. You can use these values to identify the KMS key in other AWS KMS operations. The output does not include the tags. To view the tags for a KMS key, use the `list-resource-tags` command.

Output:

```
{
  "KeyMetadata": {
```

```

    "AWSAccountId": "111122223333",
    "Arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "CreationDate": "2017-07-05T14:04:55-07:00",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "Description": "",
    "Enabled": true,
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyManager": "CUSTOMER",
    "KeySpec": "SYMMETRIC_DEFAULT",
    "KeyState": "Enabled",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "MultiRegion": false,
    "Origin": "AWS_KMS"
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ]
  }
}

```

Note: The `create-key` command does not let you specify an alias. To create an alias for the new KMS key, use the `create-alias` command.

For more information, see [Creating keys](#) in the *AWS Key Management Service Developer Guide*.

Example 2: To create an asymmetric RSA KMS key for encryption and decryption

The following `create-key` example creates a KMS key that contains an asymmetric RSA key pair for encryption and decryption.

```

aws kms create-key \
  --key-spec RSA_4096 \
  --key-usage ENCRYPT_DECRYPT

```

Output:

```

{
  "KeyMetadata": {
    "Arn": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "AWSAccountId": "111122223333",
    "CreationDate": "2021-04-05T14:04:55-07:00",
    "CustomerMasterKeySpec": "RSA_4096",

```

```

    "Description": "",
    "Enabled": true,
    "EncryptionAlgorithms": [
      "RSAES_OAEP_SHA_1",
      "RSAES_OAEP_SHA_256"
    ],
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyManager": "CUSTOMER",
    "KeySpec": "RSA_4096",
    "KeyState": "Enabled",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "MultiRegion": false,
    "Origin": "AWS_KMS"
  }
}

```

For more information, see [Asymmetric keys in AWS KMS](#) in the *AWS Key Management Service Developer Guide*.

Example 3: To create an asymmetric elliptic curve KMS key for signing and verification

To create an asymmetric KMS key that contains an asymmetric elliptic curve (ECC) key pair for signing and verification. The `--key-usage` parameter is required even though `SIGN_VERIFY` is the only valid value for ECC KMS keys.

```

aws kms create-key \
  --key-spec ECC_NIST_P521 \
  --key-usage SIGN_VERIFY

```

Output:

```

{
  "KeyMetadata": {
    "Arn": "arn:aws:kms:us-east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "AWSAccountId": "111122223333",
    "CreationDate": "2019-12-02T07:48:55-07:00",
    "CustomerMasterKeySpec": "ECC_NIST_P521",
    "Description": "",
    "Enabled": true,
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyManager": "CUSTOMER",

```

```

    "KeySpec": "ECC_NIST_P521",
    "KeyState": "Enabled",
    "KeyUsage": "SIGN_VERIFY",
    "MultiRegion": false,
    "Origin": "AWS_KMS",
    "SigningAlgorithms": [
      "ECDSA_SHA_512"
    ]
  }
}

```

For more information, see [Asymmetric keys in AWS KMS](#) in the *AWS Key Management Service Developer Guide*.

Example 4: To create an HMAC KMS key

The following create-key example creates a 384-bit HMAC KMS key. The `GENERATE_VERIFY_MAC` value for the `--key-usage` parameter is required even though it's the only valid value for HMAC KMS keys.

```

aws kms create-key \
  --key-spec HMAC_384 \
  --key-usage GENERATE_VERIFY_MAC

```

Output:

```

{
  "KeyMetadata": {
    "Arn": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "AWSAccountId": "111122223333",
    "CreationDate": "2022-04-05T14:04:55-07:00",
    "CustomerMasterKeySpec": "HMAC_384",
    "Description": "",
    "Enabled": true,
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyManager": "CUSTOMER",
    "KeySpec": "HMAC_384",
    "KeyState": "Enabled",
    "KeyUsage": "GENERATE_VERIFY_MAC",
    "MacAlgorithms": [
      "HMAC_SHA_384"
    ]
  }
}

```

```

    ],
    "MultiRegion": false,
    "Origin": "AWS_KMS"
  }
}

```

For more information, see [HMAC keys in AWS KMS](#) in the *AWS Key Management Service Developer Guide*.

Example 4: To create a multi-Region primary KMS key

The following `create-key` example creates a multi-Region primary symmetric encryption key. Because the default values for all parameters create a symmetric encryption key, only the `--multi-region` parameter is required for this KMS key. In the AWS CLI, to indicate that a Boolean parameter is true, just specify the parameter name.

```

aws kms create-key \
  --multi-region

```

Output:

```

{
  "KeyMetadata": {
    "Arn": "arn:aws:kms:us-west-2:111122223333:key/mrk-1234abcd12ab34cd56ef12345678990ab",
    "AWSAccountId": "111122223333",
    "CreationDate": "2021-09-02T016:15:21-09:00",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "Description": "",
    "Enabled": true,
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ],
    "KeyId": "mrk-1234abcd12ab34cd56ef12345678990ab",
    "KeyManager": "CUSTOMER",
    "KeySpec": "SYMMETRIC_DEFAULT",
    "KeyState": "Enabled",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "MultiRegion": true,
    "MultiRegionConfiguration": {
      "MultiRegionKeyType": "PRIMARY",
      "PrimaryKey": {

```

```

        "Arn": "arn:aws:kms:us-west-2:111122223333:key/
mrk-1234abcd12ab34cd56ef12345678990ab",
        "Region": "us-west-2"
    },
    "ReplicaKeys": []
},
"Origin": "AWS_KMS"
}
}

```

For more information, see [Asymmetric keys in AWS KMS](#) in the *AWS Key Management Service Developer Guide*.

Example 5: To create a KMS key for imported key material

The following `create-key` example creates a KMS key with no key material. When the operation is complete, you can import your own key material into the KMS key. To create this KMS key, set the `--origin` parameter to `EXTERNAL`.

```

aws kms create-key \
  --origin EXTERNAL

```

Output:

```

{
  "KeyMetadata": {
    "Arn": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "AWSAccountId": "111122223333",
    "CreationDate": "2019-12-02T07:48:55-07:00",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "Description": "",
    "Enabled": false,
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ],
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyManager": "CUSTOMER",
    "KeySpec": "SYMMETRIC_DEFAULT",
    "KeyState": "PendingImport",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "MultiRegion": false,
    "Origin": "EXTERNAL"
  }
}

```

```
}  
}
```

For more information, see [Importing key material in AWS KMS keys](#) in the *AWS Key Management Service Developer Guide*.

Example 6: To create a KMS key in an AWS CloudHSM key store

The following `create-key` example creates a KMS key in the specified AWS CloudHSM key store. The operation creates the KMS key and its metadata in AWS KMS and creates the key material in the AWS CloudHSM cluster associated with the custom key store. The `--custom-key-store-id` and `--origin` parameters are required.

```
aws kms create-key \  
  --origin AWS_CLOUDHSM \  
  --custom-key-store-id cks-1234567890abcdef0
```

Output:

```
{  
  "KeyMetadata": {  
    "Arn": "arn:aws:kms:us-  
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",  
    "AWSAccountId": "111122223333",  
    "CloudHsmClusterId": "cluster-1a23b4cdefg",  
    "CreationDate": "2019-12-02T07:48:55-07:00",  
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",  
    "CustomKeyId": "cks-1234567890abcdef0",  
    "Description": "",  
    "Enabled": true,  
    "EncryptionAlgorithms": [  
      "SYMMETRIC_DEFAULT"  
    ],  
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",  
    "KeyManager": "CUSTOMER",  
    "KeySpec": "SYMMETRIC_DEFAULT",  
    "KeyState": "Enabled",  
    "KeyUsage": "ENCRYPT_DECRYPT",  
    "MultiRegion": false,  
    "Origin": "AWS_CLOUDHSM"  
  }  
}
```


For more information, see [AWS CloudHSM key stores](#) in the *AWS Key Management Service Developer Guide*.

Example 7: To create a KMS key in an external key store

The following `create-key` example creates a KMS key in the specified external key store. The `--custom-key-store-id`, `--origin`, and `--xks-key-id` parameters are required in this command.

The `--xks-key-id` parameter specifies the ID of an existing symmetric encryption key in your external key manager. This key serves as the external key material for the KMS key. The value of the `--origin` parameter must be `EXTERNAL_KEY_STORE`. The `custom-key-store-id` parameter must identify an external key store that is connected to its external key store proxy.

```
aws kms create-key \  
  --origin EXTERNAL_KEY_STORE \  
  --custom-key-store-id cks-9876543210fedcba9 \  
  --xks-key-id bb8562717f809024
```

Output:

```
{  
  "KeyMetadata": {  
    "Arn": "arn:aws:kms:us-  
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",  
    "AWSAccountId": "111122223333",  
    "CreationDate": "2022-12-02T07:48:55-07:00",  
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",  
    "CustomKeyId": "cks-9876543210fedcba9",  
    "Description": "",  
    "Enabled": true,  
    "EncryptionAlgorithms": [  
      "SYMMETRIC_DEFAULT"  
    ],  
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",  
    "KeyManager": "CUSTOMER",  
    "KeySpec": "SYMMETRIC_DEFAULT",  
    "KeyState": "Enabled",  
    "KeyUsage": "ENCRYPT_DECRYPT",  
    "MultiRegion": false,  
    "Origin": "EXTERNAL_KEY_STORE",  
    "XksKeyConfiguration": {
```

```
        "Id": "bb8562717f809024"  
      }  
    }  
  }
```

For more information, see [External key stores](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [CreateKey](#) in *AWS CLI Command Reference*.

decrypt

The following code example shows how to use `decrypt`.

AWS CLI

Example 1: To decrypt an encrypted message with a symmetric KMS key (Linux and macOS)

The following `decrypt` command example demonstrates the recommended way to decrypt data with the AWS CLI. This version shows how to decrypt data under a symmetric KMS key.

Provide the ciphertext in a file. In the value of the `--ciphertext-blob` parameter, use the `fileb://` prefix, which tells the CLI to read the data from a binary file. If the file is not in the current directory, type the full path to file. For more information about reading AWS CLI parameter values from a file, see [Loading AWS CLI parameters from a file <https://docs.aws.amazon.com/cli/latest/userguide/cli-usage-parameters-file.html>](https://docs.aws.amazon.com/cli/latest/userguide/cli-usage-parameters-file.html) in the *AWS Command Line Interface User Guide* and [Best Practices for Local File Parameters<https://aws.amazon.com/blogs/developer/best-practices-for-local-file-parameters/>](https://aws.amazon.com/blogs/developer/best-practices-for-local-file-parameters/) in the *AWS Command Line Tool Blog*. Specify the KMS key to decrypt the ciphertext. The `--key-id` parameter is not required when decrypting with a symmetric KMS key. AWS KMS can get the key ID of the KMS key that was used to encrypt the data from the metadata in the ciphertext. But it's always a best practice to specify the KMS key you are using. This practice ensures that you use the KMS key that you intend, and prevents you from inadvertently decrypting a ciphertext using a KMS key you do not trust. Request the plaintext output as a text value. The `--query` parameter tells the CLI to get only the value of the `Plaintext` field from the output. The `--output` parameter returns the output as text. Base64-decode the plaintext and save it in a file. The following example pipes (`|`) the value of the `Plaintext` parameter to the `Base64` utility, which decodes it. Then, it redirects (`>`) the decoded output to the `ExamplePlaintext` file.

Before running this command, replace the example key ID with a valid key ID from your AWS account.

```
aws kms decrypt \  
  --ciphertext-blob fileb://ExampleEncryptedFile \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --output text \  
  --query Plaintext | base64 \  
  --decode > ExamplePlaintextFile
```

This command produces no output. The output from the decrypt command is base64-decoded and saved in a file.

For more information, see [Decrypt](#) in the *AWS Key Management Service API Reference*.

Example 2: To decrypt an encrypted message with a symmetric KMS key (Windows command prompt)

The following example is the same as the previous one except that it uses the `certutil` utility to Base64-decode the plaintext data. This procedure requires two commands, as shown in the following examples.

Before running this command, replace the example key ID with a valid key ID from your AWS account.

```
aws kms decrypt ^  
  --ciphertext-blob fileb://ExampleEncryptedFile ^  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab ^  
  --output text ^  
  --query Plaintext > ExamplePlaintextFile.base64
```

Run the `certutil` command.

```
certutil -decode ExamplePlaintextFile.base64 ExamplePlaintextFile
```

Output:

```
Input Length = 18  
Output Length = 12  
CertUtil: -decode command completed successfully.
```

For more information, see [Decrypt](#) in the *AWS Key Management Service API Reference*.

Example 3: To decrypt an encrypted message with an asymmetric KMS key (Linux and macOS)

The following decrypt command example shows how to decrypt data encrypted under an RSA asymmetric KMS key.

When using an asymmetric KMS key, the `encryption-algorithm` parameter, which specifies the algorithm used to encrypt the plaintext, is required.

Before running this command, replace the example key ID with a valid key ID from your AWS account.

```
aws kms decrypt \  
  --ciphertext-blob fileb://ExampleEncryptedFile \  
  --key-id 0987dcba-09fe-87dc-65ba-ab0987654321 \  
  --encryption-algorithm RSAES_OAEP_SHA_256 \  
  --output text \  
  --query Plaintext | base64 \  
  --decode > ExamplePlaintextFile
```

This command produces no output. The output from the decrypt command is base64-decoded and saved in a file.

For more information, see [Asymmetric keys in AWS KMS](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [Decrypt](#) in *AWS CLI Command Reference*.

delete-alias

The following code example shows how to use `delete-alias`.

AWS CLI

To delete an AWS KMS alias

The following `delete-alias` example deletes the alias `alias/example-alias`. The alias name must begin with `alias/`.

```
aws kms delete-alias \  
  --alias-name alias/example-alias
```

```
--alias-name alias/example-alias
```

This command produces no output. To find the alias, use the `list-aliases` command.

For more information, see [Deleting an alias](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [DeleteAlias](#) in *AWS CLI Command Reference*.

delete-custom-key-store

The following code example shows how to use `delete-custom-key-store`.

AWS CLI

To delete a custom key store

The following `delete-custom-key-store` example deletes the specified custom key store.

Deleting an AWS CloudHSM key store has no effect on the associated CloudHSM cluster.

Deleting an external key store has no effect on the associated external key store proxy, external key manager, or external keys.

NOTE: Before you can delete a custom key store, you must schedule the deletion of all KMS keys in the custom key store and then wait for those KMS keys to be deleted. Then, you must disconnect the custom key store. For help finding the KMS keys in your custom key store, see [Delete an AWS CloudHSM key store \(API\)](#) in the *AWS Key Management Service Developer Guide*.

```
delete-custom-key-store \  
  --custom-key-store-id cks-1234567890abcdef0
```

This command does not return any output. To verify that the custom key store is deleted, use the `describe-custom-key-stores` command.

For information about deleting an AWS CloudHSM key stores, see [Deleting an AWS CloudHSM key store](#) in the *AWS Key Management Service Developer Guide*.

For information about deleting external key stores, see [Deleting an external key store](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [DeleteCustomKeyStore](#) in *AWS CLI Command Reference*.

delete-imported-key-material

The following code example shows how to use `delete-imported-key-material`.

AWS CLI

To delete imported key material from a KMS key

The following `delete-imported-key-material` example deletes key material that had been imported into a KMS key.

```
aws kms delete-imported-key-material \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

This command produces no output. To verify that the key material is deleted, use the `describe-key` command to look for a key state of `PendingImport` or `PendingDeletion`.

For more information, see [Deleting imported key material](https://docs.aws.amazon.com/kms/latest/developerguide/importing-keys-delete-key-material.html) in the *AWS Key Management Service Developer Guide*.

- For API details, see [DeleteImportedKeyMaterial](#) in *AWS CLI Command Reference*.

describe-custom-key-stores

The following code example shows how to use `describe-custom-key-stores`.

AWS CLI

Example 1: To get details about an AWS CloudHSM key store

The following `describe-custom-key-store` example displays details about the specified AWS CloudHSM key store. The command is the same for all types of custom key stores, but the output differs with the key store type and, for an external key store, its connectivity option.

By default, this command displays information about all custom key stores in the account and Region. To display information about a particular custom key store, use the `custom-key-store-name` or `custom-key-store-id` parameter.

```
aws kms describe-custom-key-stores \  
  --custom-key-store-name ExampleCloudHSMKeyStore
```

The output of this command includes useful details about the AWS CloudHSM key store including its connection state (`ConnectionState`). If the connection state is `FAILED`, the output includes a `ConnectionErrorCode` field that describes the problem.

Output:

```
{
  "CustomKeyStores": [
    {
      "CloudHsmClusterId": "cluster-1a23b4cdefg",
      "ConnectionState": "CONNECTED",
      "CreationDate": "2022-04-05T14:04:55-07:00",
      "CustomKeyStoreId": "cks-1234567890abcdef0",
      "CustomKeyStoreName": "ExampleExternalKeyStore",
      "TrustAnchorCertificate": "<certificate appears here>"
    }
  ]
}
```

For more information, see [Viewing an AWS CloudHSM key store](#) in the *AWS Key Management Service Developer Guide*.

Example 2: To get details about an external key store with public endpoint connectivity

The following `describe-custom-key-store` example displays details about the specified external key store. The command is the same for all types of custom key stores, but the output differs with the key store type and, for an external key store, its connectivity option.

By default, this command displays information about all custom key stores in the account and Region. To display information about a particular custom key store, use the `custom-key-store-name` or `custom-key-store-id` parameter.

```
aws kms describe-custom-key-stores \
  --custom-key-store-id cks-9876543210fedcba9
```

The output of this command includes useful details about the external key store including its connection state (`ConnectionState`). If the connection state is `FAILED`, the output includes a `ConnectionErrorCode` field that describes the problem.

Output:

```
{
  "CustomKeyStores": [
    {
      "CustomKeyId": "cks-9876543210fedcba9",
      "CustomKeyName": "ExampleXKS",
      "ConnectionState": "CONNECTED",
      "CreationDate": "2022-12-02T07:48:55-07:00",
      "CustomKeyType": "EXTERNAL_KEY_STORE",
      "XksProxyConfiguration": {
        "AccessKeyId": "ABCDE12345670EXAMPLE",
        "Connectivity": "PUBLIC_ENDPOINT",
        "UriEndpoint": "https://myproxy.xks.example.com",
        "UriPath": "/example-prefix/kms/xks/v1"
      }
    }
  ]
}
```

For more information, see [Viewing an external key store](#) in the *AWS Key Management Service Developer Guide*.

Example 3: To get details about an external key store with VPC endpoint service connectivity

The following `describe-custom-key-store` example displays details about the specified external key store. The command is the same for all types of custom key stores, but the output differs with the key store type and, for an external key store, its connectivity option.

By default, this command displays information about all custom key stores in the account and Region. To display information about a particular custom key store, use the `custom-key-store-name` or `custom-key-store-id` parameter.

```
aws kms describe-custom-key-stores \
  --custom-key-store-id cks-2234567890abcdef0
```

The output of this command includes useful details about the external key store including its connection state (`ConnectionState`). If the connection state is `FAILED`, the output includes a `ConnectionErrorCode` field that describes the problem.

Output:


```
{
  "CustomKeyStores": [
    {
      "CustomKeyId": "cks-3234567890abcdef0",
      "CustomKeyName": "ExampleVPCEExternalKeyStore",
      "ConnectionState": "CONNECTED",
      "CreationDate": "2022-12-22T07:48:55-07:00",
      "CustomKeyType": "EXTERNAL_KEY_STORE",
      "XksProxyConfiguration": {
        "AccessKeyId": "ABCDE12345670EXAMPLE",
        "Connectivity": "VPC_ENDPOINT_SERVICE",
        "UriEndpoint": "https://myproxy-private.xks.example.com",
        "UriPath": "/kms/xks/v1",
        "VpcEndpointServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-
example1"
      }
    }
  ]
}
```

For more information, see [Viewing an external key store](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [DescribeCustomKeyStores](#) in *AWS CLI Command Reference*.

describe-key

The following code example shows how to use `describe-key`.

AWS CLI

Example 1: To find detailed information about a KMS key

The following `describe-key` example gets detailed information about the AWS managed key for Amazon S3 in the example account and Region. You can use this command to find details about AWS managed keys and customer managed keys.

To specify the KMS key, use the `key-id` parameter. This example uses an alias name value, but you can use a key ID, key ARN, alias name, or alias ARN in this command.

```
aws kms describe-key \
  --key-id alias/aws/s3
```

Output:

```
{
  "KeyMetadata": {
    "AWSAccountId": "846764612917",
    "KeyId": "b8a9477d-836c-491f-857e-07937918959b",
    "Arn": "arn:aws:kms:us-west-2:846764612917:key/
b8a9477d-836c-491f-857e-07937918959b",
    "CreationDate": 2017-06-30T21:44:32.140000+00:00,
    "Enabled": true,
    "Description": "Default KMS key that protects my S3 objects when no other
key is defined",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "KeyState": "Enabled",
    "Origin": "AWS_KMS",
    "KeyManager": "AWS",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ]
  }
}
```

For more information, see [Viewing keys](#) in the *AWS Key Management Service Developer Guide*.

Example 2: To get details about an RSA asymmetric KMS key

The following describe-key example gets detailed information about an asymmetric RSA KMS key used for signing and verification.

```
aws kms describe-key \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Output:

```
{
  "KeyMetadata": {
    "AWSAccountId": "111122223333",
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "Arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "CreationDate": "2019-12-02T19:47:14.861000+00:00",
    "CustomerMasterKeySpec": "RSA_2048",
```

```

    "Enabled": false,
    "Description": "",
    "KeyState": "Disabled",
    "Origin": "AWS_KMS",
    "MultiRegion": false,
    "KeyManager": "CUSTOMER",
    "KeySpec": "RSA_2048",
    "KeyUsage": "SIGN_VERIFY",
    "SigningAlgorithms": [
      "RSASSA_PKCS1_V1_5_SHA_256",
      "RSASSA_PKCS1_V1_5_SHA_384",
      "RSASSA_PKCS1_V1_5_SHA_512",
      "RSASSA_PSS_SHA_256",
      "RSASSA_PSS_SHA_384",
      "RSASSA_PSS_SHA_512"
    ]
  }
}

```

Example 3: To get details about a multi-Region replica key

The following describe-key example gets metadata for a multi-Region replica key. This multi-Region key is a symmetric encryption key. The output of a describe-key command for any multi-Region key returns information about the primary key and all of its replicas.

```

aws kms describe-key \
  --key-id arn:aws:kms:ap-northeast-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab

```

Output:

```

{
  "KeyMetadata": {
    "MultiRegion": true,
    "AWSAccountId": "111122223333",
    "Arn": "arn:aws:kms:ap-northeast-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
    "CreationDate": "2021-06-28T21:09:16.114000+00:00",
    "Description": "",
    "Enabled": true,
    "KeyId": "mrk-1234abcd12ab34cd56ef1234567890ab",
    "KeyManager": "CUSTOMER",

```

```

    "KeyState": "Enabled",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "Origin": "AWS_KMS",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ],
    "MultiRegionConfiguration": {
      "MultiRegionKeyType": "PRIMARY",
      "PrimaryKey": {
        "Arn": "arn:aws:kms:us-west-2:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
        "Region": "us-west-2"
      },
      "ReplicaKeys": [
        {
          "Arn": "arn:aws:kms:eu-west-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
          "Region": "eu-west-1"
        },
        {
          "Arn": "arn:aws:kms:ap-northeast-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
          "Region": "ap-northeast-1"
        },
        {
          "Arn": "arn:aws:kms:sa-east-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
          "Region": "sa-east-1"
        }
      ]
    }
  }
}

```

Example 4: To get details about an HMAC KMS key

The following `describe-key` example gets detailed information about an HMAC KMS key.

```

aws kms describe-key \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab

```

Output:

```
{
  "KeyMetadata": {
    "AWSAccountId": "123456789012",
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "Arn": "arn:aws:kms:us-
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "CreationDate": "2022-04-03T22:23:10.194000+00:00",
    "Enabled": true,
    "Description": "Test key",
    "KeyUsage": "GENERATE_VERIFY_MAC",
    "KeyState": "Enabled",
    "Origin": "AWS_KMS",
    "KeyManager": "CUSTOMER",
    "CustomerMasterKeySpec": "HMAC_256",
    "MacAlgorithms": [
      "HMAC_SHA_256"
    ],
    "MultiRegion": false
  }
}
```

- For API details, see [DescribeKey](#) in *AWS CLI Command Reference*.

disable-key-rotation

The following code example shows how to use `disable-key-rotation`.

AWS CLI

To disable automatic rotation of a KMS key

The following `disable-key-rotation` example disables automatic rotation of a customer managed KMS key. To reenable automatic rotation, use the `enable-key-rotation` command.

```
aws kms disable-key-rotation \
  --key-id arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab
```

This command produces no output. To verify that automatic rotation is disabled for the KMS key, use the `get-key-rotation-status` command.

For more information, see [Rotating keys](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [DisableKeyRotation](#) in *AWS CLI Command Reference*.

disable-key

The following code example shows how to use `disable-key`.

AWS CLI

To temporarily disable a KMS key

The following example uses the `disable-key` command to disable a customer managed KMS key. To re-enable the KMS key, use the `enable-key` command.

```
aws kms disable-key \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

This command produces no output.

For more information, see [Enabling and Disabling Keys](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [DisableKey](#) in *AWS CLI Command Reference*.

disconnect-custom-key-store

The following code example shows how to use `disconnect-custom-key-store`.

AWS CLI

To disconnect a custom key store

The following `disconnect-custom-key-store` example disconnects a custom key store from its AWS CloudHSM cluster. You might disconnect a key store to troubleshoot a problem, to update its settings, or to prevent KMS keys in the keystore from being used in cryptographic operations.

This command is the same for all custom key stores, including AWS CloudHSM key stores and external key stores.

Before running this command, replace the example custom key store ID with a valid one.

```
$ aws kms disconnect-custom-key-store \  
  --custom-key-store-id cks-1234567890abcdef0
```

This command produces no output. verify that the command was effective, use the `describe-custom-key-stores` command.

For more information about disconnecting an AWS CloudHSM key store, see [Connecting and disconnecting an AWS CloudHSM key store](#) in the *AWS Key Management Service Developer Guide*.

For more information about disconnecting an external key store, see [Connecting and disconnecting an external key store](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [DisconnectCustomKeyStore](#) in *AWS CLI Command Reference*.

enable-key-rotation

The following code example shows how to use `enable-key-rotation`.

AWS CLI

To enable automatic rotation of a KMS key

The following `enable-key-rotation` example enables automatic rotation of a customer managed KMS key with a rotation period of 180 days. The KMS key will be rotated one year (approximate 365 days) from the date that this command completes and every year thereafter.

The `--key-id` parameter identifies the KMS key. This example uses a key ARN value, but you can use either the key ID or the ARN of the KMS key. The `--rotation-period-in-days` parameter specifies the number of days between each rotation date. Specify a value between 90 and 2560 days. If no value is specified, the default value is 365 days.

```
aws kms enable-key-rotation \  
  --key-id arn:aws:kms:us-  
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab \  
  --rotation-period-in-days 180
```

This command produces no output. To verify that the KMS key is enabled, use the `get-key-rotation-status` command.

For more information, see [Rotating keys](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [EnableKeyRotation](#) in *AWS CLI Command Reference*.

enable-key

The following code example shows how to use `enable-key`.

AWS CLI

To enable a KMS key

The following `enable-key` example enables a customer managed key. You can use a command like this one to enable a KMS key that you temporarily disabled by using the `disable-key` command. You can also use it to enable a KMS key that is disabled because it was scheduled for deletion and the deletion was canceled.

To specify the KMS key, use the `key-id` parameter. This example uses an key ID value, but you can use a key ID or key ARN value in this command.

Before running this command, replace the example key ID with a valid one.

```
aws kms enable-key \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

This command produces no output. To verify that the KMS key is enabled, use the `describe-key` command. See the values of the `KeyState` and `Enabled` fields in the `describe-key` output.

For more information, see [Enabling and Disabling Keys](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [EnableKey](#) in *AWS CLI Command Reference*.

encrypt

The following code example shows how to use `encrypt`.

AWS CLI

Example 1: To encrypt the contents of a file on Linux or MacOS

The following `encrypt` command demonstrates the recommended way to encrypt data with the AWS CLI.

```
aws kms encrypt \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --plaintext fileb://ExamplePlaintextFile \  
  --output text \  
  --query CiphertextBlob | base64 \  
  --decode > ExampleEncryptedFile
```

The command does several things:

Uses the `--plaintext` parameter to indicate the data to encrypt. This parameter value must be base64-encoded. The value of the `plaintext` parameter must be base64-encoded, or you must use the `fileb://` prefix, which tells the AWS CLI to read binary data from the file. If the file is not in the current directory, type the full path to file. For example: `fileb:///var/tmp/ExamplePlaintextFile` or `fileb://C:\Temp\ExamplePlaintextFile`. For more information about reading AWS CLI parameter values from a file, see [Loading Parameters from a File](#) in the *AWS Command Line Interface User Guide* and [Best Practices for Local File Parameters](#) on the AWS Command Line Tool Blog. Uses the `--output` and `--query` parameters to control the command's output. These parameters extract the encrypted data, called the *ciphertext*, from the command's output. For more information about controlling output, see [Controlling Command Output](#) in the *AWS Command Line Interface User Guide*. Uses the `base64` utility to decode the extracted output into binary data. The ciphertext that is returned by a successful `encrypt` command is base64-encoded text. You must decode this text before you can use the AWS CLI to decrypt it. Saves the binary ciphertext to a file. The final part of the command (`> ExampleEncryptedFile`) saves the binary ciphertext to a file to make decryption easier. For an example command that uses the AWS CLI to decrypt data, see the [decrypt examples](#).

Example 2: Using the AWS CLI to encrypt data on Windows

This example is the same as the previous one, except that it uses the `certutil` tool instead of `base64`. This procedure requires two commands, as shown in the following example.

```
aws kms encrypt \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --plaintext fileb://ExamplePlaintextFile \  
  --output text \  
  --query CiphertextBlob > C:\Temp\ExampleEncryptedFile.base64
```

```
certutil -decode C:\Temp\ExampleEncryptedFile.base64 C:\Temp\ExampleEncryptedFile
```

Example 3: Encrypting with an asymmetric KMS key

The following `encrypt` command shows how to encrypt plaintext with an asymmetric KMS key. The `--encryption-algorithm` parameter is required. As in all `encrypt` CLI commands, the `plaintext` parameter must be base64-encoded, or you must use the `fileb://` prefix, which tells the AWS CLI to read binary data from the file.

```
aws kms encrypt \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --encryption-algorithm RSAES_OAEP_SHA_256 \  
  --plaintext fileb://ExamplePlaintextFile \  
  --output text \  
  --query CiphertextBlob | base64 \  
  --decode > ExampleEncryptedFile
```

This command produces no output.

- For API details, see [Encrypt](#) in *AWS CLI Command Reference*.

generate-data-key-pair-without-plaintext

The following code example shows how to use `generate-data-key-pair-without-plaintext`.

AWS CLI

To generate an ECC NIST P384 asymmetric data key pair

The following `generate-data-key-pair-without-plaintext` example requests an ECC NIST P384 key pair for use outside of AWS.

The command returns a plaintext public key and a copy of the private key encrypted under the specified KMS key. It does not return a plaintext private key. You can safely store the encrypted private key with the encrypted data, and call AWS KMS to decrypt the private key when you need to use it.

To request an ECC NIST P384 asymmetric data key pair, use the `key-pair-spec` parameter with a value of `ECC_NIST_P384`.

The KMS key you specify must be a symmetric encryption KMS key, that is, a KMS key with a `KeySpec` value of `SYMMETRIC_DEFAULT`.

NOTE: The values in the output of this example are truncated for display.

```
aws kms generate-data-key-pair-without-plaintext \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --key-pair-spec ECC_NIST_P384
```

Output:

```
{  
  "PrivateKeyCiphertextBlob": "AQIDAHi6LtupRpdK12aJTzkK6Fbh0tQkM1QJJH3PdtHvS/y  
+hAFFxmiD134doUDzMGmfCEtcAAAHaTCCB2UGCSqGSIb3DQEHbqCCB1...",  
  "PublicKey":  
  "MIIBojANBgkqhkiG9w0BAQEFAAOCAY8AMIIBigKCAYEA3A3eGMyPrvSn7+Ld1JE1oUoQV5HpEuHAVbd0yND  
+NmYDH/mL10SIEuLrcdZ5hrMH4pk83r401...",  
  "KeyId": "arn:aws:kms:us-  
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",  
  "KeySpec": "ECC_NIST_P384"  
}
```

The `PublicKey` and `PrivateKeyCiphertextBlob` are returned in base64-encoded format.

For more information, see [Data key pairs](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [GenerateDataKeyPairWithoutPlaintext](#) in *AWS CLI Command Reference*.

generate-data-key-pair

The following code example shows how to use `generate-data-key-pair`.

AWS CLI

To generate an 2048-bit RSA asymmetric data key pair

The following `generate-data-key-pair` example requests a 2048-bit RSA asymmetric data key pair for use outside of AWS. The command returns a plaintext public key and a plaintext private key for immediate use and deletion, and a copy of the private key encrypted under the specified KMS key. You can safely store the encrypted private key with the encrypted data.

To request a 2048-bit RSA asymmetric data key pair, use the `key-pair-spec` parameter with a value of `RSA_2048`.

The KMS key you specify must be a symmetric encryption KMS key, that is, a KMS key with a `KeySpec` value of `SYMMETRIC_DEFAULT`.

NOTE: The values in the output of this example are truncated for display.

```
aws kms generate-data-key-pair \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --key-pair-spec RSA_2048
```

Output:

```
{  
  "PrivateKeyCiphertextBlob": "AQIDAHi6LtupRpdK12aJTzkK6Fbh0tQkM1QJJH3PdtHvS/y  
+hAFFxmiD134doUDzMGmfCEtcAAAHaTCCB2UGCSqGSIB3DQEHBqCCB1...",  
  "PrivateKeyPlaintext": "MIIG/  
QIBADANBgkqhkiG9w0BAQEFAASCBUcwggbjAgEAAoIBgQDcDd4YzI  
+u9Kfv4t2UkTWhShBXkekS4cBVt07I0P42ZgMf+YvU5IgS4ut...",  
  "PublicKey":  
  "MIIB0jANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBigKCAYEA3A3eGMyPrvSn7+Ld1JE1oUoQV5HpEuHAVbd0yND  
+NmYDH/mL10SIEuLrcdZ5hrMH4pk83r40l...",  
  "KeyId": "arn:aws:kms:us-  
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",  
  "KeySpec": "RSA_2048"  
}
```

The `PublicKey`, `PrivateKeyPlaintext`, and `PrivateKeyCiphertextBlob` are returned in base64-encoded format.

For more information, see [Data key pairs](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [GenerateDataKeyPair](#) in *AWS CLI Command Reference*.

generate-data-key-without-plaintext

The following code example shows how to use `generate-data-key-without-plaintext`.

AWS CLI

To generate a 256-bit symmetric data key without a plaintext key

The following `generate-data-key-without-plaintext` example requests an encrypted copy of a 256-bit symmetric data key for use outside of AWS. You can call AWS KMS to decrypt the data key when you are ready to use it.

To request a 256-bit data key, use the `key-spec` parameter with a value of `AES_256`. To request a 128-bit data key, use the `key-spec` parameter with a value of `AES_128`. For all other data key lengths, use the `number-of-bytes` parameter.

The KMS key you specify must be a symmetric encryption KMS key, that is, a KMS key with a `key-spec` value of `SYMMETRIC_DEFAULT`.

```
aws kms generate-data-key-without-plaintext \
  --key-id "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab" \
  --key-spec AES_256
```

Output:

```
{
  "CiphertextBlob":
  "AQEDAHjRYf5WytIc0C857tFSnBaPn2F8DgfmThbJlGfR8P3WlwAAAH4wfAYJKoZIHvcNAQcGoG8wbQIBADBoBgkqhki
  "KeyId": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
}
```

The `CiphertextBlob` (encrypted data key) is returned in base64-encoded format.

For more information, see [Data keys](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [GenerateDataKeyWithoutPlaintext](#) in *AWS CLI Command Reference*.

generate-data-key

The following code example shows how to use `generate-data-key`.

AWS CLI

Example 1: To generate a 256-bit symmetric data key

The following `generate-data-key` example requests a 256-bit symmetric data key for use outside of AWS. The command returns a plaintext data key for immediate use and deletion,

and a copy of that data key encrypted under the specified KMS key. You can safely store the encrypted data key with the encrypted data.

To request a 256-bit data key, use the `key-spec` parameter with a value of `AES_256`. To request a 128-bit data key, use the `key-spec` parameter with a value of `AES_128`. For all other data key lengths, use the `number-of-bytes` parameter.

The KMS key you specify must be a symmetric encryption KMS key, that is, a KMS key with a `key-spec` value of `SYMMETRIC_DEFAULT`.

```
aws kms generate-data-key \  
  --key-id alias/ExampleAlias \  
  --key-spec AES_256
```

Output:

```
{  
  "Plaintext": "VdzKNHGzUAzJeRBVY+uUmofUGGiDzyB3+i9fVkh3piw=",  
  "KeyId": "arn:aws:kms:us-  
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",  
  "CiphertextBlob":  
  "AQEDAHjRYf5WytIc0C857tFSnBaPn2F8DgfmThbJlGfR8P3WlwAAAH4wfAYJKoZIHvcNAQcGoG8wbQIBADBoBgkqhki  
+YdhV8M1rkBQPeac0ReRVNDt9q1eAt+SHgIRF8P0H+7U="  
}
```

The `Plaintext` (plaintext data key) and the `CiphertextBlob` (encrypted data key) are returned in base64-encoded format.

For more information, see [Data keys](https://docs.aws.amazon.com/kms/latest/developerguide/concepts.html#data-keys) <<https://docs.aws.amazon.com/kms/latest/developerguide/concepts.html#data-keys>> in the *AWS Key Management Service Developer Guide*.

Example 2: To generate a 512-bit symmetric data key

The following `generate-data-key` example requests a 512-bit symmetric data key for encryption and decryption. The command returns a plaintext data key for immediate use and deletion, and a copy of that data key encrypted under the specified KMS key. You can safely store the encrypted data key with the encrypted data.

To request a key length other than 128 or 256 bits, use the `number-of-bytes` parameter. To request a 512-bit data key, the following example uses the `number-of-bytes` parameter with a value of 64 (bytes).

The KMS key you specify must be a symmetric encryption KMS key, that is, a KMS key with a key spec value of `SYMMETRIC_DEFAULT`.

NOTE: The values in the output of this example are truncated for display.

```
aws kms generate-data-key \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --number-of-bytes 64
```

Output:

```
{  
  "CiphertextBlob": "AQIBAHi6LtupRpdK12aJTzkK6Fbh0tQkM1QJJH3PdtHvS/y+hAEnX/  
QQNmMwDfg2koιNMEc8AAACaDCCAmQGCSqGSiB3DQEHBqCCA1UwggJRAgEAMIICSgYJKoZ...",  
  "Plaintext": "ty8Lr0Bk60F07M2Bwt6qbFdNB  
+G00ZLtf5MSEb4a13R2UKWG0p06njAwy2n72VRm2m7z/  
Pm9Wpbvttz6a4lSo9hgPvKhZ5y6RTm40ovEXiVfBveyX3DQxDzRSwbKDPk/...",  
  "KeyId": "arn:aws:kms:us-  
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"  
}
```

The `Plaintext` (plaintext data key) and `CiphertextBlob` (encrypted data key) are returned in base64-encoded format.

For more information, see [Data keys <https://docs.aws.amazon.com/kms/latest/developerguide/concepts.html#data-keys>](https://docs.aws.amazon.com/kms/latest/developerguide/concepts.html#data-keys) in the *AWS Key Management Service Developer Guide*.

- For API details, see [GenerateDataKey](#) in *AWS CLI Command Reference*.

generate-random

The following code example shows how to use `generate-random`.

AWS CLI

Example 1: To generate a 256-bit random byte string (Linux or macOS)

The following `generate-random` example generates a 256-bit (32-byte), base64-encoded random byte string. The example decodes the byte string and saves it in the `random` file.

When you run this command, you must use the `number-of-bytes` parameter to specify the length of the random value in bytes.

You don't specify a KMS key when you run this command. The random byte string is unrelated to any KMS key.

By default, AWS KMS generates the random number. However, if you specify a custom key store<<https://docs.aws.amazon.com/kms/latest/developerguide/custom-key-store-overview.html>>, the random byte string is generated in the AWS CloudHSM cluster associated with the custom key store.

This example uses the following parameters and values:

It uses the required `--number-of-bytes` parameter with a value of 32 to request a 32-byte (256-bit) string. It uses the `--output` parameter with a value of `text` to direct the AWS CLI to return the output as text, instead of JSON. It uses the `--query` parameter to extract the value of the `Plaintext` property from the response. It pipes (`|`) the output of the command to the `base64` utility, which decodes the extracted output. It uses the redirection operator (`>`) to save decoded byte string to the `ExampleRandom` file. It uses the redirection operator (`>`) to save the binary ciphertext to a file.

```
aws kms generate-random \  
  --number-of-bytes 32 \  
  --output text \  
  --query Plaintext | base64 --decode > ExampleRandom
```

This command produces no output.

For more information, see [GenerateRandom](#) in the *AWS Key Management Service API Reference*.

Example 2: To generate a 256-bit random number (Windows Command Prompt)

The following example uses the `generate-random` command to generate a 256-bit (32-byte), base64-encoded random byte string. The example decodes the byte string and saves it in the `random` file. This example is the same as the previous example, except that it uses the `certutil` utility in Windows to base64-decode the random byte string before saving it in a file.

First, generate a base64-encoded random byte string and saves it in a temporary file, `ExampleRandom.base64`.

```
aws kms generate-random \  
  --number-of-bytes 32 \  
  --output text \  
  --query Plaintext > ExampleRandom.base64
```


Because the output of the `generate-random` command is saved in a file, this example produces no output.

Now use the `certutil -decode` command to decode the base64-encoded byte string in the `ExampleRandom.base64` file. Then, it saves the decoded byte string in the `ExampleRandom` file.

```
certutil -decode ExampleRandom.base64 ExampleRandom
```

Output:

```
Input Length = 18
Output Length = 12
CertUtil: -decode command completed successfully.
```

For more information, see [GenerateRandom](#) in the *AWS Key Management Service API Reference*.

- For API details, see [GenerateRandom](#) in *AWS CLI Command Reference*.

get-key-policy

The following code example shows how to use `get-key-policy`.

AWS CLI

To copy a key policy from one KMS key to another KMS key

The following `get-key-policy` example gets the key policy from one KMS key and saves it in a text file. Then, it replaces the policy of a different KMS key using the text file as the policy input.

Because the `--policy` parameter of `put-key-policy` requires a string, you must use the `--output text` option to return the output as a text string instead of JSON.

```
aws kms get-key-policy \
  --policy-name default \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --query Policy \
  --output text > policy.txt

aws kms put-key-policy \
  --policy-name default \
```

```
--key-id 0987dcba-09fe-87dc-65ba-ab0987654321 \  
--policy file://policy.txt
```

This command produces no output.

For more information, see [PutKeyPolicy](#) in the *AWS KMS API Reference*.

- For API details, see [GetKeyPolicy](#) in *AWS CLI Command Reference*.

get-key-rotation-status

The following code example shows how to use `get-key-rotation-status`.

AWS CLI

To retrieve the rotation status for a KMS key.

The following `get-key-rotation-status` example returns information about the rotation status of the specified KMS key, including whether automatic rotation is enabled, the rotation period, and the next scheduled rotation date. You can use this command on customer managed KMS keys and AWS managed KMS keys. However, all AWS managed KMS keys are automatically rotated every year.

```
aws kms get-key-rotation-status \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Output:

```
{  
  "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",  
  "KeyRotationEnabled": true,  
  "NextRotationDate": "2024-02-14T18:14:33.587000+00:00",  
  "RotationPeriodInDays": 365  
}
```

For more information, see [Rotating keys](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [GetKeyRotationStatus](#) in *AWS CLI Command Reference*.

get-parameters-for-import

The following code example shows how to use `get-parameters-for-import`.

AWS CLI

To get the items required to import key material into a KMS key

The following `get-parameters-for-import` example gets the public key and import token that you need to import key material into a KMS key. When you use the `import-key-material` command, be sure to use the import token and key material encrypted by the public key that were returned in the same `get-parameters-for-import` command. Also, the wrapping algorithm that you specify in this command must be one that you use to encrypt the key material with the public key.

To specify the KMS key, use the `key-id` parameter. This example uses an key ID, but you can use a key ID or key ARN in this command.

```
aws kms get-parameters-for-import \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --wrapping-algorithm RSAES_OAEP_SHA_256 \
  --wrapping-key-spec RSA_2048
```

Output:

```
{
  "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "PublicKey": "<public key base64 encoded data>",
  "ImportToken": "<import token base64 encoded data>",
  "ParametersValidTo": 1593893322.32
}
```

For more information, see [Download the public key and import token](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [GetParametersForImport](#) in *AWS CLI Command Reference*.

get-public-key

The following code example shows how to use `get-public-key`.

AWS CLI

Example 1: To download the public key of an asymmetric KMS key

The following `get-public-key` example downloads the public key of an asymmetric KMS key.

In addition to returning the public key, the output includes information that you need to use the public key safely outside of AWS KMS, including the key usage and supported encryption algorithms.

```
aws kms get-public-key \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Output:

```
{
  "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "PublicKey": "jANBgkqhkiG9w0BAQEFAA0CAg8AMIICCgKCAgEAl5epvg1/
QtJhxSi2g9SDEVg8QV/...",
  "CustomerMasterKeySpec": "RSA_4096",
  "KeyUsage": "ENCRYPT_DECRYPT",
  "EncryptionAlgorithms": [
    "RSAES_OAEP_SHA_1",
    "RSAES_OAEP_SHA_256"
  ]
}
```

For more information about using asymmetric KMS keys in AWS KMS, see [Using Symmetric and Asymmetric Keys](#) in the *AWS Key Management Service API Reference*.

Example 2: To convert a public key to DER format (Linux and macOS)

The following `get-public-key` example downloads the public key of an asymmetric KMS key and saves it in a DER file.

When you use the `get-public-key` command in the AWS CLI, it returns a DER-encoded X.509 public key that is Base64-encoded. This example gets the value of the `PublicKey` property as text. It Base64-decodes the `PublicKey` and saves it in the `public_key.der` file. The output parameter returns the output as text, instead of JSON. The `--query` parameter gets only the `PublicKey` property, not the properties that you need to use the public key safely outside of AWS KMS.

Before running this command, replace the example key ID with a valid key ID from your AWS account.

```
aws kms get-public-key \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --output text \  
  --query PublicKey | base64 --decode > public_key.der
```

This command produces no output.

For more information about using asymmetric KMS keys in AWS KMS, see [Using Symmetric and Asymmetric Keys](#) in the *AWS Key Management Service API Reference*.

- For API details, see [GetPublicKey](#) in *AWS CLI Command Reference*.

import-key-material

The following code example shows how to use `import-key-material`.

AWS CLI

To import key material into a KMS key

The following `import-key-material` example uploads key material into a KMS key that was created with no key material. The key state of the KMS key must be `PendingImport`.

This command uses key material that you encrypted with the public key that the `get-parameters-for-import` command returned. It also uses the import token from the same `get-parameters-for-import` command.

The `expiration-model` parameter indicates that the key material automatically expires on the date and time specified by the `valid-to` parameter. When the key material expires, AWS KMS deletes the key material, the key state of the KMS key changes to `Pending import` and the KMS key becomes unusable. To restore the KMS key, you must reimport the same key material. To use different key material, you must create a new KMS key.

Before running this command, replace the example key ID with a valid key ID or key ARN from your AWS account.

```
aws kms import-key-material \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --encrypted-key-material fileb://EncryptedKeyMaterial.bin \  
  --import-token fileb://ImportToken.bin \  
  --expiration-model KEY_MATERIAL_EXPIRES \  
  --valid-to 2020-01-01T00:00:00Z
```

```
--valid-to 2021-09-21T19:00:00Z
```

This command produces no output.

For more information about importing key material, see [Importing Key Material](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [ImportKeyMaterial](#) in *AWS CLI Command Reference*.

list-aliases

The following code example shows how to use `list-aliases`.

AWS CLI

Example 1: To list all aliases in an AWS account and Region

The following example uses the `list-aliases` command to list all aliases in the default Region of the AWS account. The output includes aliases associated with AWS managed KMS keys and customer managed KMS keys.

```
aws kms list-aliases
```

Output:

```
{
  "Aliases": [
    {
      "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/testKey",
      "AliasName": "alias/testKey",
      "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    {
      "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/FinanceDept",
      "AliasName": "alias/FinanceDept",
      "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321"
    },
    {
      "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/aws/dynamodb",
      "AliasName": "alias/aws/dynamodb",
      "TargetKeyId": "1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d"
    },
    {
```

```

        "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/aws/ebs",
        "AliasName": "alias/aws/ebs",
        "TargetKeyId": "0987ab65-43cd-21ef-09ab-87654321cdef"
    },
    ...
]
}

```

Example 2: To list all aliases for a particular KMS key

The following example uses the `list-aliases` command and its `key-id` parameter to list all aliases that are associated with a particular KMS key.

Each alias is associated with only one KMS key, but a KMS key can have multiple aliases. This command is very useful because the AWS KMS console lists only one alias for each KMS key. To find all aliases for a KMS key, you must use the `list-aliases` command.

This example uses the key ID of the KMS key for the `--key-id` parameter, but you can use a key ID, key ARN, alias name, or alias ARN in this command.

```
aws kms list-aliases --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Output:

```

{
  "Aliases": [
    {
      "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
      "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/oregon-test-key",
      "AliasName": "alias/oregon-test-key"
    },
    {
      "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
      "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/project121-test",
      "AliasName": "alias/project121-test"
    }
  ]
}

```

For more information, see [Working with Aliases](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [ListAliases](#) in *AWS CLI Command Reference*.

list-grants

The following code example shows how to use `list-grants`.

AWS CLI

To view the grants on an AWS KMS key

The following `list-grants` example displays all of the grants on the specified AWS managed KMS key for Amazon DynamoDB in your account. This grant allows DynamoDB to use the KMS key on your behalf to encrypt a DynamoDB table before writing it to disk. You can use a command like this one to view the grants on the AWS managed KMS keys and customer managed KMS keys in the AWS account and Region.

This command uses the `key-id` parameter with a key ID to identify the KMS key. You can use a key ID or key ARN to identify the KMS key. To get the key ID or key ARN of an AWS managed KMS key, use the `list-keys` or `list-aliases` command.

```
aws kms list-grants \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

The output shows that the grant gives Amazon DynamoDB permission to use the KMS key for cryptographic operations, and gives it permission to view details about the KMS key (`DescribeKey`) and to retire grants (`RetireGrant`). The `EncryptionContextSubset` constraint limits these permission to requests that include the specified encryption context pairs. As a result, the permissions in the grant are effective only on specified account and DynamoDB table.

```
{
  "Grants": [
    {
      "Constraints": {
        "EncryptionContextSubset": {
          "aws:dynamodb:subscriberId": "123456789012",
          "aws:dynamodb:tableName": "Services"
        }
      },
      "IssuingAccount": "arn:aws:iam::123456789012:root",
      "Name": "8276b9a6-6cf0-46f1-b2f0-7993a7f8c89a",
      "Operations": [
        "Decrypt",
```



```

        "Encrypt",
        "GenerateDataKey",
        "ReEncryptFrom",
        "ReEncryptTo",
        "RetireGrant",
        "DescribeKey"
    ],
    "GrantId":
    "1667b97d27cf748cf05b487217dd4179526c949d14fb3903858e25193253fe59",
    "KeyId": "arn:aws:kms:us-
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "RetiringPrincipal": "dynamodb.us-west-2.amazonaws.com",
    "GranteePrincipal": "dynamodb.us-west-2.amazonaws.com",
    "CreationDate": "2021-05-13T18:32:45.144000+00:00"
    }
]
}

```

For more information, see [Grants in AWS KMS](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [ListGrants](#) in *AWS CLI Command Reference*.

list-key-policies

The following code example shows how to use `list-key-policies`.

AWS CLI

To get the names of key policies for a KMS key

The following `list-key-policies` example gets the names of the key policies for a customer managed key in the example account and Region. You can use this command to find the names of key policies for AWS managed keys and customer managed keys.

Because the only valid key policy name is `default`, this command is not useful.

To specify the KMS key, use the `key-id` parameter. This example uses a key ID value, but you can use a key ID or key ARN in this command.

```

aws kms list-key-policies \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab

```

Output:

```
{
  "PolicyNames": [
    "default"
  ]
}
```

For more information about AWS KMS key policies, see [Using Key Policies in AWS KMS](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [ListKeyPolicies](#) in *AWS CLI Command Reference*.

list-key-rotations

The following code example shows how to use `list-key-rotations`.

AWS CLI**To retrieve information about all completed key material rotations**

The following `list-key-rotations` example lists information about all completed key material rotations for the specified KMS key.

```
aws kms list-key-rotations \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Output:

```
{
  "Rotations": [
    {
      "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
      "RotationDate": "2024-03-02T10:11:36.564000+00:00",
      "RotationType": "AUTOMATIC"
    },
    {
      "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
      "RotationDate": "2024-04-05T15:14:47.757000+00:00",
      "RotationType": "ON_DEMAND"
    }
  ]
}
```

```
  ],
  "Truncated": false
}
```

For more information, see [Rotating keys](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [ListKeyRotations](#) in *AWS CLI Command Reference*.

list-keys

The following code example shows how to use `list-keys`.

AWS CLI

To get the KMS keys in an account and Region

The following `list-keys` example gets the KMS keys in an account and Region. This command returns both AWS managed keys and customer managed keys.

```
aws kms list-keys
```

Output:

```
{
  "Keys": [
    {
      "KeyArn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    {
      "KeyArn": "arn:aws:kms:us-
west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321",
      "KeyId": "0987dcba-09fe-87dc-65ba-ab0987654321"
    },
    {
      "KeyArn": "arn:aws:kms:us-
east-2:111122223333:key/1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d",
      "KeyId": "1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d"
    }
  ]
}
```

For more information, see [Viewing Keys](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [ListKeys](#) in *AWS CLI Command Reference*.

list-resource-tags

The following code example shows how to use `list-resource-tags`.

AWS CLI

To get the tags on a KMS key

The following `list-resource-tags` example gets the tags for a KMS key. To add or replace resource tags on KMS keys, use the `tag-resource` command. The output shows that this KMS key has two resource tags, each of which has a key and value.

To specify the KMS key, use the `key-id` parameter. This example uses a key ID value, but you can use a key ID or key ARN in this command.

```
aws kms list-resource-tags \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Output:

```
{
  "Tags": [
    {
      "TagKey": "Dept",
      "TagValue": "IT"
    },
    {
      "TagKey": "Purpose",
      "TagValue": "Test"
    }
  ],
  "Truncated": false
}
```

For more information about using tags in AWS KMS, see [Tagging keys](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [ListResourceTags](#) in *AWS CLI Command Reference*.

list-retirable-grants

The following code example shows how to use `list-retirable-grants`.

AWS CLI

To view the grants that a principal can retire

The following `list-retirable-grants` example displays all of the grants that the `ExampleAdmin` user can retire on the KMS keys in an AWS account and Region. You can use a command like this one to view the grants that any account principal can retire on KMS keys in the AWS account and Region.

The value of the required `retiring-principal` parameter must be the Amazon Resource Name (ARN) of an account, user, or role.

You cannot specify a service for the value of `retiring-principal` in this command, even though a service can be the retiring principal. To find the grants in which a particular service is the retiring principal, use the `list-grants` command.

The output shows that `ExampleAdmin` user has permission to retire grants on two different KMS keys in the account and region. In addition to the retiring principal, the account has permission to retire any grant in the account.

```
aws kms list-retirable-grants \  
  --retiring-principal arn:aws:iam::111122223333:user/ExampleAdmin
```

Output:

```
{  
  "Grants": [  
    {  
      "KeyId": "arn:aws:kms:us-  
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",  
      "GrantId":  
"156b69c63cb154aa21f59929fff19760717be8d9d82b99df53e18b94a15a5e88e",  
      "Name": "",  
      "CreationDate": 2021-01-14T20:17:36.419000+00:00,  
      "GranteePrincipal": "arn:aws:iam::111122223333:user/ExampleUser",  
      "RetiringPrincipal": "arn:aws:iam::111122223333:user/ExampleAdmin",  
      "IssuingAccount": "arn:aws:iam::111122223333:root",  
      "Operations": [  

```

```

        "Encrypt"
    ],
    "Constraints": {
        "EncryptionContextSubset": {
            "Department": "IT"
        }
    }
},
{
    "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321",
    "GrantId":
"8c94d1f12f5e69f440bae30eaec9570bb1fb7358824f9ddfa1aa5a0dab1a59b2",
    "Name": "",
    "CreationDate": "2021-02-02T19:49:49.638000+00:00",
    "GranteePrincipal": "arn:aws:iam::111122223333:role/ExampleRole",
    "RetiringPrincipal": "arn:aws:iam::111122223333:user/ExampleAdmin",
    "IssuingAccount": "arn:aws:iam::111122223333:root",
    "Operations": [
        "Decrypt"
    ],
    "Constraints": {
        "EncryptionContextSubset": {
            "Department": "IT"
        }
    }
}
],
"Truncated": false
}

```

For more information, see [Grants in AWS KMS](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [ListRetirableGrants](#) in *AWS CLI Command Reference*.

put-key-policy

The following code example shows how to use put-key-policy.

AWS CLI

To change the key policy for a KMS key

The following `put-key-policy` example changes the key policy for a customer managed key.

To begin, create a key policy and save it in a local JSON file. In this example, the file is `key_policy.json`. You can also specify the key policy as a string value of the `policy` parameter.

The first statement in this key policy gives the AWS account permission to use IAM policies to control access to the KMS key. The second statement gives the `test-user` user permission to run the `describe-key` and `list-keys` commands on the KMS key.

Contents of `key_policy.json`:

```
{
  "Version" : "2012-10-17",
  "Id" : "key-default-1",
  "Statement" : [
    {
      "Sid" : "Enable IAM User Permissions",
      "Effect" : "Allow",
      "Principal" : {
        "AWS" : "arn:aws:iam::111122223333:root"
      },
      "Action" : "kms:*",
      "Resource" : "*"
    },
    {
      "Sid" : "Allow Use of Key",
      "Effect" : "Allow",
      "Principal" : {
        "AWS" : "arn:aws:iam::111122223333:user/test-user"
      },
      "Action" : [
        "kms:DescribeKey",
        "kms:ListKeys"
      ],
      "Resource" : "*"
    }
  ]
}
```

To identify the KMS key, this example uses the key ID, but you can also use a key ARN. To specify the key policy, the command uses the `policy` parameter. To indicate that the policy

is in a file, it uses the required `file://` prefix. This prefix is required to identify files on all supported operating systems. Finally, the command uses the `policy-name` parameter with a value of `default`. If no policy name is specified, the default value is `default`. The only valid value is `default`.

```
aws kms put-key-policy \  
  --policy-name default \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --policy file://key_policy.json
```

This command does not produce any output. To verify that the command was effective, use the `get-key-policy` command. The following example command gets the key policy for the same KMS key. The output parameter with a value of `text` returns a text format that is easy to read.

```
aws kms get-key-policy \  
  --policy-name default \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --output text
```

Output:

```
{  
  "Version" : "2012-10-17",  
  "Id" : "key-default-1",  
  "Statement" : [  
    {  
      "Sid" : "Enable IAM User Permissions",  
      "Effect" : "Allow",  
      "Principal" : {  
        "AWS" : "arn:aws:iam::111122223333:root"  
      },  
      "Action" : "kms:*",  
      "Resource" : "*"   
    },  
    {  
      "Sid" : "Allow Use of Key",  
      "Effect" : "Allow",  
      "Principal" : {  
        "AWS" : "arn:aws:iam::111122223333:user/test-user"  
      },  
    },  
  ],  
}
```



```
        "Action" : [ "kms:Describe", "kms:List" ],
        "Resource" : "*"
    }
]
}
```

For more information, see [Changing a Key Policy](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [PutKeyPolicy](#) in *AWS CLI Command Reference*.

re-encrypt

The following code example shows how to use `re-encrypt`.

AWS CLI

Example 1: To re-encrypt an encrypted message under a different symmetric KMS key (Linux and macOS).

The following `re-encrypt` command example demonstrates the recommended way to re-encrypt data with the AWS CLI.

Provide the ciphertext in a file. In the value of the `--ciphertext-blob` parameter, use the `fileb://` prefix, which tells the CLI to read the data from a binary file. If the file is not in the current directory, type the full path to file. For more information about reading AWS CLI parameter values from a file, see [Loading AWS CLI parameters from a file](https://docs.aws.amazon.com/cli/latest/userguide/cli-usage-parameters-file.html) <<https://docs.aws.amazon.com/cli/latest/userguide/cli-usage-parameters-file.html>> in the *AWS Command Line Interface User Guide* and [Best Practices for Local File Parameters](https://aws.amazon.com/blogs/developer/best-practices-for-local-file-parameters/) <<https://aws.amazon.com/blogs/developer/best-practices-for-local-file-parameters/>> in the *AWS Command Line Tool Blog*. Specify the source KMS key, which decrypts the ciphertext. The `--source-key-id` parameter is not required when decrypting with symmetric encryption KMS keys. AWS KMS can get the KMS key that was used to encrypt the data from the metadata in the ciphertext blob. But it's always a best practice to specify the KMS key you are using. This practice ensures that you use the KMS key that you intend, and prevents you from inadvertently decrypting a ciphertext using a KMS key you do not trust. Specify the destination KMS key, which re-encrypts the data. The `--destination-key-id` parameter is always required. This example uses a key ARN, but you can use any valid key identifier. Request the plaintext output as a text value. The `--query` parameter tells the CLI to get only the value of the `Plaintext` field from the output. The `--output` parameter returns the output as text. Base64-decode

the plaintext and save it in a file. The following example pipes (|) the value of the Plaintext parameter to the Base64 utility, which decodes it. Then, it redirects (>) the decoded output to the ExamplePlaintext file.

Before running this command, replace the example key IDs with valid key identifiers from your AWS account.

```
aws kms re-encrypt \  
  --ciphertext-blob fileb://ExampleEncryptedFile \  
  --source-key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --destination-key-id 0987dcba-09fe-87dc-65ba-ab0987654321 \  
  --query CiphertextBlob \  
  --output text | base64 --decode > ExampleReEncryptedFile
```

This command produces no output. The output from the re-encrypt command is base64-decoded and saved in a file.

For more information, see ReEncrypt <https://docs.aws.amazon.com/kms/latest/APIReference/API_ReEncrypt.html> in the *AWS Key Management Service API Reference*.

Example 2: To re-encrypt an encrypted message under a different symmetric KMS key (Windows command prompt).

The following re-encrypt command example is the same as the previous one except that it uses the certutil utility to Base64-decode the plaintext data. This procedure requires two commands, as shown in the following examples.

Before running this command, replace the example key ID with a valid key ID from your AWS account.

```
aws kms re-encrypt ^  
  --ciphertext-blob fileb://ExampleEncryptedFile ^  
  --source-key-id 1234abcd-12ab-34cd-56ef-1234567890ab ^  
  --destination-key-id 0987dcba-09fe-87dc-65ba-ab0987654321 ^  
  --query CiphertextBlob ^  
  --output text > ExampleReEncryptedFile.base64
```

Then use the certutil utility

```
certutil -decode ExamplePlaintextFile.base64 ExamplePlaintextFile
```

Output:

```
Input Length = 18
Output Length = 12
CertUtil: -decode command completed successfully.
```

For more information, see `ReEncrypt` <https://docs.aws.amazon.com/kms/latest/APIReference/API_ReEncrypt.html> in the *AWS Key Management Service API Reference*.

- For API details, see [ReEncrypt](#) in *AWS CLI Command Reference*.

retire-grant

The following code example shows how to use `retire-grant`.

AWS CLI**To retire a grant on a customer master key**

The following `retire-grant` example deletes a grant from a KMS key.

The following example command specifies the `grant-id` and the `key-id` parameters. The value of the `key-id` parameter must be the key ARN of the KMS key.

```
aws kms retire-grant \
  --grant-id 1234a2345b8a4e350500d432bccf8ecd6506710e1391880c4f7f7140160c9af3 \
  --key-id arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab
```

This command produces no output. To confirm that the grant was retired, use the `list-grants` command.

For more information, see [Retiring and revoking grants](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [RetireGrant](#) in *AWS CLI Command Reference*.

revoke-grant

The following code example shows how to use `revoke-grant`.

AWS CLI

To revoke a grant on a customer master key

The following `revoke-grant` example deletes a grant from a KMS key. The following example command specifies the `grant-id` and the `key-id` parameters. The value of the `key-id` parameter can be the key ID or key ARN of the KMS key.

```
aws kms revoke-grant \  
  --grant-id 1234a2345b8a4e350500d432bccf8ecd6506710e1391880c4f7f7140160c9af3 \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

This command produces no output. To confirm that the grant was revoked, use the `list-grants` command.

For more information, see [Retiring and revoking grants](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [RevokeGrant](#) in *AWS CLI Command Reference*.

rotate-key-on-demand

The following code example shows how to use `rotate-key-on-demand`.

AWS CLI

To perform on-demand rotation of a KMS key

The following `rotate-key-on-demand` example immediately initiates rotation of the key material for the specified KMS key.

```
aws kms rotate-key-on-demand \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Output:

```
{  
  "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"  
}
```

For more information, see [How to perform on-demand key rotation](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [RotateKeyOnDemand](#) in *AWS CLI Command Reference*.

schedule-key-deletion

The following code example shows how to use `schedule-key-deletion`.

AWS CLI

To schedule the deletion of a customer managed KMS key.

The following `schedule-key-deletion` example schedules the specified customer managed KMS key to be deleted in 15 days.

The `--key-id` parameter identifies the KMS key. This example uses a key ARN value, but you can use either the key ID or the ARN of the KMS key. The `--pending-window-in-days` parameter specifies the length of the 7-30 day waiting period. By default, the waiting period is 30 days. This example specifies a value of 15, which tells AWS to permanently delete the KMS key 15 days after the command completes.

```
aws kms schedule-key-deletion \  
  --key-id arn:aws:kms:us-  
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab \  
  --pending-window-in-days 15
```

The response includes the key ARN, key state, waiting period (`PendingWindowInDays`), and the deletion date in Unix time. To view the deletion date in local time, use the AWS KMS console. KMS keys in the `PendingDeletion` key state cannot be used in cryptographic operations.

```
{  
  "KeyId": "arn:aws:kms:us-  
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab",  
  "DeletionDate": "2022-06-18T23:43:51.272000+00:00",  
  "KeyState": "PendingDeletion",  
  "PendingWindowInDays": 15  
}
```

For more information, see [Deleting keys](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [ScheduleKeyDeletion](#) in *AWS CLI Command Reference*.

sign

The following code example shows how to use `sign`.

AWS CLI

Example 1: To generate a digital signature for a message

The following `sign` example generates a cryptographic signature for a short message. The output of the command includes a base-64 encoded `Signature` field that you can verify by using the `verify` command.

You must specify a message to sign and a signing algorithm that your asymmetric KMS key supports. To get the signing algorithms for your KMS key, use the `describe-key` command.

In AWS CLI 2.0, the value of the `message` parameter must be Base64-encoded. Or, you can save the message in a file and use the `fileb://` prefix, which tells the AWS CLI to read binary data from the file.

Before running this command, replace the example key ID with a valid key ID from your AWS account. The key ID must represent an asymmetric KMS key with a key usage of `SIGN_VERIFY`.

```
msg=(echo 'Hello World' | base64)

aws kms sign \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --message fileb://UnsignedMessage \
  --message-type RAW \
  --signing-algorithm RSASSA_PKCS1_V1_5_SHA_256
```

Output:

```
{
  "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "Signature": "ABCDEFhpyVYyTxbafE74ccSvEJLJr3zuoV1Hfymz4qv+
fxmxNLA7SE1SiF8lHw80fKZZ3bJ...",
  "SigningAlgorithm": "RSASSA_PKCS1_V1_5_SHA_256"
```

```
}
```

For more information about using asymmetric KMS keys in AWS KMS, see [Asymmetric keys in AWS KMS](#) in the *AWS Key Management Service Developer Guide*.

Example 2: To save a digital signature in a file (Linux and macOS)

The following `sign` example generates a cryptographic signature for a short message stored in a local file. The command also gets the `Signature` property from the response, Base64-decodes it and saves it in the `ExampleSignature` file. You can use the signature file in a `verify` command that verifies the signature.

The `sign` command requires a Base64-encoded message and a signing algorithm that your asymmetric KMS key supports. To get the signing algorithms that your KMS key supports, use the `describe-key` command.

Before running this command, replace the example key ID with a valid key ID from your AWS account. The key ID must represent an asymmetric KMS key with a key usage of `SIGN_VERIFY`.

```
echo 'hello world' | base64 > EncodedMessage

aws kms sign \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --message fileb://EncodedMessage \
  --message-type RAW \
  --signing-algorithm RSASSA_PKCS1_V1_5_SHA_256 \
  --output text \
  --query Signature | base64 --decode > ExampleSignature
```

This command produces no output. This example extracts the `Signature` property of the output and saves it in a file.

For more information about using asymmetric KMS keys in AWS KMS, see [Asymmetric keys in AWS KMS](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [Sign](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add a tag to a KMS key

The following `tag-resource` example adds "Purpose": "Test" and "Dept": "IT" tags to a customer managed KMS key. You can use tags like these to label KMS keys and create categories of KMS keys for permissions and auditing.

To specify the KMS key, use the `key-id` parameter. This example uses a key ID value, but you can use a key ID or key ARN in this command.

```
aws kms tag-resource \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --tags TagKey='Purpose',TagValue='Test' TagKey='Dept',TagValue='IT'
```

This command produces no output. To view the tags on an AWS KMS KMS key, use the `list-resource-tags` command.

For more information about using tags in AWS KMS, see [Tagging keys](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To delete a tag from a KMS key

The following `untag-resource` example deletes the tag with the "Purpose" key from a customer managed KMS key.

To specify the KMS key, use the `key-id` parameter. This example uses a key ID value, but you can use a key ID or key ARN in this command. Before running this command, replace the example key ID with a valid key ID from your AWS account.

```
aws kms untag-resource \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --tag-key 'Purpose'
```


This command produces no output. To view the tags on an AWS KMS key, use the `list-resource-tags` command.

For more information about using tags in AWS KMS, see [Tagging keys](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-alias

The following code example shows how to use `update-alias`.

AWS CLI

To associate an alias with a different KMS key

The following `update-alias` example associates the alias `alias/test-key` with a different KMS key.

The `--alias-name` parameter specifies the alias. The alias name value must begin with `alias/`. The `--target-key-id` parameter specifies the KMS key to associate with the alias. You don't need to specify the current KMS key for the alias.

```
aws kms update-alias \  
  --alias-name alias/test-key \  
  --target-key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

This command produces no output. To find the alias, use the `list-aliases` command.

For more information, see [Updating aliases](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [UpdateAlias](#) in *AWS CLI Command Reference*.

update-custom-key-store

The following code example shows how to use `update-custom-key-store`.

AWS CLI

Example 1: To edit the friendly name of a custom key store

The following `update-custom-key-store` example changes the name of the custom key store. This example works for an AWS CloudHSM key store or an external key store.

Use the `custom-key-store-id` to identify the key store. Use the `new-custom-key-store-name` parameter to specify the new friendly name.

To update the friendly name of an AWS CloudHSM key store, you must first disconnect the key store, such as by using the `disconnect-custom-key-store` command. You can update the friendly name of an external key store while it is connected or disconnected. To find the connection state of your custom key store, use the `describe-custom-key-store` command.

```
aws kms update-custom-key-store \  
  --custom-key-store-id cks-1234567890abcdef0 \  
  --new-custom-key-store-name ExampleKeyStore
```

This command does not return any data. To verify that the command worked, use a `describe-custom-key-stores` command.

For more information about updating an AWS CloudHSM key store, see [Editing AWS CloudHSM key store settings](#) in the *AWS Key Management Service Developer Guide*.

For more information about updating an external key store, see [Editing external key store properties](#) in the *AWS Key Management Service Developer Guide*.

Example 2: To edit the `kmsuser` password of an AWS CloudHSM key store

The following `update-custom-key-store` example updates the value of the `kmsuser` password to the current password for the `kmsuser` in the CloudHSM cluster associated with the specified key store. This command doesn't change the `kmsuser` password in the cluster. It just tells AWS KMS the current password. If KMS doesn't have the current `kmsuser` password, it cannot connect to the AWS CloudHSM key store.

NOTE: Before updating an AWS CloudHSM key store, you must disconnect it. Use the `disconnect-custom-key-store` command. After the command completes, you can reconnect the AWS CloudHSM key store. Use the `connect-custom-key-store` command.

```
aws kms update-custom-key-store \  
  --custom-key-store-id cks-1234567890abcdef0 \  
  --key-store-password ExamplePassword
```

This command does not return any output. To verify that the change was effective, use a `describe-custom-key-stores` command.

For more information about updating an AWS CloudHSM key store, see [Editing AWS CloudHSM key store settings](#) in the *AWS Key Management Service Developer Guide*.

Example 3: To edit the AWS CloudHSM cluster of an AWS CloudHSM key store

The following example changes the AWS CloudHSM cluster that is associated with an AWS CloudHSM key store to a related cluster, such as a different backup of the same cluster.

NOTE: Before updating an AWS CloudHSM key store, you must disconnect it. Use the `disconnect-custom-key-store` command. After the command completes, you can reconnect the AWS CloudHSM key store. Use the `connect-custom-key-store` command.

```
aws kms update-custom-key-store \  
  --custom-key-store-id cks-1234567890abcdef0 \  
  --cloud-hsm-cluster-id cluster-1a23b4cdefg
```

This command does not return any output. To verify that the change was effective, use a `describe-custom-key-stores` command.

For more information about updating an AWS CloudHSM key store, see [Editing AWS CloudHSM key store settings](#) in the *AWS Key Management Service Developer Guide*.

Example 4: To edit the proxy authentication credential of an external key store

The following example updates the proxy authentication credential for your external key store. You must specify both the `raw-secret-access-key` and the `access-key-id`, even if you are changing only one of the values. You can use this feature to fix an invalid credential or to change the credential when the external key store proxy rotates it.

Establish the proxy authentication credential for AWS KMS on your external key store. Then use this command to provide the credential to AWS KMS. AWS KMS uses this credential to sign its requests to your external key store proxy.

You can update the proxy authentication credential while the external key store is connected or disconnected. To find the connection state of your custom key store, use the `describe-custom-key-store` command.

```
aws kms update-custom-key-store \  
  --raw-secret-access-key   
  --access-key-id   
  --external-key-store-id   
  --external-key-store-arn   
  --external-key-store-proxy-arn
```

```
--custom-key-store-id cks-1234567890abcdef0 \  
--xks-proxy-authentication-credential "AccessKeyId=ABCDE12345670EXAMPLE,  
RawSecretAccessKey=DXjSUawne12fr6SKC7G25CNxTyWKE5PF9XX6H/u9pSo="
```

This command does not return any output. To verify that the change was effective, use a `describe-custom-key-stores` command.

For more information about updating an external key store, see [Editing external key store properties](#) in the *AWS Key Management Service Developer Guide*.

Example 5: To edit the proxy connectivity of an external key store

The following example changes the external key store proxy connectivity option from public endpoint connectivity to VPC endpoint service connectivity. In addition to changing the `xks-proxy-connectivity` value, you must change the `xks-proxy-uri-endpoint` value to reflect the private DNS name associated with the VPC endpoint service. You must also add an `xks-proxy-vpc-endpoint-service-name` value.

NOTE: Before updating the proxy connectivity of an external store, you must disconnect it. Use the `disconnect-custom-key-store` command. After the command completes, you can reconnect the external key store by using the `connect-custom-key-store` command.

```
aws kms update-custom-key-store \  
  --custom-key-store-id cks-1234567890abcdef0 \  
  --xks-proxy-connectivity VPC_ENDPOINT_SERVICE \  
  --xks-proxy-uri-endpoint "https://myproxy-private.xks.example.com" \  
  --xks-proxy-vpc-endpoint-service-name "com.amazonaws.vpce.us-east-1.vpce-svc-  
example"
```

This command does not return any output. To verify that the change was effective, use a `describe-custom-key-stores` command.

For more information about updating an external key store, see [Editing external key store properties](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [UpdateCustomKeyStore](#) in *AWS CLI Command Reference*.

update-key-description

The following code example shows how to use `update-key-description`.

AWS CLI

Example 1: To add or change a description to a customer managed KMS key

The following `update-key-description` example adds a description to a customer managed KMS key. You can use the same command to change an existing description.

The `--key-id` parameter identifies the KMS key in the command. This example uses a key ARN value, but you can use either the key ID or the key ARN of the KMS key. The `--description` parameter specifies the new description. The value of this parameter replaces the current description of the KMS key, if any.

```
aws kms update-key-description \  
  --key-id arn:aws:kms:us-  
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab \  
  --description "IT Department test key"
```

This command produces no output. To view the description of a KMS key, use the `describe-key` command.

For more information, see [UpdateKeyDescription](#) in the *AWS Key Management Service API Reference*.

Example 2: To delete the description of a customer managed KMS key

The following `update-key-description` example deletes the description to a customer managed KMS key.

The `--key-id` parameter identifies the KMS key in the command. This example uses a key ID value, but you can use either the key ID or the key ARN of the KMS key. The `--description` parameter with an empty string value (`''`) deletes the existing description.

```
aws kms update-key-description \  
  --key-id 0987dcba-09fe-87dc-65ba-ab0987654321 \  
  --description ''
```

This command produces no output. To view the description of a KMS key, use the `describe-key` command.

For more information, see [UpdateKeyDescription](#) in the *AWS Key Management Service API Reference*.

- For API details, see [UpdateKeyDescription](#) in *AWS CLI Command Reference*.

verify

The following code example shows how to use `verify`.

AWS CLI

To verify a digital signature

The following `verify` example verifies a cryptographic signature for a short, Base64-encoded message. The key ID, message, message type, and signing algorithm must be same ones that were used to sign the message. The signature that you specify cannot be base64-encoded. For help decoding the signature that the `sign` command returns, see the `sign` command examples.

The output of the command includes a Boolean `SignatureValid` field that indicates that the signature was verified. If the signature validation fails, the `verify` command fails, too.

Before running this command, replace the example key ID with a valid key ID from your AWS account.

```
aws kms verify \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --message fileb://EncodedMessage \  
  --message-type RAW \  
  --signing-algorithm RSASSA_PKCS1_V1_5_SHA_256 \  
  --signature fileb://ExampleSignature
```

Output:

```
{  
  "KeyId": "arn:aws:kms:us-  
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",  
  "SignatureValid": true,  
  "SigningAlgorithm": "RSASSA_PKCS1_V1_5_SHA_256"  
}
```

For more information about using asymmetric KMS keys in AWS KMS, see [Using asymmetric keys](#) in the *AWS Key Management Service Developer Guide*.

- For API details, see [Verify](#) in *AWS CLI Command Reference*.

Lake Formation examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Lake Formation.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-lf-tags-to-resource

The following code example shows how to use `add-lf-tags-to-resource`.

AWS CLI

To attach one or more LF-tags to an existing resource

The following `add-lf-tags-to-resource` example attaches given LF-tag to the table resource.

```
aws lakeformation add-lf-tags-to-resource \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
```

```
"CatalogId": "123456789111",
"Resource": {
  "Table": {
    "CatalogId": "123456789111",
    "DatabaseName": "tpc",
    "Name": "dl_tpc_promotion"
  }
},
"LFTags": [{
  "CatalogId": "123456789111",
  "TagKey": "usergroup",
  "TagValues": [
    "analyst"
  ]
}]
}
```

Output:

```
{
  "Failures": []
}
```

For more information, see [Assigning LF-Tags to Data Catalog resources](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [AddLfTagsToResource](#) in *AWS CLI Command Reference*.

batch-grant-permissions

The following code example shows how to use batch-grant-permissions.

AWS CLI

To bulk grant permissions on resources to the principals

The following batch-grant-permissions example bulk grants access on specified resources to the principals.

```
aws lakeformation batch-grant-permissions \
  --cli-input-json file://input.json
```


Contents of input.json:

```
{
  "CatalogId": "123456789111",
  "Entries": [{
    "Id": "1",
    "Principal": {
      "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
developer"
    },
    "Resource": {
      "Table": {
        "CatalogId": "123456789111",
        "DatabaseName": "tpc",
        "Name": "dl_tpc_promotion"
      }
    },
    "Permissions": [
      "ALL"
    ],
    "PermissionsWithGrantOption": [
      "ALL"
    ]
  },
  {
    "Id": "2",
    "Principal": {
      "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
developer"
    },
    "Resource": {
      "Table": {
        "CatalogId": "123456789111",
        "DatabaseName": "tpc",
        "Name": "dl_tpc_customer"
      }
    },
    "Permissions": [
      "ALL"
    ],
    "PermissionsWithGrantOption": [
      "ALL"
    ]
  },
}
```

```
{
  "Id": "3",
  "Principal": {
    "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
business-analyst"
  },
  "Resource": {
    "Table": {
      "CatalogId": "123456789111",
      "DatabaseName": "tpc",
      "Name": "dl_tpc_promotion"
    }
  },
  "Permissions": [
    "ALL"
  ],
  "PermissionsWithGrantOption": [
    "ALL"
  ]
},
{
  "Id": "4",
  "Principal": {
    "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
developer"
  },
  "Resource": {
    "DataCellsFilter": {
      "TableCatalogId": "123456789111",
      "DatabaseName": "tpc",
      "TableName": "dl_tpc_item",
      "Name": "developer_item"
    }
  },
  "Permissions": [
    "SELECT"
  ],
  "PermissionsWithGrantOption": []
}
]
```

Output:

```
{
  "Failures": []
}
```

For more information, see [Granting and revoking permissions on Data Catalog resources](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [BatchGrantPermissions](#) in *AWS CLI Command Reference*.

batch-revoke-permissions

The following code example shows how to use `batch-revoke-permissions`.

AWS CLI

To bulk revoke permissions on resources from the principals

The following `batch-revoke-permissions` example bulk revokes access on specified resources from the principals.

```
aws lakeformation batch-revoke-permissions \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "CatalogId": "123456789111",
  "Entries": [{
    "Id": "1",
    "Principal": {
      "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
developer"
    },
    "Resource": {
      "Table": {
        "CatalogId": "123456789111",
        "DatabaseName": "tpc",
        "Name": "dl_tpc_promotion"
      }
    },
    "Permissions": [
```

```

        "ALL"
      ],
      "PermissionsWithGrantOption": [
        "ALL"
      ]
    },
    {
      "Id": "2",
      "Principal": {
        "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
business-analyst"
      },
      "Resource": {
        "Table": {
          "CatalogId": "123456789111",
          "DatabaseName": "tpc",
          "Name": "dl_tpc_promotion"
        }
      },
      "Permissions": [
        "ALL"
      ],
      "PermissionsWithGrantOption": [
        "ALL"
      ]
    }
  ]
}

```

Output:

```

{
  "Failures": []
}

```

For more information, see [Granting and revoking permissions on Data Catalog resources](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [BatchRevokePermissions](#) in *AWS CLI Command Reference*.

cancel-transaction

The following code example shows how to use `cancel-transaction`.

AWS CLI

To cancel a transaction

The following `cancel-transaction` example cancels the transaction.

```
aws lakeformation cancel-transaction \  
  --transaction-id='b014d972ca8347b89825e33c5774aec4'
```

This command produces no output.

For more information, see [Reading from and writing to the data lake within transactions](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [CancelTransaction](#) in *AWS CLI Command Reference*.

commit-transaction

The following code example shows how to use `commit-transaction`.

AWS CLI

To commit transaction

The following `commit-transaction` example commits the transaction.

```
aws lakeformation commit-transaction \  
  --transaction-id='b014d972ca8347b89825e33c5774aec4'
```

Output:

```
{  
  "TransactionStatus": "committed"  
}
```

For more information, see [Reading from and writing to the data lake within transactions](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [CommitTransaction](#) in *AWS CLI Command Reference*.

create-data-cells-filter

The following code example shows how to use `create-data-cells-filter`.

AWS CLI

Example 1: To create data cell filter

The following `create-data-cells-filter` example creates a data cell filter to allow one to grant access to certain columns based on row condition.

```
aws lakeformation create-data-cells-filter \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "TableData": {  
    "ColumnNames": ["p_channel_details", "p_start_date_sk", "p_promo_name"],  
    "DatabaseName": "tpc",  
    "Name": "developer_promotion",  
    "RowFilter": {  
      "FilterExpression": "p_promo_name='ese'"  
    },  
    "TableCatalogId": "123456789111",  
    "TableName": "dl_tpc_promotion"  
  }  
}
```

This command produces no output.

For more information, see [Data filtering and cell-level security in Lake Formation](#) in the *AWS Lake Formation Developer Guide*.

Example 2: To create column filter

The following `create-data-cells-filter` example creates a data filter to allow one to grant access to certain columns.

```
aws lakeformation create-data-cells-filter \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "TableData": {
    "ColumnNames": ["p_channel_details", "p_start_date_sk", "p_promo_name"],
    "DatabaseName": "tpc",
    "Name": "developer_promotion_allrows",
    "RowFilter": {
      "AllRowsWildcard": {}
    },
    "TableCatalogId": "123456789111",
    "TableName": "dl_tpc_promotion"
  }
}
```

This command produces no output.

For more information, see [Data filtering and cell-level security in Lake Formation](#) in the *AWS Lake Formation Developer Guide*.

Example 3: To create data filter with exclude columns

The following `create-data-cells-filter` example creates a data filter to allow one to grant access all except the mentioned columns.

```
aws lakeformation create-data-cells-filter \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "TableData": {
    "ColumnWildcard": {
      "ExcludedColumnNames": ["p_channel_details", "p_start_date_sk"]
    },
    "DatabaseName": "tpc",
    "Name": "developer_promotion_excludecolumn",
    "RowFilter": {
      "AllRowsWildcard": {}
    },
    "TableCatalogId": "123456789111",
    "TableName": "dl_tpc_promotion"
  }
}
```

This command produces no output.

For more information, see [Data filtering and cell-level security in Lake Formation](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [CreateDataCellsFilter](#) in *AWS CLI Command Reference*.

create-lf-tag

The following code example shows how to use `create-lf-tag`.

AWS CLI

To create LF-Tag

The following `create-lf-tag` example creates an LF-Tag with the specified name and values.

```
aws lakeformation create-lf-tag \  
  --catalog-id '123456789111' \  
  --tag-key 'usergroup' \  
  --tag-values '["developer","analyst","campaign"]'
```

This command produces no output.

For more information, see [Managing LF-Tags for metadata access control](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [CreateLfTag](#) in *AWS CLI Command Reference*.

delete-data-cells-filter

The following code example shows how to use `delete-data-cells-filter`.

AWS CLI

To delete data cell filter

The following `delete-data-cells-filter` example deletes given data cell filter.

```
aws lakeformation delete-data-cells-filter \  
  --cli-input-json file://input.json
```

Contents of `input.json`:


```
{
  "TableCatalogId": "123456789111",
  "DatabaseName": "tpc",
  "TableName": "dl_tpc_promotion",
  "Name": "developer_promotion"
}
```

This command produces no output.

For more information, see [Data filtering and cell-level security in Lake Formation](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [DeleteDataCellsFilter](#) in *AWS CLI Command Reference*.

delete-lf-tag

The following code example shows how to use `delete-lf-tag`.

AWS CLI

To delete LF-Tag definition

The following `delete-lf-tag` example deletes LF-Tag definition.

```
aws lakeformation delete-lf-tag \
  --catalog-id '123456789111' \
  --tag-key 'usergroup'
```

This command produces no output.

For more information, see [Managing LF-Tags for metadata access control](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [DeleteLfTag](#) in *AWS CLI Command Reference*.

delete-objects-on-cancel

The following code example shows how to use `delete-objects-on-cancel`.

AWS CLI

To delete object when transaction is cancelled

The following `delete-objects-on-cancel` example deletes the listed s3 object when the transaction is cancelled.

```
aws lakeformation delete-objects-on-cancel \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "CatalogId": "012345678901",  
  "DatabaseName": "tpc",  
  "TableName": "dl_tpc_household_demographics_gov",  
  "TransactionId": "1234d972ca8347b89825e33c5774aec4",  
  "Objects": [{  
    "Uri": "s3://lf-data-lake-012345678901/target/  
dl_tpc_household_demographics_gov/run-unnamed-1-part-block-0-r-00000-snappy-  
ff26b17504414fe88b302cd795eabd00.parquet",  
    "ETag": "1234ab1fc50a316b149b4e1f21a73800"  
  }]  
}
```

This command produces no output.

For more information, see [Reading from and writing to the data lake within transactions](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [DeleteObjectsOnCancel](#) in *AWS CLI Command Reference*.

deregister-resource

The following code example shows how to use `deregister-resource`.

AWS CLI

To deregister data lake storage

The following `deregister-resource` example deregisters the resource as managed by the Lake Formation.

```
aws lakeformation deregister-resource \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "ResourceArn": "arn:aws:s3:::lf-emr-athena-result-123"
}
```

This command produces no output.

For more information, see [Adding an Amazon S3 location to your data lake](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [DeregisterResource](#) in *AWS CLI Command Reference*.

describe-transaction

The following code example shows how to use `describe-transaction`.

AWS CLI

To retrieve a transaction details

The following `describe-transaction` example returns the details of a single transaction.

```
aws lakeformation describe-transaction \
  --transaction-id='8cb4b1a7cc8d486fbaca9a64e7d9f5ce'
```

Output:

```
{
  "TransactionDescription": {
    "TransactionId": "12345972ca8347b89825e33c5774aec4",
    "TransactionStatus": "committed",
    "TransactionStartTime": "2022-08-10T14:29:04.046000+00:00",
    "TransactionEndTime": "2022-08-10T14:29:09.681000+00:00"
  }
}
```

For more information, see [Reading from and writing to the data lake within transactions](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [DescribeTransaction](#) in *AWS CLI Command Reference*.

extend-transaction

The following code example shows how to use `extend-transaction`.

AWS CLI

To extend a transaction

The following `extend-transaction` example extends the transaction.

```
aws lakeformation extend-transaction \  
  --transaction-id='8cb4b1a7cc8d486fbaca9a64e7d9f5ce'
```

This command produces no output.

For more information, see [Reading from and writing to the data lake within transactions](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [ExtendTransaction](#) in *AWS CLI Command Reference*.

get-data-lake-settings

The following code example shows how to use `get-data-lake-settings`.

AWS CLI

To retrieve AWS Lake Formation-managed data lake settings

The following `get-data-lake-settings` example retrieves the list of data lake administrators and other data lake settings.

```
aws lakeformation get-data-lake-settings \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "CatalogId": "123456789111"  
}
```

Output:

```

{
  "DataLakeSettings": {
    "DataLakeAdmins": [{
      "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-admin"
    }],
    "CreateDatabaseDefaultPermissions": [],
    "CreateTableDefaultPermissions": [
      {
        "Principal": {
          "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
        },
        "Permissions": [
          "ALL"
        ]
      }
    ],
    "TrustedResourceOwners": [],
    "AllowExternalDataFiltering": true,
    "ExternalDataFilteringAllowList": [{
      "DataLakePrincipalIdentifier": "123456789111"
    }],
    "AuthorizedSessionTagValueList": [
      "Amazon EMR"
    ]
  }
}

```

For more information, see [Changing the default security settings for your data lake](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [GetDataLakeSettings](#) in *AWS CLI Command Reference*.

get-effective-permissions-for-path

The following code example shows how to use `get-effective-permissions-for-path`.

AWS CLI

To retrieve permissions on resources located at specific path

The following `get-effective-permissions-for-path` example returns the Lake Formation permissions for a specified table or database resource located at a path in Amazon S3.

```
aws lakeformation get-effective-permissions-for-path \  
  --cli-input-json file://input.json
```

Contents of input.json:

```
{  
  "CatalogId": "123456789111",  
  "ResourceArn": "arn:aws:s3:::lf-data-lake-123456789111"  
}
```

Output:

```
{  
  "Permissions": [{  
    "Principal": {  
      "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-  
campaign-manager"  
    },  
    "Resource": {  
      "Database": {  
        "Name": "tpc"  
      }  
    },  
    "Permissions": [  
      "DESCRIBE"  
    ],  
    "PermissionsWithGrantOption": []  
  },  
  {  
    "Principal": {  
      "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:role/EMR-  
RuntimeRole"  
    },  
    "Resource": {  
      "Database": {  
        "Name": "tpc"  
      }  
    },  
    "Permissions": [  
      "ALL"  
    ],  
    "PermissionsWithGrantOption": []  
  }  
}
```

```
    },
    {
      "Principal": {
        "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:saml-
provider/oktaSAMLProvider:user/emr-developer"
      },
      "Resource": {
        "Database": {
          "Name": "tpc"
        }
      },
      "Permissions": [
        "ALL",
        "DESCRIBE"
      ],
      "PermissionsWithGrantOption": []
    },
    {
      "Principal": {
        "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
admin"
      },
      "Resource": {
        "Database": {
          "Name": "tpc"
        }
      },
      "Permissions": [
        "ALL",
        "ALTER",
        "CREATE_TABLE",
        "DESCRIBE",
        "DROP"
      ],
      "PermissionsWithGrantOption": [
        "ALL",
        "ALTER",
        "CREATE_TABLE",
        "DESCRIBE",
        "DROP"
      ]
    },
    {
      "Principal": {
```

```

        "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:role/LF-
GlueServiceRole"
      },
      "Resource": {
        "Database": {
          "Name": "tpc"
        }
      },
      "Permissions": [
        "CREATE_TABLE"
      ],
      "PermissionsWithGrantOption": []
    }
  ],
  "NextToken":
  "E5S1JDSTZ1eUp6SWpvaU9UQTN0RE0zTXpFeE5Ua3pJbjE5TENKbGVIQnBjbUYwYVc5dUlqcDdJbk5sWTI5dVpITW1P
}

```

For more information, see [Managing Lake Formation permissions](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [GetEffectivePermissionsForPath](#) in *AWS CLI Command Reference*.

get-lf-tag

The following code example shows how to use `get-lf-tag`.

AWS CLI

To retrieve LF-tag definition

The following `get-lf-tag` example retrieves LF-tag definition.

```

aws lakeformation get-lf-tag \
  --catalog-id '123456789111' \
  --tag-key 'usergroup'

```

Output:

```

{
  "CatalogId": "123456789111",
  "TagKey": "usergroup",
  "TagValues": [

```



```
    "analyst",
    "campaign",
    "developer"
  ]
}
```

For more information, see [Managing LF-Tags for metadata access control](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [GetLfTag](#) in *AWS CLI Command Reference*.

get-query-state

The following code example shows how to use `get-query-state`.

AWS CLI

To retrieve state of a submitted query

The following `get-query-state` example returns the state of a query previously submitted.

```
aws lakeformation get-query-state \
  --query-id='1234273f-4a62-4cda-8d98-69615ee8be9b'
```

Output:

```
{
  "State": "FINISHED"
}
```

For more information, see [Transactional data operations](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [GetQueryState](#) in *AWS CLI Command Reference*.

get-query-statistics

The following code example shows how to use `get-query-statistics`.

AWS CLI

To retrieve query statistics

The following `get-query-statistics` example retrieves statistics on the planning and execution of a query.

```
aws lakeformation get-query-statistics \  
  --query-id='1234273f-4a62-4cda-8d98-69615ee8be9b'
```

Output:

```
{  
  "ExecutionStatistics": {  
    "AverageExecutionTimeMillis": 0,  
    "DataScannedBytes": 0,  
    "WorkUnitsExecutedCount": 0  
  },  
  "PlanningStatistics": {  
    "EstimatedDataToScanBytes": 43235,  
    "PlanningTimeMillis": 2377,  
    "QueueTimeMillis": 440,  
    "WorkUnitsGeneratedCount": 1  
  },  
  "QuerySubmissionTime": "2022-08-11T02:14:38.641870+00:00"  
}
```

For more information, see [Transactional data operations](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [GetQueryStatistics](#) in *AWS CLI Command Reference*.

get-resource-lf-tags

The following code example shows how to use `get-resource-lf-tags`.

AWS CLI

To list LF-tags

The following `list-lf-tags` example returns list of LF-tags that the requester has permission to view.

```
aws lakeformation list-lf-tags \  
  --cli-input-json file://input.json
```

Contents of input.json:

```
{
  "CatalogId": "123456789111",
  "ResourceShareType": "ALL",
  "MaxResults": 2
}
```

Output:

```
{
  "LFTags": [{
    "CatalogId": "123456789111",
    "TagKey": "category",
    "TagValues": [
      "private",
      "public"
    ]
  },
  {
    "CatalogId": "123456789111",
    "TagKey": "group",
    "TagValues": [
      "analyst",
      "campaign",
      "developer"
    ]
  }
],
  "NextToken": "kIiwiZXhwaXJhdGlvbiI6eyJzZWNVbmRzIjoxNjYwMDY4dCI6ZmFsc2V9"
}
```

For more information, see [Managing LF-Tags for metadata access control](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [GetResourceLfTags](#) in *AWS CLI Command Reference*.

get-table-objects

The following code example shows how to use `get-table-objects`.

AWS CLI

To list objects of governed table

The following `get-table-objects` example returns the set of Amazon S3 objects that make up the specified governed table.

```
aws lakeformation get-table-objects \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "CatalogId": "012345678901",  
  "DatabaseName": "tpc",  
  "TableName": "dl_tpc_household_demographics_gov",  
  "QueryAsOfTime": "2022-08-10T15:00:00"  
}
```

Output:

```
{  
  "Objects": [{  
    "PartitionValues": [],  
    "Objects": [{  
      "Uri": "s3://lf-data-lake-012345678901/target/  
dl_tpc_household_demographics_gov/run-unnamed-1-part-block-0-r-00000-snappy-  
ff26b17504414fe88b302cd795eabd00.parquet",  
      "ETag": "12345b1fc50a316b149b4e1f21a73800",  
      "Size": 43235  
    }]  
  }]  
}
```

For more information, see [Reading from and writing to the data lake within transactions](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [GetTableObjects](#) in *AWS CLI Command Reference*.

get-work-unit-results

The following code example shows how to use `get-work-unit-results`.

AWS CLI

To retrieve work units of given query

The following `get-work-unit-results` example returns the work units resulting from the query.

```
aws lakeformation get-work-units \  
  --query-id='1234273f-4a62-4cda-8d98-69615ee8be9b' \  
  --work-unit-id '0' \  
  --work-unit-token 'B2fMSdmQXe9umX8Ux8XCo4=' outfile
```

Output:

```
outfile with Blob content.
```

For more information, see [Transactional data operations](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [GetWorkUnitResults](#) in *AWS CLI Command Reference*.

get-work-units

The following code example shows how to use `get-work-units`.

AWS CLI

To retrieve work units

The following `get-work-units` example retrieves the work units generated by the `StartQueryPlanning` operation.

```
aws lakeformation get-work-units \  
  --query-id='1234273f-4a62-4cda-8d98-69615ee8be9b'
```

Output:

```
{  
  "WorkUnitRanges": [{  
    "WorkUnitIdMax": 0,  
    "WorkUnitIdMin": 0,  
    "WorkUnitToken":  
    "1234eMAk4kL04umqEL4Z5WuxL04AXwABABVhd3MtY3J5cHRvLXB1YmxpYy1rZXkAREEwYm9QbkhINmFYTWphbmMxZW  
+f88jzGrYq22gE6jkQlp0B  
+0et2eqNUMfudAAAAfjB8BqkqhkiG9w0BBwagbzBtAgEAMGgGCSqGSIb3DQEHATAeBg1ghkgBZQMEAS4wEQQMCOEWRda
```

```
wAAAAEAAAAAAAAAAAAAAAAAAEAAACX3/w5h75QAPomfKH+cyEKYU1yccUmB1
+VSojiG0tdsUk7vcjYXUUb0Ym3dvdqRqX2s4gROM0n
+Ij8R0/8jYmnHkpvYAFNVRPyETyIKg7k5Z9+5I1c2d3446Jw/moWGGxjH8AEG9h27ytm0hozxD0Ei/
F2ZoXz6w1GDfGUo/2WxCkY0hTyNaw6TM
+7drTM7yrW4iNVLUM0LX0xnFjIAhLhooWJek6vjQZUAZzB1AjBH8okRtYP8R7AY2W1s/
hqFBhG0V4l42AC0LxsuZbMQrE2SzWZUZ0E9Uew7/n0cyX4CMQDR79INyv4ysMByW9kKGGKyba+cCNk1ExMR
+btBQBmMuB2fMSdmQXe9umX8Ux8XCo4="
  }],
  "QueryId": "1234273f-4a62-4cda-8d98-69615ee8be9b"
}
```

For more information, see [Transactional data operations](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [GetWorkUnits](#) in *AWS CLI Command Reference*.

grant-permissions

The following code example shows how to use `grant-permissions`.

AWS CLI

Example 1: To grant permissions to the principal on resources using LF-Tags

The following `grant-permissions` example grants ALL permissions to the principal on database resource that matches the LF-Tag policy.

```
aws lakeformation grant-permissions \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "CatalogId": "123456789111",
  "Principal": {
    "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-admin"
  },
  "Resource": {
    "LFTagPolicy": {
      "CatalogId": "123456789111",
      "ResourceType": "DATABASE",
      "Expression": [{"
```

```

        "TagKey": "usergroup",
        "TagValues": [
            "analyst",
            "developer"
        ]
    }
}
    ],
    "Permissions": [
        "ALL"
    ],
    "PermissionsWithGrantOption": [
        "ALL"
    ]
}

```

This command produces no output.

For more information, see [Granting and revoking permissions on Data Catalog resources](#) in the *AWS Lake Formation Developer Guide*.

Example 2: To grant column level permissions to the principal

The following `grant-permissions` example grants permission to select specific column to the principal.

```

aws lakeformation grant-permissions \
  --cli-input-json file://input.json

```

Contents of `input.json`:

```

{
  "CatalogId": "123456789111",
  "Principal": {
    "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-developer"
  },
  "Resource": {
    "TableWithColumns": {
      "CatalogId": "123456789111",
      "ColumnNames": ["p_end_date_sk"],
      "DatabaseName": "tpc",
      "Name": "dl_tpc_promotion"
    }
  }
}

```

```
    }
  },
  "Permissions": [
    "SELECT"
  ],
  "PermissionsWithGrantOption": []
}
```

This command produces no output.

For more information, see [Granting and revoking permissions on Data Catalog resources](#) in the *AWS Lake Formation Developer Guide*.

Example 3: To grant table permissions to the principal

The following `grant-permissions` example grants select permission on all tables of given database to the principal.

```
aws lakeformation grant-permissions \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "CatalogId": "123456789111",
  "Principal": {
    "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-developer"
  },
  "Resource": {
    "Table": {
      "CatalogId": "123456789111",
      "DatabaseName": "tpc",
      "TableWildcard": {}
    }
  },
  "Permissions": [
    "SELECT"
  ],
  "PermissionsWithGrantOption": []
}
```

This command produces no output.

For more information, see [Granting and revoking permissions on Data Catalog resources](#) in the *AWS Lake Formation Developer Guide*.

Example 4: To grant permissions on LF-Tags to the principal

The following `grant-permissions` example grants associate permission on LF-Tags to the principal.

```
aws lakeformation grant-permissions \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "CatalogId": "123456789111",  
  "Principal": {  
    "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-developer"  
  },  
  "Resource": {  
    "LFTag": {  
      "CatalogId": "123456789111",  
      "TagKey": "category",  
      "TagValues": [  
        "private", "public"  
      ]  
    }  
  },  
  "Permissions": [  
    "ASSOCIATE"  
  ],  
  "PermissionsWithGrantOption": []  
}
```

This command produces no output.

For more information, see [Granting and revoking permissions on Data Catalog resources](#) in the *AWS Lake Formation Developer Guide*.

Example 5: To grant permissions on data locations to the principal

The following `grant-permissions` example grants permission on data location to the principal.

```
aws lakeformation grant-permissions \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "CatalogId": "123456789111",  
  "Principal": {  
    "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-developer"  
  },  
  "Resource": {  
    "DataLocation": {  
      "CatalogId": "123456789111",  
      "ResourceArn": "arn:aws:s3:::lf-data-lake-123456789111"  
    }  
  },  
  "Permissions": [  
    "DATA_LOCATION_ACCESS"  
  ],  
  "PermissionsWithGrantOption": []  
}
```

This command produces no output.

For more information, see [Granting and revoking permissions on Data Catalog resources](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [GrantPermissions](#) in *AWS CLI Command Reference*.

list-data-cells-filter

The following code example shows how to use `list-data-cells-filter`.

AWS CLI

To list data cell filters

The following `list-data-cells-filter` example list data cell filter for given table.

```
aws lakeformation list-data-cells-filter \  
  --cli-input-json file://input.json
```

Contents of input.json:

```
{
  "MaxResults": 2,
  "Table": {
    "CatalogId": "123456789111",
    "DatabaseName": "tpc",
    "Name": "dl_tpc_promotion"
  }
}
```

Output:

```
{
  "DataCellsFilters": [{
    "TableCatalogId": "123456789111",
    "DatabaseName": "tpc",
    "TableName": "dl_tpc_promotion",
    "Name": "developer_promotion",
    "RowFilter": {
      "FilterExpression": "p_promo_name='ese'"
    }
  },
  "ColumnNames": [
    "p_channel_details",
    "p_start_date_sk",
    "p_purpose",
    "p_promo_id",
    "p_promo_name",
    "p_end_date_sk",
    "p_discount_active"
  ]
},
{
  "TableCatalogId": "123456789111",
  "DatabaseName": "tpc",
  "TableName": "dl_tpc_promotion",
  "Name": "developer_promotion_allrows",
  "RowFilter": {
    "FilterExpression": "TRUE",
    "AllRowsWildcard": {}
  },
  "ColumnNames": [
    "p_channel_details",
```

```

        "p_start_date_sk",
        "p_promo_name"
    ]
}
],
"NextToken": "2MDA2MTgwNiwibmFub3MiOjE0MDAwMDAwMH19"
}

```

For more information, see [Data filtering and cell-level security in Lake Formation](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [ListDataCellsFilter](#) in *AWS CLI Command Reference*.

list-permissions

The following code example shows how to use `list-permissions`.

AWS CLI

Example 1: To retrieve list of principal permissions on the resource

The following `list-permissions` example returns a list of principal permissions on the database resources.

```
aws lakeformation list-permissions \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "CatalogId": "123456789111",
  "ResourceType": "DATABASE",
  "MaxResults": 2
}
```

Output:

```
{
  "PrincipalResourcePermissions": [{
    "Principal": {
      "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
campaign-manager"
    }
  }
]
```

```

    },
    "Resource": {
      "Database": {
        "CatalogId": "123456789111",
        "Name": "tpc"
      }
    },
    "Permissions": [
      "DESCRIBE"
    ],
    "PermissionsWithGrantOption": []
  ]],
  "NextToken":
  "E5S1JDSTZ1eUp6SWpvaU9UQTN0RE0zTXpFeE5Ua3pJbjE5TENKbGVIQnBjbUYwYVc5dUlqcDdJbk5sWTI5dVpITW1P
}

```

For more information, see [Managing Lake Formation permissions](#) in the *AWS Lake Formation Developer Guide*.

Example 2: To retrieve list of principal permissions on the table with data filters

The following `list-permissions` example list the permissions on the table with related data filters granted to the principal.

```

aws lakeformation list-permissions \
  --cli-input-json file://input.json

```

Contents of `input.json`:

```

{
  "CatalogId": "123456789111",
  "Resource": {
    "Table": {
      "CatalogId": "123456789111",
      "DatabaseName": "tpc",
      "Name": "dl_tpc_customer"
    }
  },
  "IncludeRelated": "TRUE",
  "MaxResults": 10
}

```

Output:

```
{
  "PrincipalResourcePermissions": [{
    "Principal": {
      "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:role/
Admin"
    },
    "Resource": {
      "Table": {
        "CatalogId": "123456789111",
        "DatabaseName": "customer",
        "Name": "customer_invoice"
      }
    },
    "Permissions": [
      "ALL",
      "ALTER",
      "DELETE",
      "DESCRIBE",
      "DROP",
      "INSERT"
    ],
    "PermissionsWithGrantOption": [
      "ALL",
      "ALTER",
      "DELETE",
      "DESCRIBE",
      "DROP",
      "INSERT"
    ]
  }],
  {
    "Principal": {
      "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:role/
Admin"
    },
    "Resource": {
      "TableWithColumns": {
        "CatalogId": "123456789111",
        "DatabaseName": "customer",
        "Name": "customer_invoice",
        "ColumnWildcard": {}
      }
    }
  }
}
```

```

    },
    "Permissions": [
      "SELECT"
    ],
    "PermissionsWithGrantOption": [
      "SELECT"
    ]
  },
  {
    "Principal": {
      "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:role/
Admin"
    },
    "Resource": {
      "DataCellsFilter": {
        "TableCatalogId": "123456789111",
        "DatabaseName": "customer",
        "TableName": "customer_invoice",
        "Name": "dl_us_customer"
      }
    },
    "Permissions": [
      "DESCRIBE",
      "SELECT",
      "DROP"
    ],
    "PermissionsWithGrantOption": []
  }
],
"NextToken": "VyeUFjY291bnRQZXJtaXNzaW9ucyI6ZmFsc2V9"
}

```

For more information, see [Managing Lake Formation permissions](#) in the *AWS Lake Formation Developer Guide*.

Example 3: To retrieve list of principal permissions on the LF-Tags

The following `list-permissions` example list the permissions on the LF-Tags granted to the principal.

```

aws lakeformation list-permissions \
  --cli-input-json file://input.json

```

Contents of input.json:

```
{
  "CatalogId": "123456789111",
  "Resource": {
    "LFTag": {
      "CatalogId": "123456789111",
      "TagKey": "category",
      "TagValues": [
        "private"
      ]
    }
  },
  "MaxResults": 10
}
```

Output:

```
{
  "PrincipalResourcePermissions": [{
    "Principal": {
      "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-admin"
    },
    "Resource": {
      "LFTag": {
        "CatalogId": "123456789111",
        "TagKey": "category",
        "TagValues": [
          "*"
        ]
      }
    },
    "Permissions": [
      "DESCRIBE"
    ],
    "PermissionsWithGrantOption": [
      "DESCRIBE"
    ]
  },
  {
    "Principal": {
```



```

        "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-
admin"
      },
      "Resource": {
        "LFTag": {
          "CatalogId": "123456789111",
          "TagKey": "category",
          "TagValues": [
            "*"
          ]
        }
      },
      "Permissions": [
        "ASSOCIATE"
      ],
      "PermissionsWithGrantOption": [
        "ASSOCIATE"
      ]
    }
  ],
  "NextToken": "EJwY21GMGFxOxVJanA3SW50cm1pc3Npb25zIjpmYWxzZX0="
}

```

For more information, see [Managing Lake Formation permissions](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [ListPermissions](#) in *AWS CLI Command Reference*.

list-resources

The following code example shows how to use `list-resources`.

AWS CLI

To lists the resources managed by the Lake Formation

The following `list-resources` example lists the resources matching the condition that is managed by the Lake Formation.

```

aws lakeformation list-resources \
  --cli-input-json file://input.json

```

Contents of `input.json`:

```
{
  "FilterConditionList": [{
    "Field": "ROLE_ARN",
    "ComparisonOperator": "CONTAINS",
    "StringValueList": [
      "123456789111"
    ]
  }],
  "MaxResults": 10
}
```

Output:

```
{
  "ResourceInfoList": [{
    "ResourceArn": "arn:aws:s3:::lf-data-lake-123456789111",
    "RoleArn": "arn:aws:iam::123456789111:role/LF-GlueServiceRole",
    "LastModified": "2022-07-21T02:12:46.669000+00:00"
  },
  {
    "ResourceArn": "arn:aws:s3:::lf-emr-test-123456789111",
    "RoleArn": "arn:aws:iam::123456789111:role/EMRLFS3Role",
    "LastModified": "2022-07-29T16:22:03.211000+00:00"
  }
  ]
}
```

For more information, see [Managing Lake Formation permissions](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [ListResources](#) in *AWS CLI Command Reference*.

list-transactions

The following code example shows how to use `list-transactions`.

AWS CLI

To list all transactions details

The following `list-transactions` example returns metadata about transactions and their status.

```
aws lakeformation list-transactions \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "CatalogId": "123456789111",  
  "StatusFilter": "ALL",  
  "MaxResults": 3  
}
```

Output:

```
{  
  "Transactions": [{  
    "TransactionId": "1234569f08804cb790d950d4d0fe485e",  
    "TransactionStatus": "committed",  
    "TransactionStartTime": "2022-08-10T14:32:29.220000+00:00",  
    "TransactionEndTime": "2022-08-10T14:32:33.751000+00:00"  
  },  
  {  
    "TransactionId": "12345972ca8347b89825e33c5774aec4",  
    "TransactionStatus": "committed",  
    "TransactionStartTime": "2022-08-10T14:29:04.046000+00:00",  
    "TransactionEndTime": "2022-08-10T14:29:09.681000+00:00"  
  },  
  {  
    "TransactionId": "12345daf6cb047dbba8ad9b0414613b2",  
    "TransactionStatus": "committed",  
    "TransactionStartTime": "2022-08-10T13:56:51.261000+00:00",  
    "TransactionEndTime": "2022-08-10T13:56:51.547000+00:00"  
  }  
  ],  
  "NextToken": "77X1ebypsI7os+X21hHsZLGNC DK3nNGpwRdFpicS0Hg cX1/  
QMoniUAKcpR3kj3ts3PVdMA=="  
}
```

For more information, see [Reading from and writing to the data lake within transactions](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [ListTransactions](#) in *AWS CLI Command Reference*.

put-data-lake-settings

The following code example shows how to use `put-data-lake-settings`.

AWS CLI

To set AWS Lake Formation-managed data lake settings

The following `put-data-lake-settings` example sets the list of data lake administrators and other data lake settings.

```
aws lakeformation put-data-lake-settings \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "DataLakeSettings": {  
    "DataLakeAdmins": [{  
      "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-  
admin"  
    }  
  ],  
  "CreateDatabaseDefaultPermissions": [],  
  "CreateTableDefaultPermissions": [],  
  "TrustedResourceOwners": [],  
  "AllowExternalDataFiltering": true,  
  "ExternalDataFilteringAllowList": [{  
    "DataLakePrincipalIdentifier": "123456789111"  
  }],  
  "AuthorizedSessionTagValueList": ["Amazon EMR"]  
}
```

This command produces no output.

For more information, see [Changing the default security settings for your data lake](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [PutDataLakeSettings](#) in *AWS CLI Command Reference*.

register-resource

The following code example shows how to use `register-resource`.

AWS CLI

Example 1: To register data lake storage using Service Linked Role

The following `register-resource` example registers the resource as managed by the Lake Formation using Service linked role.

```
aws lakeformation register-resource \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "ResourceArn": "arn:aws:s3:::lf-emr-athena-result-123",  
  "UseServiceLinkedRole": true  
}
```

This command produces no output.

For more information, see [Adding an Amazon S3 location to your data lake](#) in the *AWS Lake Formation Developer Guide*.

Example 2: To register data lake storage using custom role

The following `register-resource` example registers the resource as managed by the Lake Formation using custom role.

```
aws lakeformation register-resource \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "ResourceArn": "arn:aws:s3:::lf-emr-athena-result-123",  
  "UseServiceLinkedRole": false,  
  "RoleArn": "arn:aws:iam::123456789111:role/LF-GlueServiceRole"  
}
```

This command produces no output.

For more information, see [Adding an Amazon S3 location to your data lake](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [RegisterResource](#) in *AWS CLI Command Reference*.

remove-lf-tags-from-resource

The following code example shows how to use `remove-lf-tags-from-resource`.

AWS CLI

To remove LF-Tag from a resource

The following `remove-lf-tags-from-resource` example removes the LF-Tag association with the table resource.

```
aws lakeformation remove-lf-tags-from-resource \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "CatalogId": "123456789111",  
  "Resource": {  
    "Table": {  
      "CatalogId": "123456789111",  
      "DatabaseName": "tpc",  
      "Name": "dl_tpc_promotion"  
    }  
  },  
  "LFTags": [{  
    "CatalogId": "123456789111",  
    "TagKey": "usergroup",  
    "TagValues": [  
      "developer"  
    ]  
  }]  
}
```

Output:

```
{
  "Failures": []
}
```

For more information, see [Assigning LF-Tags to Data Catalog resources](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [RemoveLfTagsFromResource](#) in *AWS CLI Command Reference*.

revoke-permissions

The following code example shows how to use `revoke-permissions`.

AWS CLI

To revoke permissions on resources from the principal

The following `revoke-permissions` example revoke principal access to specific table of a given database.

```
aws lakeformation revoke-permissions \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "CatalogId": "123456789111",
  "Principal": {
    "DataLakePrincipalIdentifier": "arn:aws:iam::123456789111:user/lf-developer"
  },
  "Resource": {
    "Table": {
      "CatalogId": "123456789111",
      "DatabaseName": "tpc",
      "Name": "dl_tpc_promotion"
    }
  },
  "Permissions": [
    "ALL"
  ],
  "PermissionsWithGrantOption": []
}
```

This command produces no output.

For more information, see [Granting and revoking permissions on Data Catalog resources](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [RevokePermissions](#) in *AWS CLI Command Reference*.

search-databases-by-lf-tags

The following code example shows how to use search-databases-by-lf-tags.

AWS CLI

To search on database resources by LFTags

The following search-databases-by-lf-tags example search on database resources matching LFTag expression.

```
aws lakeformation search-databases-by-lf-tags \  
  --cli-input-json file://input.json
```

Contents of input.json:

```
{  
  "MaxResults": 1,  
  "CatalogId": "123456789111",  
  "Expression": [{  
    "TagKey": "usergroup",  
    "TagValues": [  
      "developer"  
    ]  
  }]  
}
```

Output:

```
{  
  "DatabaseList": [{  
    "Database": {  
      "CatalogId": "123456789111",  
      "Name": "tpc"  
    },  
  },  
}
```



```
    "LFTags": [{
      "CatalogId": "123456789111",
      "TagKey": "usergroup",
      "TagValues": [
        "developer"
      ]
    }]
  ]
}
```

For more information, see [Viewing the resources that a LF-Tag is assigned to](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [SearchDatabasesByLfTags](#) in *AWS CLI Command Reference*.

search-tables-by-lf-tags

The following code example shows how to use `search-tables-by-lf-tags`.

AWS CLI

To search on table resources by LFTags

The following `search-tables-by-lf-tags` example search on table resources matching LFTag expression.

```
aws lakeformation search-tables-by-lf-tags \
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{
  "MaxResults": 2,
  "CatalogId": "123456789111",
  "Expression": [{
    "TagKey": "usergroup",
    "TagValues": [
      "developer"
    ]
  }]
}
```

Output:

```
{
  "NextToken": "c2VhcmNoQWxsVGFnc0luVGFibGVzIjpmYWxzZX0=",
  "TableList": [{
    "Table": {
      "CatalogId": "123456789111",
      "DatabaseName": "tpc",
      "Name": "dl_tpc_item"
    },
    "LFTagOnDatabase": [{
      "CatalogId": "123456789111",
      "TagKey": "usergroup",
      "TagValues": [
        "developer"
      ]
    }
  ]},
  "LFTagsOnTable": [{
    "CatalogId": "123456789111",
    "TagKey": "usergroup",
    "TagValues": [
      "developer"
    ]
  }
  ],
  "LFTagsOnColumns": [{
    "Name": "i_item_desc",
    "LFTags": [{
      "CatalogId": "123456789111",
      "TagKey": "usergroup",
      "TagValues": [
        "developer"
      ]
    }
  ]
  },
  {
    "Name": "i_container",
    "LFTags": [{
      "CatalogId": "123456789111",
      "TagKey": "usergroup",
      "TagValues": [
        "developer"
      ]
    }
  ]
  },
}
```

```
{
  "Name": "i_wholesale_cost",
  "LFTags": [{
    "CatalogId": "123456789111",
    "TagKey": "usergroup",
    "TagValues": [
      "developer"
    ]
  }]
},
{
  "Name": "i_manufact_id",
  "LFTags": [{
    "CatalogId": "123456789111",
    "TagKey": "usergroup",
    "TagValues": [
      "developer"
    ]
  }]
},
{
  "Name": "i_brand_id",
  "LFTags": [{
    "CatalogId": "123456789111",
    "TagKey": "usergroup",
    "TagValues": [
      "developer"
    ]
  }]
},
{
  "Name": "i_formulation",
  "LFTags": [{
    "CatalogId": "123456789111",
    "TagKey": "usergroup",
    "TagValues": [
      "developer"
    ]
  }]
},
{
  "Name": "i_current_price",
  "LFTags": [{
    "CatalogId": "123456789111",
```

```
        "TagKey": "usergroup",
        "TagValues": [
            "developer"
        ]
    }
},
{
    "Name": "i_size",
    "LFTags": [{
        "CatalogId": "123456789111",
        "TagKey": "usergroup",
        "TagValues": [
            "developer"
        ]
    }]
},
{
    "Name": "i_rec_start_date",
    "LFTags": [{
        "CatalogId": "123456789111",
        "TagKey": "usergroup",
        "TagValues": [
            "developer"
        ]
    }]
},
{
    "Name": "i_manufact",
    "LFTags": [{
        "CatalogId": "123456789111",
        "TagKey": "usergroup",
        "TagValues": [
            "developer"
        ]
    }]
},
{
    "Name": "i_item_sk",
    "LFTags": [{
        "CatalogId": "123456789111",
        "TagKey": "usergroup",
        "TagValues": [
            "developer"
        ]
    }]
}
```

```
    ]],
  },
  {
    "Name": "i_manager_id",
    "LFTags": [{
      "CatalogId": "123456789111",
      "TagKey": "usergroup",
      "TagValues": [
        "developer"
      ]
    }]
  },
  {
    "Name": "i_item_id",
    "LFTags": [{
      "CatalogId": "123456789111",
      "TagKey": "usergroup",
      "TagValues": [
        "developer"
      ]
    }]
  },
  {
    "Name": "i_class_id",
    "LFTags": [{
      "CatalogId": "123456789111",
      "TagKey": "usergroup",
      "TagValues": [
        "developer"
      ]
    }]
  },
  {
    "Name": "i_class",
    "LFTags": [{
      "CatalogId": "123456789111",
      "TagKey": "usergroup",
      "TagValues": [
        "developer"
      ]
    }]
  },
  {
    "Name": "i_category",
```

```
    "LFTags": [{
      "CatalogId": "123456789111",
      "TagKey": "usergroup",
      "TagValues": [
        "developer"
      ]
    }]
  },
  {
    "Name": "i_category_id",
    "LFTags": [{
      "CatalogId": "123456789111",
      "TagKey": "usergroup",
      "TagValues": [
        "developer"
      ]
    }]
  },
  {
    "Name": "i_brand",
    "LFTags": [{
      "CatalogId": "123456789111",
      "TagKey": "usergroup",
      "TagValues": [
        "developer"
      ]
    }]
  },
  {
    "Name": "i_units",
    "LFTags": [{
      "CatalogId": "123456789111",
      "TagKey": "usergroup",
      "TagValues": [
        "developer"
      ]
    }]
  },
  {
    "Name": "i_rec_end_date",
    "LFTags": [{
      "CatalogId": "123456789111",
      "TagKey": "usergroup",
      "TagValues": [
```

```
        "developer"
      ]
    ]}
  },
  {
    "Name": "i_color",
    "LFTags": [{
      "CatalogId": "123456789111",
      "TagKey": "usergroup",
      "TagValues": [
        "developer"
      ]
    }]
  },
  {
    "Name": "i_product_name",
    "LFTags": [{
      "CatalogId": "123456789111",
      "TagKey": "usergroup",
      "TagValues": [
        "developer"
      ]
    }]
  }
]
}]
}
```

For more information, see [Viewing the resources that a LF-Tag is assigned to](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [SearchTablesByLfTags](#) in *AWS CLI Command Reference*.

start-query-planning

The following code example shows how to use start-query-planning.

AWS CLI

To process query statement

The following start-query-planning example submits a request to process a query statement.

```
aws lakeformation start-query-planning \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "QueryPlanningContext": {  
    "CatalogId": "012345678901",  
    "DatabaseName": "tpc"  
  },  
  "QueryString": "select * from dl_tpc_household_demographics_gov where  
hd_income_band_sk=9"  
}
```

Output:

```
{  
  "QueryId": "772a273f-4a62-4cda-8d98-69615ee8be9b"  
}
```

For more information, see [Reading from and writing to the data lake within transactions](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [StartQueryPlanning](#) in *AWS CLI Command Reference*.

start-transaction

The following code example shows how to use `start-transaction`.

AWS CLI

To start new transaction

The following `start-transaction` example starts a new transaction and returns its transaction ID.

```
aws lakeformation start-transaction \  
  --transaction-type = 'READ_AND_WRITE'
```

Output:


```
{
  "TransactionId": "b014d972ca8347b89825e33c5774aec4"
}
```

For more information, see [Reading from and writing to the data lake within transactions](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [StartTransaction](#) in *AWS CLI Command Reference*.

update-lf-tag

The following code example shows how to use `update-lf-tag`.

AWS CLI

To update LF-Tag definition

The following `update-lf-tag` example updates LF-Tag definition.

```
aws lakeformation update-lf-tag \
  --catalog-id '123456789111' \
  --tag-key 'usergroup' \
  --tag-values-to-add ['"admin"']
```

This command produces no output.

For more information, see [Managing LF-Tags for metadata access control](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [UpdateLfTag](#) in *AWS CLI Command Reference*.

update-table-objects

The following code example shows how to use `update-table-objects`.

AWS CLI

To modify objects of governed table

The following `update-table-objects` example adds provided S3 objects to the specified governed table.

```
aws lakeformation update-table-objects \  
  --cli-input-json file://input.json
```

Contents of `input.json`:

```
{  
  "CatalogId": "012345678901",  
  "DatabaseName": "tpc",  
  "TableName": "dl_tpc_household_demographics_gov",  
  "TransactionId": "12347a9f75424b9b915f6ff201d2a190",  
  "WriteOperations": [{  
    "AddObject": {  
      "Uri": "s3://lf-data-lake-012345678901/target/  
dl_tpc_household_demographics_gov/run-unnamed-1-part-block-0-r-00000-snappy-  
ff26b17504414fe88b302cd795eabd00.parquet",  
      "ETag": "1234ab1fc50a316b149b4e1f21a73800",  
      "Size": 42200  
    }  
  }  
}]  
}
```

This command produces no output.

For more information, see [Reading from and writing to the data lake within transactions](#) in the *AWS Lake Formation Developer Guide*.

- For API details, see [UpdateTableObjects](#) in *AWS CLI Command Reference*.

Lambda examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Lambda.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-layer-version-permission

The following code example shows how to use `add-layer-version-permission`.

AWS CLI

To add permissions to a layer version

The following `add-layer-version-permission` example grants permission for the specified account to use version 1 of the layer `my-layer`.

```
aws lambda add-layer-version-permission \  
  --layer-name my-layer \  
  --statement-id xaccount \  
  --action lambda:GetLayerVersion \  
  --principal 123456789012 \  
  --version-number 1
```

Output:

```
{  
  "RevisionId": "35d87451-f796-4a3f-a618-95a3671b0a0c",  
  "Statement":  
  {  
    "Sid": "xaccount",  
    "Effect": "Allow",  
    "Principal": {  
      "AWS": "arn:aws:iam::210987654321:root"  
    },  
    "Action": "lambda:GetLayerVersion",  
    "Resource": "arn:aws:lambda:us-east-2:123456789012:layer:my-layer:1"  
  }  
}
```

For more information, see [AWS Lambda Layers](#) in the *AWS Lambda Developer Guide*.

- For API details, see [AddLayerVersionPermission](#) in *AWS CLI Command Reference*.

add-permission

The following code example shows how to use `add-permission`.

AWS CLI

To add permissions to an existing Lambda function

The following `add-permission` example grants the Amazon SNS service permission to invoke a function named `my-function`.

```
aws lambda add-permission \  
  --function-name my-function \  
  --action lambda:InvokeFunction \  
  --statement-id sns \  
  --principal sns.amazonaws.com
```

Output:

```
{  
  "Statement":  
  {  
    "Sid": "sns",  
    "Effect": "Allow",  
    "Principal": {  
      "Service": "sns.amazonaws.com"  
    },  
    "Action": "lambda:InvokeFunction",  
    "Resource": "arn:aws:lambda:us-east-2:123456789012:function:my-function"  
  }  
}
```

For more information, see [Using Resource-based Policies for AWS Lambda](#) in the *AWS Lambda Developer Guide*.

- For API details, see [AddPermission](#) in *AWS CLI Command Reference*.

create-alias

The following code example shows how to use `create-alias`.

AWS CLI

To create an alias for a Lambda function

The following `create-alias` example creates an alias named `LIVE` that points to version 1 of the `my-function` Lambda function.

```
aws lambda create-alias \  
  --function-name my-function \  
  --description "alias for live version of function" \  
  --function-version 1 \  
  --name LIVE
```

Output:

```
{  
  "FunctionVersion": "1",  
  "Name": "LIVE",  
  "AliasArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function:LIVE",  
  "RevisionId": "873282ed-4cd3-4dc8-a069-d0c647e470c6",  
  "Description": "alias for live version of function"  
}
```

For more information, see [Configuring AWS Lambda Function Aliases](#) in the *AWS Lambda Developer Guide*.

- For API details, see [CreateAlias](#) in *AWS CLI Command Reference*.

create-event-source-mapping

The following code example shows how to use `create-event-source-mapping`.

AWS CLI

To create a mapping between an event source and an AWS Lambda function

The following `create-event-source-mapping` example creates a mapping between an SQS queue and the `my-function` Lambda function.

```
aws lambda create-event-source-mapping \  
  --function-name my-function \  
  --event-source-arn sqs:us-west-2:123456789012:queue/my-queue
```

```
--batch-size 5 \  
--event-source-arn arn:aws:sqs:us-west-2:123456789012:mySQSqueue
```

Output:

```
{  
  "UUID": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
  "StateTransitionReason": "USER_INITIATED",  
  "LastModified": 1569284520.333,  
  "BatchSize": 5,  
  "State": "Creating",  
  "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function",  
  "EventSourceArn": "arn:aws:sqs:us-west-2:123456789012:mySQSqueue"  
}
```

For more information, see [AWS Lambda Event Source Mapping](#) in the *AWS Lambda Developer Guide*.

- For API details, see [CreateEventSourceMapping](#) in *AWS CLI Command Reference*.

create-function

The following code example shows how to use `create-function`.

AWS CLI

To create a Lambda function

The following `create-function` example creates a Lambda function named `my-function`.

```
aws lambda create-function \  
  --function-name my-function \  
  --runtime nodejs18.x \  
  --zip-file fileb://my-function.zip \  
  --handler my-function.handler \  
  --role arn:aws:iam::123456789012:role/service-role/MyTestFunction-role-tges6bf4
```

Contents of `my-function.zip`:

This file is a deployment package that contains your function code and any dependencies.

Output:

```
{
  "TracingConfig": {
    "Mode": "PassThrough"
  },
  "CodeSha256": "PFn4S+er27qk+UuZSTKEQfNKG/XNn7QJs90mJgq6oH8=",
  "FunctionName": "my-function",
  "CodeSize": 308,
  "RevisionId": "873282ed-4cd3-4dc8-a069-d0c647e470c6",
  "MemorySize": 128,
  "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function",
  "Version": "$LATEST",
  "Role": "arn:aws:iam::123456789012:role/service-role/MyTestFunction-role-zgur6bf4",
  "Timeout": 3,
  "LastModified": "2023-10-14T22:26:11.234+0000",
  "Handler": "my-function.handler",
  "Runtime": "nodejs18.x",
  "Description": ""
}
```

For more information, see [AWS Lambda Function Configuration](#) in the *AWS Lambda Developer Guide*.

- For API details, see [CreateFunction](#) in *AWS CLI Command Reference*.

delete-alias

The following code example shows how to use `delete-alias`.

AWS CLI

To delete an alias of a Lambda function

The following `delete-alias` example deletes the alias named `LIVE` from the `my-function` Lambda function.

```
aws lambda delete-alias \
  --function-name my-function \
  --name LIVE
```

This command produces no output.

For more information, see [Configuring AWS Lambda Function Aliases](#) in the *AWS Lambda Developer Guide*.

- For API details, see [DeleteAlias](#) in *AWS CLI Command Reference*.

delete-event-source-mapping

The following code example shows how to use `delete-event-source-mapping`.

AWS CLI

To delete the mapping between an event source and an AWS Lambda function

The following `delete-event-source-mapping` example deletes the mapping between an SQS queue and the `my-function` Lambda function.

```
aws lambda delete-event-source-mapping \  
  --uuid a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
```

Output:

```
{  
  "UUID": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
  "StateTransitionReason": "USER_INITIATED",  
  "LastModified": 1569285870.271,  
  "BatchSize": 5,  
  "State": "Deleting",  
  "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function",  
  "EventSourceArn": "arn:aws:sqs:us-west-2:123456789012:mySQSqueue"  
}
```

For more information, see [AWS Lambda Event Source Mapping](#) in the *AWS Lambda Developer Guide*.

- For API details, see [DeleteEventSourceMapping](#) in *AWS CLI Command Reference*.

delete-function-concurrency

The following code example shows how to use `delete-function-concurrency`.

AWS CLI

To remove the reserved concurrent execution limit from a function

The following `delete-function-concurrency` example deletes the reserved concurrent execution limit from the `my-function` function.

```
aws lambda delete-function-concurrency \  
  --function-name my-function
```

This command produces no output.

For more information, see [Reserving Concurrency for a Lambda Function](#) in the *AWS Lambda Developer Guide*.

- For API details, see [DeleteFunctionConcurrency](#) in *AWS CLI Command Reference*.

`delete-function-event-invoke-config`

The following code example shows how to use `delete-function-event-invoke-config`.

AWS CLI

To delete an asynchronous invocation configuration

The following `delete-function-event-invoke-config` example deletes the asynchronous invocation configuration for the GREEN alias of the specified function.

```
aws lambda delete-function-event-invoke-config --function-name my-function:GREEN
```

- For API details, see [DeleteFunctionEventInvokeConfig](#) in *AWS CLI Command Reference*.

`delete-function`

The following code example shows how to use `delete-function`.

AWS CLI

Example 1: To delete a Lambda function by function name

The following `delete-function` example deletes the Lambda function named `my-function` by specifying the function's name.

```
aws lambda delete-function \  
  --function-name my-function
```

This command produces no output.

Example 2: To delete a Lambda function by function ARN

The following `delete-function` example deletes the Lambda function named `my-function` by specifying the function's ARN.

```
aws lambda delete-function \  
  --function-name arn:aws:lambda:us-west-2:123456789012:function:my-function
```

This command produces no output.

Example 3: To delete a Lambda function by partial function ARN

The following `delete-function` example deletes the Lambda function named `my-function` by specifying the function's partial ARN.

```
aws lambda delete-function \  
  --function-name 123456789012:function:my-function
```

This command produces no output.

For more information, see [AWS Lambda Function Configuration](#) in the *AWS Lambda Developer Guide*.

- For API details, see [DeleteFunction](#) in *AWS CLI Command Reference*.

delete-layer-version

The following code example shows how to use `delete-layer-version`.

AWS CLI

To delete a version of a Lambda layer

The following `delete-layer-version` example deletes version 2 of the layer named `my-layer`.

```
aws lambda delete-layer-version \  
  --layer-name my-layer \  
  --version-number 2
```

This command produces no output.

For more information, see [AWS Lambda Layers](#) in the *AWS Lambda Developer Guide*.

- For API details, see [DeleteLayerVersion](#) in *AWS CLI Command Reference*.

delete-provisioned-concurrency-config

The following code example shows how to use `delete-provisioned-concurrency-config`.

AWS CLI

To delete a provisioned concurrency configuration

The following `delete-provisioned-concurrency-config` example deletes the provisioned concurrency configuration for the GREEN alias of the specified function.

```
aws lambda delete-provisioned-concurrency-config \  
  --function-name my-function \  
  --qualifier GREEN
```

- For API details, see [DeleteProvisionedConcurrencyConfig](#) in *AWS CLI Command Reference*.

get-account-settings

The following code example shows how to use `get-account-settings`.

AWS CLI

To retrieve details about your account in an AWS Region

The following `get-account-settings` example displays the Lambda limits and usage information for your account.

```
aws lambda get-account-settings
```

Output:

```
{
  "AccountLimit": {
    "CodeSizeUnzipped": 262144000,
    "UnreservedConcurrentExecutions": 1000,
    "ConcurrentExecutions": 1000,
    "CodeSizeZipped": 52428800,
    "TotalCodeSize": 80530636800
  },
  "AccountUsage": {
    "FunctionCount": 4,
    "TotalCodeSize": 9426
  }
}
```

For more information, see [AWS Lambda Limits](#) in the *AWS Lambda Developer Guide*.

- For API details, see [GetAccountSettings](#) in *AWS CLI Command Reference*.

get-alias

The following code example shows how to use `get-alias`.

AWS CLI

To retrieve details about a function alias

The following `get-alias` example displays details for the alias named `LIVE` on the `my-function` Lambda function.

```
aws lambda get-alias \
  --function-name my-function \
  --name LIVE
```

Output:

```
{
  "FunctionVersion": "3",
  "Name": "LIVE",
  "AliasArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function:LIVE",
  "RevisionId": "594f41fb-b85f-4c20-95c7-6ca5f2a92c93",
  "Description": "alias for live version of function"
```

```
}
```

For more information, see [Configuring AWS Lambda Function Aliases](#) in the *AWS Lambda Developer Guide*.

- For API details, see [GetAlias](#) in *AWS CLI Command Reference*.

get-event-source-mapping

The following code example shows how to use `get-event-source-mapping`.

AWS CLI

To retrieve details about an event source mapping

The following `get-event-source-mapping` example displays the details for the mapping between an SQS queue and the `my-function` Lambda function.

```
aws lambda get-event-source-mapping \  
  --uuid "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"
```

Output:

```
{  
  "UUID": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
  "StateTransitionReason": "USER_INITIATED",  
  "LastModified": 1569284520.333,  
  "BatchSize": 5,  
  "State": "Enabled",  
  "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function",  
  "EventSourceArn": "arn:aws:sqs:us-west-2:123456789012:mySQSqueue"  
}
```

For more information, see [AWS Lambda Event Source Mapping](#) in the *AWS Lambda Developer Guide*.

- For API details, see [GetEventSourceMapping](#) in *AWS CLI Command Reference*.

get-function-concurrency

The following code example shows how to use `get-function-concurrency`.

AWS CLI

To view the reserved concurrency setting for a function

The following `get-function-concurrency` example retrieves the reserved concurrency setting for the specified function.

```
aws lambda get-function-concurrency \  
  --function-name my-function
```

Output:

```
{  
  "ReservedConcurrentExecutions": 250  
}
```

- For API details, see [GetFunctionConcurrency](#) in *AWS CLI Command Reference*.

get-function-configuration

The following code example shows how to use `get-function-configuration`.

AWS CLI

To retrieve the version-specific settings of a Lambda function

The following `get-function-configuration` example displays the settings for version 2 of the `my-function` function.

```
aws lambda get-function-configuration \  
  --function-name my-function:2
```

Output:

```
{  
  "FunctionName": "my-function",  
  "LastModified": "2019-09-26T20:28:40.438+0000",  
  "RevisionId": "e52502d4-9320-4688-9cd6-152a6ab7490d",  
  "MemorySize": 256,  
  "Version": "2",  
  "Role": "arn:aws:iam::123456789012:role/service-role/my-function-role-uy3l9qq",  
  "Timeout": 3,  
}
```

```
"Runtime": "nodejs10.x",
"TracingConfig": {
  "Mode": "PassThrough"
},
"CodeSha256": "5tT2qgzYUHaqwR716pZ2dpkn/0J1FrzJm1KidWoaCgk=",
"Description": "",
"VpcConfig": {
  "SubnetIds": [],
  "VpcId": "",
  "SecurityGroupIds": []
},
"CodeSize": 304,
"FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function:2",
"Handler": "index.handler"
}
```

For more information, see [AWS Lambda Function Configuration](#) in the *AWS Lambda Developer Guide*.

- For API details, see [GetFunctionConfiguration](#) in *AWS CLI Command Reference*.

get-function-event-invoke-config

The following code example shows how to use `get-function-event-invoke-config`.

AWS CLI

To view an asynchronous invocation configuration

The following `get-function-event-invoke-config` example retrieves the asynchronous invocation configuration for the BLUE alias of the specified function.

```
aws lambda get-function-event-invoke-config \
  --function-name my-function:BLUE
```

Output:

```
{
  "LastModified": 1577824396.653,
  "FunctionArn": "arn:aws:lambda:us-east-2:123456789012:function:my-
function:BLUE",
  "MaximumRetryAttempts": 0,
  "MaximumEventAgeInSeconds": 3600,
}
```

```
"DestinationConfig": {
  "OnSuccess": {},
  "OnFailure": {
    "Destination": "arn:aws:sqs:us-east-2:123456789012:failed-invocations"
  }
}
```

- For API details, see [GetFunctionEventInvokeConfig](#) in *AWS CLI Command Reference*.

get-function

The following code example shows how to use `get-function`.

AWS CLI

To retrieve information about a function

The following `get-function` example displays information about the `my-function` function.

```
aws lambda get-function \
  --function-name my-function
```

Output:

```
{
  "Concurrency": {
    "ReservedConcurrentExecutions": 100
  },
  "Code": {
    "RepositoryType": "S3",
    "Location": "https://awslambda-us-west-2-tasks.s3.us-west-2.amazonaws.com/snapshots/123456789012/my-function..."
  },
  "Configuration": {
    "TracingConfig": {
      "Mode": "PassThrough"
    },
    "Version": "$LATEST",
    "CodeSha256": "5tT2qgzYUHoqwR616pZ2dpkn/0J1FrzJm1KidWaaCgk=",
    "FunctionName": "my-function",
    "VpcConfig": {
```



```

        "SubnetIds": [],
        "VpcId": "",
        "SecurityGroupIds": []
    },
    "MemorySize": 128,
    "RevisionId": "28f0fb31-5c5c-43d3-8955-03e76c5c1075",
    "CodeSize": 304,
    "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function",
    "Handler": "index.handler",
    "Role": "arn:aws:iam::123456789012:role/service-role/helloWorldPython-role-uy3l9qq",
    "Timeout": 3,
    "LastModified": "2019-09-24T18:20:35.054+0000",
    "Runtime": "nodejs10.x",
    "Description": ""
}
}

```

For more information, see [AWS Lambda Function Configuration](#) in the *AWS Lambda Developer Guide*.

- For API details, see [GetFunction](#) in *AWS CLI Command Reference*.

get-layer-version-by-arn

The following code example shows how to use `get-layer-version-by-arn`.

AWS CLI

To retrieve information about a Lambda layer version

The following `get-layer-version-by-arn` example displays information about the layer version with the specified Amazon Resource Name (ARN).

```

aws lambda get-layer-version-by-arn \
  --arn "arn:aws:lambda:us-west-2:123456789012:layer:AWSLambda-Python311-SciPy1x:2"

```

Output:

```

{
  "LayerVersionArn": "arn:aws:lambda:us-west-2:123456789012:layer:AWSLambda-Python311-SciPy1x:2",

```

```

    "Description": "AWS Lambda SciPy layer for Python 3.11 (scipy-1.1.0,
numpy-1.15.4) https://github.com/scipy/scipy/releases/tag/v1.1.0 https://
github.com/numpy/numpy/releases/tag/v1.15.4",
    "CreateDate": "2023-10-12T10:09:38.398+0000",
    "LayerArn": "arn:aws:lambda:us-west-2:123456789012:layer:AWSLambda-Python311-
SciPy1x",
    "Content": {
      "CodeSize": 41784542,
      "CodeSha256": "GGmv8ocUw4cly0T8HL0Vx/f5V4RmSCGNjDIslY4VskM=",
      "Location": "https://awslambda-us-west-2-layers.s3.us-west-2.amazonaws.com/
snapshots/123456789012/..."
    },
    "Version": 2,
    "CompatibleRuntimes": [
      "python3.11"
    ],
    "LicenseInfo": "SciPy: https://github.com/scipy/scipy/blob/main/LICENSE.txt,
NumPy: https://github.com/numpy/numpy/blob/main/LICENSE.txt"
  }

```

For more information, see [AWS Lambda Layers](#) in the *AWS Lambda Developer Guide*.

- For API details, see [GetLayerVersionByArn](#) in *AWS CLI Command Reference*.

get-layer-version-policy

The following code example shows how to use `get-layer-version-policy`.

AWS CLI

To retrieve the permissions policy for a Lambda layer version

The following `get-layer-version-policy` example displays policy information about version 1 for the layer named `my-layer`.

```

aws lambda get-layer-version-policy \
  --layer-name my-layer \
  --version-number 1

```

Output:

```
{
```

```

    "Policy": {
      "Version": "2012-10-17",
      "Id": "default",
      "Statement": [
        {
          "Sid": "xaccount",
          "Effect": "Allow",
          "Principal": {"AWS": "arn:aws:iam::123456789012:root"},
          "Action": "lambda:GetLayerVersion",
          "Resource": "arn:aws:lambda:us-west-2:123456789012:layer:my-layer:1"
        }
      ],
      "RevisionId": "c68f21d2-cbf0-4026-90f6-1375ee465cd0"
    }
  }
}

```

For more information, see [AWS Lambda Layers](#) in the *AWS Lambda Developer Guide*.

- For API details, see [GetLayerVersionPolicy](#) in *AWS CLI Command Reference*.

get-layer-version

The following code example shows how to use `get-layer-version`.

AWS CLI

To retrieve information about a Lambda layer version

The following `get-layer-version` example displays information for version 1 of the layer named `my-layer`.

```

aws lambda get-layer-version \
  --layer-name my-layer \
  --version-number 1

```

Output:

```

{
  "Content": {
    "Location": "https://awslambda-us-east-2-layers.s3.us-east-2.amazonaws.com/snapshots/123456789012/my-layer-4aaa2fbb-ff77-4b0a-ad92-5b78a716a96a?versionId=27iWyA73cCAYqyH...",

```

```

    "CodeSha256": "tv9jJ0+rPbXUUXuRKi7CwHzKtLDkDRJLB3cC3Z/ouXo=",
    "CodeSize": 169
  },
  "LayerArn": "arn:aws:lambda:us-east-2:123456789012:layer:my-layer",
  "LayerVersionArn": "arn:aws:lambda:us-east-2:123456789012:layer:my-layer:1",
  "Description": "My Python layer",
  "CreateDate": "2018-11-14T23:03:52.894+0000",
  "Version": 1,
  "LicenseInfo": "MIT",
  "CompatibleRuntimes": [
    "python3.10",
    "python3.11"
  ]
}

```

For more information, see [AWS Lambda Layers](#) in the *AWS Lambda Developer Guide*.

- For API details, see [GetLayerVersion](#) in *AWS CLI Command Reference*.

get-policy

The following code example shows how to use get-policy.

AWS CLI

To retrieve the resource-based IAM policy for a function, version, or alias

The following get-policy example displays policy information about the my-function Lambda function.

```
aws lambda get-policy \
  --function-name my-function
```

Output:

```

{
  "Policy": {
    "Version": "2012-10-17",
    "Id": "default",
    "Statement": [
      {
        "Sid": "iot-events",

```

```

        "Effect": "Allow",
        "Principal": {"Service": "iotevents.amazonaws.com"},
        "Action": "lambda:InvokeFunction",
        "Resource": "arn:aws:lambda:us-west-2:123456789012:function:my-
function"
    }
]
},
"RevisionId": "93017fc9-59cb-41dc-901b-4845ce4bf668"
}

```

For more information, see [Using Resource-based Policies for AWS Lambda](#) in the *AWS Lambda Developer Guide*.

- For API details, see [GetPolicy](#) in *AWS CLI Command Reference*.

get-provisioned-concurrency-config

The following code example shows how to use `get-provisioned-concurrency-config`.

AWS CLI

To view a provisioned concurrency configuration

The following `get-provisioned-concurrency-config` example displays details for the provisioned concurrency configuration for the BLUE alias of the specified function.

```

aws lambda get-provisioned-concurrency-config \
  --function-name my-function \
  --qualifier BLUE

```

Output:

```

{
  "RequestedProvisionedConcurrentExecutions": 100,
  "AvailableProvisionedConcurrentExecutions": 100,
  "AllocatedProvisionedConcurrentExecutions": 100,
  "Status": "READY",
  "LastModified": "2019-12-31T20:28:49+0000"
}

```

- For API details, see [GetProvisionedConcurrencyConfig](#) in *AWS CLI Command Reference*.

invoke

The following code example shows how to use `invoke`.

AWS CLI

Example 1: To invoke a Lambda function synchronously

The following `invoke` example invokes the `my-function` function synchronously. The `cli-binary-format` option is required if you're using AWS CLI version 2. For more information, see [AWS CLI supported global command line options](#) in the *AWS Command Line Interface User Guide*.

```
aws lambda invoke \  
  --function-name my-function \  
  --cli-binary-format raw-in-base64-out \  
  --payload '{ "name": "Bob" }' \  
  response.json
```

Output:

```
{  
  "ExecutedVersion": "$LATEST",  
  "StatusCode": 200  
}
```

For more information, see [Synchronous Invocation](#) in the *AWS Lambda Developer Guide*.

Example 2: To invoke a Lambda function asynchronously

The following `invoke` example invokes the `my-function` function asynchronously. The `cli-binary-format` option is required if you're using AWS CLI version 2. For more information, see [AWS CLI supported global command line options](#) in the *AWS Command Line Interface User Guide*.

```
aws lambda invoke \  
  --function-name my-function \  
  --invocation-type Event \  
  --cli-binary-format raw-in-base64-out \  
  --payload '{ "name": "Bob" }' \  
  response.json
```

Output:

```
{
  "StatusCode": 202
}
```

For more information, see [Asynchronous Invocation](#) in the *AWS Lambda Developer Guide*.

- For API details, see [Invoke](#) in *AWS CLI Command Reference*.

list-aliases

The following code example shows how to use `list-aliases`.

AWS CLI**To retrieve the list of aliases for a Lambda function**

The following `list-aliases` example displays a list of the aliases for the `my-function` Lambda function.

```
aws lambda list-aliases \
  --function-name my-function
```

Output:

```
{
  "Aliases": [
    {
      "AliasArn": "arn:aws:lambda:us-west-2:123456789012:function:my-
function:BETA",
      "RevisionId": "a410117f-ab16-494e-8035-7e204bb7933b",
      "FunctionVersion": "2",
      "Name": "BETA",
      "Description": "alias for beta version of function"
    },
    {
      "AliasArn": "arn:aws:lambda:us-west-2:123456789012:function:my-
function:LIVE",
      "RevisionId": "21d40116-f8b1-40ba-9360-3ea284da1bb5",
      "FunctionVersion": "1",
      "Name": "LIVE",
      "Description": "alias for live version of function"
    }
  ]
}
```

```
    }
  ]
}
```

For more information, see [Configuring AWS Lambda Function Aliases](#) in the *AWS Lambda Developer Guide*.

- For API details, see [ListAliases](#) in *AWS CLI Command Reference*.

list-event-source-mappings

The following code example shows how to use `list-event-source-mappings`.

AWS CLI

To list the event source mappings for a function

The following `list-event-source-mappings` example displays a list of the event source mappings for the `my-function` Lambda function.

```
aws lambda list-event-source-mappings \
  --function-name my-function
```

Output:

```
{
  "EventSourceMappings": [
    {
      "UUID": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
      "StateTransitionReason": "USER_INITIATED",
      "LastModified": 1569284520.333,
      "BatchSize": 5,
      "State": "Enabled",
      "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-
function",
      "EventSourceArn": "arn:aws:sqs:us-west-2:123456789012:mySQSqueue"
    }
  ]
}
```

For more information, see [AWS Lambda Event Source Mapping](#) in the *AWS Lambda Developer Guide*.

- For API details, see [ListEventSourceMappings](#) in *AWS CLI Command Reference*.

list-function-event-invoke-configs

The following code example shows how to use `list-function-event-invoke-configs`.

AWS CLI

To view a list of asynchronous invocation configurations

The following `list-function-event-invoke-configs` example lists the asynchronous invocation configurations for the specified function.

```
aws lambda list-function-event-invoke-configs \  
  --function-name my-function
```

Output:

```
{  
  "FunctionEventInvokeConfigs": [  
    {  
      "LastModified": 1577824406.719,  
      "FunctionArn": "arn:aws:lambda:us-east-2:123456789012:function:my-  
function:GREEN",  
      "MaximumRetryAttempts": 2,  
      "MaximumEventAgeInSeconds": 1800  
    },  
    {  
      "LastModified": 1577824396.653,  
      "FunctionArn": "arn:aws:lambda:us-east-2:123456789012:function:my-  
function:BLUE",  
      "MaximumRetryAttempts": 0,  
      "MaximumEventAgeInSeconds": 3600  
    }  
  ]  
}
```

- For API details, see [ListFunctionEventInvokeConfigs](#) in *AWS CLI Command Reference*.

list-functions

The following code example shows how to use `list-functions`.

AWS CLI

To retrieve a list of Lambda functions

The following `list-functions` example displays a list of all of the functions for the current user.

```
aws lambda list-functions
```

Output:

```
{
  "Functions": [
    {
      "TracingConfig": {
        "Mode": "PassThrough"
      },
      "Version": "$LATEST",
      "CodeSha256": "dBG9m8SGdmlEjw/JYXlhhvCrAv5TxvXsbl/RM10fT/I=",
      "FunctionName": "helloworld",
      "MemorySize": 128,
      "RevisionId": "1718e831-badf-4253-9518-d0644210af7b",
      "CodeSize": 294,
      "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:helloworld",
      "Handler": "helloworld.handler",
      "Role": "arn:aws:iam::123456789012:role/service-role/MyTestFunction-role-zgur6bf4",
      "Timeout": 3,
      "LastModified": "2023-09-23T18:32:33.857+0000",
      "Runtime": "nodejs18.x",
      "Description": ""
    },
    {
      "TracingConfig": {
        "Mode": "PassThrough"
      },
      "Version": "$LATEST",
      "CodeSha256": "sU0cJ2/h0ZevwV/1TxCuQqK3gDZP3i8gUoqUUVrMvY6E=",
      "FunctionName": "my-function",
      "VpcConfig": {
        "SubnetIds": [],
        "VpcId": ""
      }
    }
  ]
}
```

```

        "SecurityGroupIds": [],
    },
    "MemorySize": 256,
    "RevisionId": "93017fc9-59cb-41dc-901b-4845ce4bf668",
    "CodeSize": 266,
    "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-
function",
    "Handler": "index.handler",
    "Role": "arn:aws:iam::123456789012:role/service-role/helloWorldPython-
role-uy3l9qyq",
    "Timeout": 3,
    "LastModified": "2023-10-01T16:47:28.490+0000",
    "Runtime": "nodejs18.x",
    "Description": ""
},
{
    "Layers": [
        {
            "CodeSize": 41784542,
            "Arn": "arn:aws:lambda:us-west-2:420165488524:layer:AWSLambda-
Python37-SciPy1x:2"
        },
        {
            "CodeSize": 4121,
            "Arn": "arn:aws:lambda:us-
west-2:123456789012:layer:pythonLayer:1"
        }
    ],
    "TracingConfig": {
        "Mode": "PassThrough"
    },
    "Version": "$LATEST",
    "CodeSha256": "ZQukCqxtkqFgyF2cU41Avj99TKQ/hNihPtDtRcc08mI=",
    "FunctionName": "my-python-function",
    "VpcConfig": {
        "SubnetIds": [],
        "VpcId": "",
        "SecurityGroupIds": []
    },
    "MemorySize": 128,
    "RevisionId": "80b4eabc-acf7-4ea8-919a-e874c213707d",
    "CodeSize": 299,
    "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-
python-function",

```

```
        "Handler": "lambda_function.lambda_handler",
        "Role": "arn:aws:iam::123456789012:role/service-role/my-python-function-
role-z5g7dr6n",
        "Timeout": 3,
        "LastModified": "2023-10-01T19:40:41.643+0000",
        "Runtime": "python3.11",
        "Description": ""
    }
]
}
```

For more information, see [AWS Lambda Function Configuration](#) in the *AWS Lambda Developer Guide*.

- For API details, see [ListFunctions](#) in *AWS CLI Command Reference*.

list-layer-versions

The following code example shows how to use `list-layer-versions`.

AWS CLI

To list the versions of an AWS Lambda layer

The following `list-layers-versions` example displays information about the versions for the layer named `my-layer`.

```
aws lambda list-layer-versions \
    --layer-name my-layer
```

Output:

```
{
  "Layers": [
    {
      "LayerVersionArn": "arn:aws:lambda:us-east-2:123456789012:layer:my-
layer:2",
      "Version": 2,
      "Description": "My layer",
      "CreateDate": "2023-11-15T00:37:46.592+0000",
      "CompatibleRuntimes": [
        "python3.10",
```

```
        "python3.11"  
      ]  
    }  
  ]  
}
```

For more information, see [AWS Lambda Layers](#) in the *AWS Lambda Developer Guide*.

- For API details, see [ListLayerVersions](#) in *AWS CLI Command Reference*.

list-layers

The following code example shows how to use `list-layers`.

AWS CLI

To list the layers that are compatible with your function's runtime

The following `list-layers` example displays information about layers that are compatible with the Python 3.11 runtime.

```
aws lambda list-layers \  
  --compatible-runtime python3.11
```

Output:

```
{  
  "Layers": [  
    {  
      "LayerName": "my-layer",  
      "LayerArn": "arn:aws:lambda:us-east-2:123456789012:layer:my-layer",  
      "LatestMatchingVersion": {  
        "LayerVersionArn": "arn:aws:lambda:us-east-2:123456789012:layer:my-  
layer:2",  
        "Version": 2,  
        "Description": "My layer",  
        "CreateDate": "2023-11-15T00:37:46.592+0000",  
        "CompatibleRuntimes": [  
          "python3.10",  
          "python3.11"  
        ]  
      }  
    }  
  ]  
}
```

```
]
}
```

For more information, see [AWS Lambda Layers](#) in the *AWS Lambda Developer Guide*.

- For API details, see [ListLayers](#) in *AWS CLI Command Reference*.

list-provisioned-concurrency-configs

The following code example shows how to use `list-provisioned-concurrency-configs`.

AWS CLI

To get a list of provisioned concurrency configurations

The following `list-provisioned-concurrency-configs` example lists the provisioned concurrency configurations for the specified function.

```
aws lambda list-provisioned-concurrency-configs \
  --function-name my-function
```

Output:

```
{
  "ProvisionedConcurrencyConfigs": [
    {
      "FunctionArn": "arn:aws:lambda:us-east-2:123456789012:function:my-
function:GREEN",
      "RequestedProvisionedConcurrentExecutions": 100,
      "AvailableProvisionedConcurrentExecutions": 100,
      "AllocatedProvisionedConcurrentExecutions": 100,
      "Status": "READY",
      "LastModified": "2019-12-31T20:29:00+0000"
    },
    {
      "FunctionArn": "arn:aws:lambda:us-east-2:123456789012:function:my-
function:BLUE",
      "RequestedProvisionedConcurrentExecutions": 100,
      "AvailableProvisionedConcurrentExecutions": 100,
      "AllocatedProvisionedConcurrentExecutions": 100,
      "Status": "READY",
      "LastModified": "2019-12-31T20:28:49+0000"
    }
  ]
}
```

```
]
}
```

- For API details, see [ListProvisionedConcurrencyConfigs](#) in *AWS CLI Command Reference*.

list-tags

The following code example shows how to use `list-tags`.

AWS CLI

To retrieve the list of tags for a Lambda function

The following `list-tags` example displays the tags attached to the `my-function` Lambda function.

```
aws lambda list-tags \
  --resource arn:aws:lambda:us-west-2:123456789012:function:my-function
```

Output:

```
{
  "Tags": {
    "Category": "Web Tools",
    "Department": "Sales"
  }
}
```

For more information, see [Tagging Lambda Functions](#) in the *AWS Lambda Developer Guide*.

- For API details, see [ListTags](#) in *AWS CLI Command Reference*.

list-versions-by-function

The following code example shows how to use `list-versions-by-function`.

AWS CLI

To retrieve a list of versions of a function

The following `list-versions-by-function` example displays the list of versions for the `my-function` Lambda function.

```
aws lambda list-versions-by-function \  
  --function-name my-function
```

Output:

```
{  
  "Versions": [  
    {  
      "TracingConfig": {  
        "Mode": "PassThrough"  
      },  
      "Version": "$LATEST",  
      "CodeSha256": "sU0cJ2/h0ZevwV/1TxCuQqK3gDZP3i8gUoqUUVRmY6E=",  
      "FunctionName": "my-function",  
      "VpcConfig": {  
        "SubnetIds": [],  
        "VpcId": "",  
        "SecurityGroupIds": []  
      },  
      "MemorySize": 256,  
      "RevisionId": "93017fc9-59cb-41dc-901b-4845ce4bf668",  
      "CodeSize": 266,  
      "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-  
function:$LATEST",  
      "Handler": "index.handler",  
      "Role": "arn:aws:iam::123456789012:role/service-role/helloWorldPython-  
role-uy3l9qqq",  
      "Timeout": 3,  
      "LastModified": "2019-10-01T16:47:28.490+0000",  
      "Runtime": "nodejs10.x",  
      "Description": ""  
    },  
    {  
      "TracingConfig": {  
        "Mode": "PassThrough"  
      },  
      "Version": "1",  
      "CodeSha256": "5tT2qgzYUHoqwR616pZ2dpkn/0J1FrzJmlKidWaaCgk=",  
      "FunctionName": "my-function",  
      "VpcConfig": {  
        "SubnetIds": [],  
        "VpcId": "",  
        "SecurityGroupIds": []  
      }  
    }  
  ]  
}
```



```
    },
    "MemorySize": 256,
    "RevisionId": "949c8914-012e-4795-998c-e467121951b1",
    "CodeSize": 304,
    "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-
function:1",
    "Handler": "index.handler",
    "Role": "arn:aws:iam::123456789012:role/service-role/helloWorldPython-
role-uy3l9qyq",
    "Timeout": 3,
    "LastModified": "2019-09-26T20:28:40.438+0000",
    "Runtime": "nodejs10.x",
    "Description": "new version"
  },
  {
    "TracingConfig": {
      "Mode": "PassThrough"
    },
    "Version": "2",
    "CodeSha256": "sU0cJ2/h0ZevwV/1TxCuQqK3gDZP3i8gUoqUUVRmY6E=",
    "FunctionName": "my-function",
    "VpcConfig": {
      "SubnetIds": [],
      "VpcId": "",
      "SecurityGroupIds": []
    },
    "MemorySize": 256,
    "RevisionId": "cd669f21-0f3d-4e1c-9566-948837f2e2ea",
    "CodeSize": 266,
    "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-
function:2",
    "Handler": "index.handler",
    "Role": "arn:aws:iam::123456789012:role/service-role/helloWorldPython-
role-uy3l9qyq",
    "Timeout": 3,
    "LastModified": "2019-10-01T16:47:28.490+0000",
    "Runtime": "nodejs10.x",
    "Description": "newer version"
  }
]
}
```

For more information, see [Configuring AWS Lambda Function Aliases](#) in the *AWS Lambda Developer Guide*.

- For API details, see [ListVersionsByFunction](#) in *AWS CLI Command Reference*.

publish-layer-version

The following code example shows how to use `publish-layer-version`.

AWS CLI

To create a Lambda layer version

The following `publish-layer-version` example creates a new Python library layer version. The command retrieves the layer content a file named `layer.zip` in the specified S3 bucket.

```
aws lambda publish-layer-version \  
  --layer-name my-layer \  
  --description "My Python layer" \  
  --license-info "MIT" \  
  --content S3Bucket=lambda-layers-us-west-2-123456789012,S3Key=layer.zip \  
  --compatible-runtimes python3.10 python3.11
```

Output:

```
{  
  "Content": {  
    "Location": "https://awslambda-us-west-2-layers.s3.us-west-2.amazonaws.com/  
snapshots/123456789012/my-layer-4aaa2fbb-ff77-4b0a-ad92-5b78a716a96a?  
versionId=27iWyA73cCAYqyH...",  
    "CodeSha256": "tv9jJ0+rPbXUUXuRKi7CwHzKtLDkDRJLB3cC3Z/ouXo=",  
    "CodeSize": 169  
  },  
  "LayerArn": "arn:aws:lambda:us-west-2:123456789012:layer:my-layer",  
  "LayerVersionArn": "arn:aws:lambda:us-west-2:123456789012:layer:my-layer:1",  
  "Description": "My Python layer",  
  "CreateDate": "2023-11-14T23:03:52.894+0000",  
  "Version": 1,  
  "LicenseInfo": "MIT",  
  "CompatibleRuntimes": [  
    "python3.10",  
    "python3.11"  
  ]  
}
```

```
]
}
```

For more information, see [AWS Lambda Layers](#) in the *AWS Lambda Developer Guide*.

- For API details, see [PublishLayerVersion](#) in *AWS CLI Command Reference*.

publish-version

The following code example shows how to use `publish-version`.

AWS CLI

To publish a new version of a function

The following `publish-version` example publishes a new version of the `my-function` Lambda function.

```
aws lambda publish-version \  
  --function-name my-function
```

Output:

```
{  
  "TracingConfig": {  
    "Mode": "PassThrough"  
  },  
  "CodeSha256": "dBG9m8SGdm1Ejw/JYX1hhvCrAv5TxvXsbl/RM1r0fT/I=",  
  "FunctionName": "my-function",  
  "CodeSize": 294,  
  "RevisionId": "f31d3d39-cc63-4520-97d4-43cd44c94c20",  
  "MemorySize": 128,  
  "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function:3",  
  "Version": "2",  
  "Role": "arn:aws:iam::123456789012:role/service-role/MyTestFunction-role-zgur6bf4",  
  "Timeout": 3,  
  "LastModified": "2019-09-23T18:32:33.857+0000",  
  "Handler": "my-function.handler",  
  "Runtime": "nodejs10.x",  
  "Description": ""  
}
```

For more information, see [Configuring AWS Lambda Function Aliases](#) in the *AWS Lambda Developer Guide*.

- For API details, see [PublishVersion](#) in *AWS CLI Command Reference*.

put-function-concurrency

The following code example shows how to use `put-function-concurrency`.

AWS CLI

To configure a reserved concurrency limit for a function

The following `put-function-concurrency` example configures 100 reserved concurrent executions for the `my-function` function.

```
aws lambda put-function-concurrency \  
  --function-name my-function \  
  --reserved-concurrent-executions 100
```

Output:

```
{  
  "ReservedConcurrentExecutions": 100  
}
```

For more information, see [Reserving Concurrency for a Lambda Function](#) in the *AWS Lambda Developer Guide*.

- For API details, see [PutFunctionConcurrency](#) in *AWS CLI Command Reference*.

put-function-event-invoke-config

The following code example shows how to use `put-function-event-invoke-config`.

AWS CLI

To configure error handling for asynchronous invocation

The following `put-function-event-invoke-config` example sets a maximum event age of one hour and disables retries for the specified function.

```
aws lambda put-function-event-invoke-config \  
  --function-name my-function \  
  --maximum-event-age-in-seconds 3600 \  
  --maximum-retry-attempts 0
```

Output:

```
{  
  "LastModified": 1573686021.479,  
  "FunctionArn": "arn:aws:lambda:us-east-2:123456789012:function:my-function:  
$LATEST",  
  "MaximumRetryAttempts": 0,  
  "MaximumEventAgeInSeconds": 3600,  
  "DestinationConfig": {  
    "OnSuccess": {},  
    "OnFailure": {}  
  }  
}
```

- For API details, see [PutFunctionEventInvokeConfig](#) in *AWS CLI Command Reference*.

put-provisioned-concurrency-config

The following code example shows how to use `put-provisioned-concurrency-config`.

AWS CLI

To allocate provisioned concurrency

The following `put-provisioned-concurrency-config` example allocates 100 provisioned concurrency for the BLUE alias of the specified function.

```
aws lambda put-provisioned-concurrency-config \  
  --function-name my-function \  
  --qualifier BLUE \  
  --provisioned-concurrent-executions 100
```

Output:

```
{  
  "Requested ProvisionedConcurrentExecutions": 100,
```

```
"Allocated ProvisionedConcurrentExecutions": 0,  
"Status": "IN_PROGRESS",  
"LastModified": "2019-11-21T19:32:12+0000"  
}
```

- For API details, see [PutProvisionedConcurrencyConfig](#) in *AWS CLI Command Reference*.

remove-layer-version-permission

The following code example shows how to use `remove-layer-version-permission`.

AWS CLI

To delete layer-version permissions

The following `remove-layer-version-permission` example deletes permission for an account to configure a layer version.

```
aws lambda remove-layer-version-permission \  
  --layer-name my-layer \  
  --statement-id xaccount \  
  --version-number 1
```

This command produces no output.

For more information, see [AWS Lambda Layers](#) in the *AWS Lambda Developer Guide*.

- For API details, see [RemoveLayerVersionPermission](#) in *AWS CLI Command Reference*.

remove-permission

The following code example shows how to use `remove-permission`.

AWS CLI

To remove permissions from an existing Lambda function

The following `remove-permission` example removes permission to invoke a function named `my-function`.

```
aws lambda remove-permission \  
  --function-name my-function \  
  --statement-id xaccount
```

```
--statement-id sns
```

This command produces no output.

For more information, see [Using Resource-based Policies for AWS Lambda](#) in the *AWS Lambda Developer Guide*.

- For API details, see [RemovePermission](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add tags to an existing Lambda function

The following `tag-resource` example adds a tag with the key name `DEPARTMENT` and a value of `Department A` to the specified Lambda function.

```
aws lambda tag-resource \  
  --resource arn:aws:lambda:us-west-2:123456789012:function:my-function \  
  --tags "DEPARTMENT=Department A"
```

This command produces no output.

For more information, see [Tagging Lambda Functions](#) in the *AWS Lambda Developer Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from an existing Lambda function

The following `untag-resource` example removes the tag with the key name `DEPARTMENT` tag from the `my-function` Lambda function.

```
aws lambda untag-resource \  
  --resource arn:aws:lambda:us-west-2:123456789012:function:my-function \  
  --tag-key DEPARTMENT
```

```
--tag-keys DEPARTMENT
```

This command produces no output.

For more information, see [Tagging Lambda Functions](#) in the *AWS Lambda Developer Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-alias

The following code example shows how to use `update-alias`.

AWS CLI

To update a function alias

The following `update-alias` example updates the alias named `LIVE` to point to version 3 of the `my-function` Lambda function.

```
aws lambda update-alias \  
  --function-name my-function \  
  --function-version 3 \  
  --name LIVE
```

Output:

```
{  
  "FunctionVersion": "3",  
  "Name": "LIVE",  
  "AliasArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function:LIVE",  
  "RevisionId": "594f41fb-b85f-4c20-95c7-6ca5f2a92c93",  
  "Description": "alias for live version of function"  
}
```

For more information, see [Configuring AWS Lambda Function Aliases](#) in the *AWS Lambda Developer Guide*.

- For API details, see [UpdateAlias](#) in *AWS CLI Command Reference*.

update-event-source-mapping

The following code example shows how to use `update-event-source-mapping`.

AWS CLI

To update the mapping between an event source and an AWS Lambda function

The following `update-event-source-mapping` example updates the batch size to 8 in the specified mapping.

```
aws lambda update-event-source-mapping \  
  --uuid "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE" \  
  --batch-size 8
```

Output:

```
{  
  "UUID": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",  
  "StateTransitionReason": "USER_INITIATED",  
  "LastModified": 1569284520.333,  
  "BatchSize": 8,  
  "State": "Updating",  
  "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function",  
  "EventSourceArn": "arn:aws:sqs:us-west-2:123456789012:mySQSqueue"  
}
```

For more information, see [AWS Lambda Event Source Mapping](#) in the *AWS Lambda Developer Guide*.

- For API details, see [UpdateEventSourceMapping](#) in *AWS CLI Command Reference*.

update-function-code

The following code example shows how to use `update-function-code`.

AWS CLI

To update the code of a Lambda function

The following `update-function-code` example replaces the code of the unpublished (\$LATEST) version of the `my-function` function with the contents of the specified zip file.

```
aws lambda update-function-code \  
  --function-name my-function \  
  --zip-file file://my-function.zip
```

```
--zip-file fileb://my-function.zip
```

Output:

```
{
  "FunctionName": "my-function",
  "LastModified": "2019-09-26T20:28:40.438+0000",
  "RevisionId": "e52502d4-9320-4688-9cd6-152a6ab7490d",
  "MemorySize": 256,
  "Version": "$LATEST",
  "Role": "arn:aws:iam::123456789012:role/service-role/my-function-role-uy3l9qq",
  "Timeout": 3,
  "Runtime": "nodejs10.x",
  "TracingConfig": {
    "Mode": "PassThrough"
  },
  "CodeSha256": "5tT2qgzYUHaqwR716pZ2dpkn/0J1FrzJmlKidWoaCgk=",
  "Description": "",
  "VpcConfig": {
    "SubnetIds": [],
    "VpcId": "",
    "SecurityGroupIds": []
  },
  "CodeSize": 304,
  "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function",
  "Handler": "index.handler"
}
```

For more information, see [AWS Lambda Function Configuration](#) in the *AWS Lambda Developer Guide*.

- For API details, see [UpdateFunctionCode](#) in *AWS CLI Command Reference*.

update-function-configuration

The following code example shows how to use `update-function-configuration`.

AWS CLI

To modify the configuration of a function

The following `update-function-configuration` example modifies the memory size to be 256 MB for the unpublished (`$LATEST`) version of the `my-function` function.

```
aws lambda update-function-configuration \  
  --function-name my-function \  
  --memory-size 256
```

Output:

```
{  
  "FunctionName": "my-function",  
  "LastModified": "2019-09-26T20:28:40.438+0000",  
  "RevisionId": "e52502d4-9320-4688-9cd6-152a6ab7490d",  
  "MemorySize": 256,  
  "Version": "$LATEST",  
  "Role": "arn:aws:iam::123456789012:role/service-role/my-function-role-uy3l9qqq",  
  "Timeout": 3,  
  "Runtime": "nodejs10.x",  
  "TracingConfig": {  
    "Mode": "PassThrough"  
  },  
  "CodeSha256": "5tT2qqzYUHaqwR716pZ2dpkn/0J1FrzJmLKidWoaCgk=",  
  "Description": "",  
  "VpcConfig": {  
    "SubnetIds": [],  
    "VpcId": "",  
    "SecurityGroupIds": []  
  },  
  "CodeSize": 304,  
  "FunctionArn": "arn:aws:lambda:us-west-2:123456789012:function:my-function",  
  "Handler": "index.handler"  
}
```

For more information, see [AWS Lambda Function Configuration](#) in the *AWS Lambda Developer Guide*.

- For API details, see [UpdateFunctionConfiguration](#) in *AWS CLI Command Reference*.

update-function-event-invoke-config

The following code example shows how to use `update-function-event-invoke-config`.

AWS CLI

To update an asynchronous invocation configuration

The following `update-function-event-invoke-config` example adds an on-failure destination to the existing asynchronous invocation configuration for the specified function.

```
aws lambda update-function-event-invoke-config \  
  --function-name my-function \  
  --destination-config '{"OnFailure":{"Destination": "arn:aws:sqs:us-  
east-2:123456789012:destination"}}'
```

Output:

```
{  
  "LastModified": 1573687896.493,  
  "FunctionArn": "arn:aws:lambda:us-east-2:123456789012:function:my-function:  
$LATEST",  
  "MaximumRetryAttempts": 0,  
  "MaximumEventAgeInSeconds": 3600,  
  "DestinationConfig": {  
    "OnSuccess": {},  
    "OnFailure": {  
      "Destination": "arn:aws:sqs:us-east-2:123456789012:destination"  
    }  
  }  
}
```

- For API details, see [UpdateFunctionEventInvokeConfig](#) in *AWS CLI Command Reference*.

License Manager examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with License Manager.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-license-configuration

The following code example shows how to use `create-license-configuration`.

AWS CLI

Example 1: To create a license configuration

The following `create-license-configuration` example creates a license configuration with a hard limit of 10 cores.

```
aws license-manager create-license-configuration --name my-license-configuration \  
  --license-counting-type Core \  
  --license-count 10 \  
  --license-count-hard-limit
```

Output:

```
{  
  "LicenseConfigurationArn": "arn:aws:license-manager:us-  
west-2:123456789012:license-configuration:lic-6eb6586f508a786a2ba41EXAMPLE1111"  
}
```

Example 2: To create a license configuration

The following `create-license-configuration` example creates a license configuration with a soft limit of 100 vCPUs. It uses a rule to enable vCPU optimization.

```
aws license-manager create-license-configuration --name my-license-configuration \  
  --license-counting-type vCPU \  
  --license-count 100 \  
  --license-rules "#honorVcpuOptimization=true"
```

Output:

```
{
  "LicenseConfigurationArn": "arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-6eb6586f508a786a2ba41EXAMPLE2222"
}
```

- For API details, see [CreateLicenseConfiguration](#) in *AWS CLI Command Reference*.

delete-license-configuration

The following code example shows how to use `delete-license-configuration`.

AWS CLI

To delete a license configuration

The following `delete-license-configuration` example deletes the specified license configuration.

```
aws license-manager delete-license-configuration \
  --license-configuration-arn arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-6eb6586f508a786a2ba4f56c1EXAMPLE
```

This command produces no output.

- For API details, see [DeleteLicenseConfiguration](#) in *AWS CLI Command Reference*.

get-license-configuration

The following code example shows how to use `get-license-configuration`.

AWS CLI

To get license configuration information

The following `get-license-configuration` example displays details for the specified license configuration.

```
aws license-manager get-license-configuration \
  --license-configuration-arn arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-38b658717b87478aaa7c00883EXAMPLE
```

Output:

```
{
  "LicenseConfigurationId": "lic-38b658717b87478aaa7c00883EXAMPLE",
  "LicenseConfigurationArn": "arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-38b658717b87478aaa7c00883EXAMPLE",
  "Name": "my-license-configuration",
  "LicenseCountingType": "vCPU",
  "LicenseRules": [],
  "LicenseCountHardLimit": false,
  "ConsumedLicenses": 0,
  "Status": "AVAILABLE",
  "OwnerAccountId": "123456789012",
  "ConsumedLicenseSummaryList": [
    {
      "ResourceType": "EC2_INSTANCE",
      "ConsumedLicenses": 0
    },
    {
      "ResourceType": "EC2_HOST",
      "ConsumedLicenses": 0
    },
    {
      "ResourceType": "SYSTEMS_MANAGER_MANAGED_INSTANCE",
      "ConsumedLicenses": 0
    }
  ],
  "ManagedResourceSummaryList": [
    {
      "ResourceType": "EC2_INSTANCE",
      "AssociationCount": 0
    },
    {
      "ResourceType": "EC2_HOST",
      "AssociationCount": 0
    },
    {
      "ResourceType": "EC2_AMI",
      "AssociationCount": 2
    },
    {
      "ResourceType": "SYSTEMS_MANAGER_MANAGED_INSTANCE",
      "AssociationCount": 0
    }
  ]
}
```

```
]
}
```

- For API details, see [GetLicenseConfiguration](#) in *AWS CLI Command Reference*.

get-service-settings

The following code example shows how to use `get-service-settings`.

AWS CLI

To get the License Manager settings

The following `get-service-settings` example displays the service settings for License Manager in the current Region.

```
aws license-manager get-service-settings
```

The following shows example output if cross-account resource discovery is disabled.

```
{
  "OrganizationConfiguration": {
    "EnableIntegration": false
  },
  "EnableCrossAccountsDiscovery": false
}
```

The following shows example output if cross-account resource discovery is enabled.

```
{
  "S3BucketArn": "arn:aws:s3:::aws-license-manager-service-c22d6279-35c4-47c4-bb",
  "OrganizationConfiguration": {
    "EnableIntegration": true
  },
  "EnableCrossAccountsDiscovery": true
}
```

- For API details, see [GetServiceSettings](#) in *AWS CLI Command Reference*.

list-associations-for-license-configuration

The following code example shows how to use `list-associations-for-license-configuration`.

AWS CLI

To get associations for a license configuration

The following `list-associations-for-license-configuration` example displays detailed information for the associations of the specified license configuration.

```
aws license-manager list-associations-for-license-configuration \
  --license-configuration-arn arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-38b658717b87478aaa7c00883EXAMPLE
```

Output:

```
{
  "LicenseConfigurationAssociations": [
    {
      "ResourceArn": "arn:aws:ec2:us-west-2::image/ami-1234567890abcdef0",
      "ResourceType": "EC2_AMI",
      "ResourceOwnerId": "123456789012",
      "AssociationTime": 1568825118.617
    },
    {
      "ResourceArn": "arn:aws:ec2:us-west-2::image/ami-0abcdef1234567890",
      "ResourceType": "EC2_AMI",
      "ResourceOwnerId": "123456789012",
      "AssociationTime": 1568825118.946
    }
  ]
}
```

- For API details, see [ListAssociationsForLicenseConfiguration](#) in *AWS CLI Command Reference*.

list-license-configurations

The following code example shows how to use `list-license-configurations`.

AWS CLI

Example 1: To list all of your license configurations

The following `list-license-configurations` example lists all your license configurations.

```
aws license-manager list-license-configurations
```

Output:

```
{
  "LicenseConfigurations": [
    {
      "LicenseConfigurationId": "lic-6eb6586f508a786a2ba4f56c1EXAMPLE",
      "LicenseConfigurationArn": "arn:aws:license-manager:us-west-2:123456789012:license-configuration:lic-6eb6586f508a786a2ba4f56c1EXAMPLE",
      "Name": "my-license-configuration",
      "LicenseCountingType": "Core",
      "LicenseRules": [],
      "LicenseCount": 10,
      "LicenseCountHardLimit": true,
      "ConsumedLicenses": 0,
      "Status": "AVAILABLE",
      "OwnerAccountId": "123456789012",
      "ConsumedLicenseSummaryList": [
        {
          "ResourceType": "EC2_INSTANCE",
          "ConsumedLicenses": 0
        },
        {
          "ResourceType": "EC2_HOST",
          "ConsumedLicenses": 0
        },
        {
          "ResourceType": "SYSTEMS_MANAGER_MANAGED_INSTANCE",
          "ConsumedLicenses": 0
        }
      ],
      "ManagedResourceSummaryList": [
        {
          "ResourceType": "EC2_INSTANCE",
          "AssociationCount": 0
        }
      ]
    }
  ]
}
```

```

    {
      "ResourceType": "EC2_HOST",
      "AssociationCount": 0
    },
    {
      "ResourceType": "EC2_AMI",
      "AssociationCount": 0
    },
    {
      "ResourceType": "SYSTEMS_MANAGER_MANAGED_INSTANCE",
      "AssociationCount": 0
    }
  ]
},
{
  ...
}
]
}

```

Example 2: To list a specific license configuration

The following `list-license-configurations` example lists only the specified license configuration.

```

aws license-manager list-license-configurations \
  --license-configuration-arns arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-38b658717b87478aaa7c00883EXAMPLE

```

- For API details, see [ListLicenseConfigurations](#) in *AWS CLI Command Reference*.

list-license-specifications-for-resource

The following code example shows how to use `list-license-specifications-for-resource`.

AWS CLI

To list the license configurations for a resource

The following `list-license-specifications-for-resource` example lists the license configurations associated with the specified Amazon Machine Image (AMI).

```
aws license-manager list-license-specifications-for-resource \  
  --resource-arn arn:aws:ec2:us-west-2::image/ami-1234567890abcdef0
```

Output:

```
{  
  "LicenseConfigurationArn": "arn:aws:license-manager:us-  
west-2:123456789012:license-configuration:lic-38b658717b87478aaa7c00883EXAMPLE"  
}
```

- For API details, see [ListLicenseSpecificationsForResource](#) in *AWS CLI Command Reference*.

list-resource-inventory

The following code example shows how to use `list-resource-inventory`.

AWS CLI

To list resources in the resource inventory

The following `list-resource-inventory` example lists the resources managed using Systems Manager inventory.

```
aws license-manager list-resource-inventory
```

Output:

```
{  
  "ResourceInventoryList": [  
    {  
      "Platform": "Red Hat Enterprise Linux Server",  
      "ResourceType": "EC2Instance",  
      "PlatformVersion": "7.4",  
      "ResourceArn": "arn:aws:ec2:us-west-2:1234567890129:instance/  
i-05d3cdfb05bd36376",  
      "ResourceId": "i-05d3cdfb05bd36376",  
      "ResourceOwningAccountId": "1234567890129"  
    },  
    {  
      "Platform": "Amazon Linux",
```

```

        "ResourceType": "EC2Instance",
        "PlatformVersion": "2",
        "ResourceArn": "arn:aws:ec2:us-west-2:1234567890129:instance/
i-0b1d036cfd4594808",
        "ResourceId": "i-0b1d036cfd4594808",
        "ResourceOwningAccountId": "1234567890129"
    },
    {
        "Platform": "Microsoft Windows Server 2019 Datacenter",
        "ResourceType": "EC2Instance",
        "PlatformVersion": "10.0.17763",
        "ResourceArn": "arn:aws:ec2:us-west-2:1234567890129:instance/
i-0cdb3b54a2a8246ad",
        "ResourceId": "i-0cdb3b54a2a8246ad",
        "ResourceOwningAccountId": "1234567890129"
    }
]
}

```

- For API details, see [ListResourceInventory](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags for a license configuration

The following `list-tags-for-resource` example lists the tags for the specified license configuration.

```

aws license-manager list-tags-for-resource \
  --resource-arn arn:aws:license-manager:us-west-2:123456789012:license-
configuration:lic-6eb6586f508a786a2ba4f56c1EXAMPLE

```

Output:

```

{
  "Tags": [
    {

```

```

        "Key": "project",
        "Value": "lima"
      }
    ]
  }

```

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

list-usage-for-license-configuration

The following code example shows how to use `list-usage-for-license-configuration`.

AWS CLI

To list the licenses in use for a license configuration

The following `list-usage-for-license-configuration` example lists information about the resources using licenses for the specified license configuration. For example, if the license type is vCPU, any instances consume one license per vCPU.

```

aws license-manager list-usage-for-license-configuration \
  --license-configuration-arn arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-38b658717b87478aaa7c00883EXAMPLE

```

Output:

```

{
  "LicenseConfigurationUsageList": [
    {
      "ResourceArn": "arn:aws:ec2:us-west-2:123456789012:instance/
i-04a636d18e83cfac",
      "ResourceType": "EC2_INSTANCE",
      "ResourceStatus": "running",
      "ResourceOwnerId": "123456789012",
      "AssociationTime": 1570892850.519,
      "ConsumedLicenses": 2
    }
  ]
}

```

- For API details, see [ListUsageForLicenseConfiguration](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add a tag a license configuration

The following `tag-resource` example adds the specified tag (key name and value) to the specified license configuration.

```
aws license-manager tag-resource \  
  --tags Key=project,Value=lima \  
  --resource-arn arn:aws:license-manager:us-west-2:123456789012:license-  
configuration:lic-6eb6586f508a786a2ba4f56c1EXAMPLE
```

This command produces no output.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from a license configuration

The following `untag-resource` example removes the specified tag (key name and resource) from the specified license configuration.

```
aws license-manager untag-resource \  
  --tag-keys project \  
  --resource-arn arn:aws:license-manager:us-west-2:123456789012:license-  
configuration:lic-6eb6586f508a786a2ba4f56c1EXAMPLE
```

This command produces no output.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-license-configuration

The following code example shows how to use `update-license-configuration`.

AWS CLI

To update a license configuration

The following `update-license-configuration` example updates the specified license configuration to remove the hard limit.

```
aws license-manager update-license-configuration \  
  --no-license-count-hard-limit \  
  --license-configuration-arn arn:aws:license-manager:us-  
west-2:880185128111:license-configuration:lic-6eb6586f508a786a2ba4f56c1EXAMPLE
```

This command produces no output.

The following `update-license-configuration` example updates the specified license configuration to change its status to `DISABLED`.

```
aws license-manager update-license-configuration \  
  --license-configuration-status DISABLED  
  --license-configuration-arn arn:aws:license-manager:us-  
west-2:880185128111:license-configuration:lic-6eb6586f508a786a2ba4f56c1EXAMPLE
```

This command produces no output.

- For API details, see [UpdateLicenseConfiguration](#) in *AWS CLI Command Reference*.

update-license-specifications-for-resource

The following code example shows how to use `update-license-specifications-for-resource`.

AWS CLI

To update the license configurations for a resource

The following `update-license-specifications-for-resource` example replaces the license configuration associated with the specified Amazon Machine Image (AMI) by removing one license configuration and adding another.

```
aws license-manager update-license-specifications-for-resource \  
  --resource-arn arn:aws:ec2:us-west-2::image/ami-1234567890abcdef0 \  
  --license-configuration-arn arn:aws:license-manager:us-west-2:880185128111:license-configuration:lic-6eb6586f508a786a2ba4f56c1EXAMPLE
```



```
--remove-license-specifications LicenseConfigurationArn=arn:aws:license-
manager:us-west-2:123456789012:license-
configuration:lic-38b658717b87478aaa7c00883EXAMPLE \
--add-license-specifications LicenseConfigurationArn=arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-42b6deb06e5399a980d555927EXAMPLE
```

This command produces no output.

- For API details, see [UpdateLicenseSpecificationsForResource](#) in *AWS CLI Command Reference*.

update-service-settings

The following code example shows how to use `update-service-settings`.

AWS CLI

To update the License Manager settings

The following `update-service-settings` example enables cross-account resource discovery for License Manager in the current AWS Region. The Amazon S3 bucket is the Resource Data Sync required for Systems Manager inventory.

```
aws license-manager update-service-settings \
--organization-configuration EnableIntegration=true \
--enable-cross-accounts-discovery \
--s3-bucket-arn arn:aws:s3:::aws-license-manager-service-abcd1234EXAMPLE
```

This command produces no output.

- For API details, see [UpdateServiceSettings](#) in *AWS CLI Command Reference*.

Lightsail examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Lightsail.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

allocate-static-ip

The following code example shows how to use `allocate-static-ip`.

AWS CLI

To create a static IP

The following `allocate-static-ip` example creates the specified static IP, which can be attached to an instance.

```
aws lightsail allocate-static-ip \  
  --static-ip-name StaticIp-1
```

Output:

```
{  
  "operations": [  
    {  
      "id": "b5d06d13-2f19-4683-889f-dEXAMPLEed79",  
      "resourceName": "StaticIp-1",  
      "resourceType": "StaticIp",  
      "createdAt": 1571071325.076,  
      "location": {  
        "availabilityZone": "all",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": true,  
      "operationType": "AllocateStaticIp",  
      "status": "Succeeded",  
      "statusChangedAt": 1571071325.274  
    }  
  ]  
}
```

- For API details, see [AllocateStaticIp](#) in *AWS CLI Command Reference*.

attach-disk

The following code example shows how to use attach-disk.

AWS CLI

To attach a block storage disk to an instance

The following attach-disk example attaches disk Disk-1 to instance WordPress_Multisite-1 with the disk path of /dev/xvdf

```
aws lightsail attach-disk \  
  --disk-name Disk-1 \  
  --disk-path /dev/xvdf \  
  --instance-name WordPress_Multisite-1
```

Output:

```
{  
  "operations": [  
    {  
      "id": "10a08267-19ce-43be-b913-6EXAMPLE7e80",  
      "resourceName": "Disk-1",  
      "resourceType": "Disk",  
      "createdAt": 1571071465.472,  
      "location": {  
        "availabilityZone": "us-west-2a",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": false,  
      "operationDetails": "WordPress_Multisite-1",  
      "operationType": "AttachDisk",  
      "status": "Started",  
      "statusChangedAt": 1571071465.472  
    },  
    {  
      "id": "2912c477-5295-4539-88c9-bEXAMPLEd1f0",  
      "resourceName": "WordPress_Multisite-1",  
      "resourceType": "Instance",  
      "createdAt": 1571071465.474,  
      "location": {
```

```

        "availabilityZone": "us-west-2a",
        "regionName": "us-west-2"
    },
    "isTerminal": false,
    "operationDetails": "Disk-1",
    "operationType": "AttachDisk",
    "status": "Started",
    "statusChangedAt": 1571071465.474
  }
]
}

```

- For API details, see [AttachDisk](#) in *AWS CLI Command Reference*.

attach-instances-to-load-balancer

The following code example shows how to use `attach-instances-to-load-balancer`.

AWS CLI

To attach instances to a load balancer

The following `attach-instances-to-load-balancer` example attaches instances MEAN-1, MEAN-2, and MEAN-3 to the load balancer LoadBalancer-1.

```

aws lightsail attach-instances-to-load-balancer \
  --instance-names {"MEAN-1","MEAN-2","MEAN-3"} \
  --load-balancer-name LoadBalancer-1

```

Output:

```

{
  "operations": [
    {
      "id": "8055d19d-abb2-40b9-b527-1EXAMPLE3c7b",
      "resourceName": "LoadBalancer-1",
      "resourceType": "LoadBalancer",
      "createdAt": 1571071699.892,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      }
    },
  ],
}

```

```
    "isTerminal": false,
    "operationDetails": "MEAN-2",
    "operationType": "AttachInstancesToLoadBalancer",
    "status": "Started",
    "statusChangedAt": 1571071699.892
  },
  {
    "id": "c35048eb-8538-456a-a118-0EXAMPLEfb73",
    "resourceName": "MEAN-2",
    "resourceType": "Instance",
    "createdAt": 1571071699.887,
    "location": {
      "availabilityZone": "all",
      "regionName": "us-west-2"
    },
    "isTerminal": false,
    "operationDetails": "LoadBalancer-1",
    "operationType": "AttachInstancesToLoadBalancer",
    "status": "Started",
    "statusChangedAt": 1571071699.887
  },
  {
    "id": "910d09e0-adc5-4372-bc2e-0EXAMPLEd891",
    "resourceName": "LoadBalancer-1",
    "resourceType": "LoadBalancer",
    "createdAt": 1571071699.882,
    "location": {
      "availabilityZone": "all",
      "regionName": "us-west-2"
    },
    "isTerminal": false,
    "operationDetails": "MEAN-3",
    "operationType": "AttachInstancesToLoadBalancer",
    "status": "Started",
    "statusChangedAt": 1571071699.882
  },
  {
    "id": "178b18ac-43e8-478c-9bed-1EXAMPLE4755",
    "resourceName": "MEAN-3",
    "resourceType": "Instance",
    "createdAt": 1571071699.901,
    "location": {
      "availabilityZone": "all",
      "regionName": "us-west-2"
    }
  }
}
```

```
    },
    "isTerminal": false,
    "operationDetails": "LoadBalancer-1",
    "operationType": "AttachInstancesToLoadBalancer",
    "status": "Started",
    "statusChangedAt": 1571071699.901
  },
  {
    "id": "fb62536d-2a98-4190-a6fc-4EXAMPLE7470",
    "resourceName": "LoadBalancer-1",
    "resourceType": "LoadBalancer",
    "createdAt": 1571071699.885,
    "location": {
      "availabilityZone": "all",
      "regionName": "us-west-2"
    },
    "isTerminal": false,
    "operationDetails": "MEAN-1",
    "operationType": "AttachInstancesToLoadBalancer",
    "status": "Started",
    "statusChangedAt": 1571071699.885
  },
  {
    "id": "787dac0d-f98d-46c3-8571-3EXAMPLE5a85",
    "resourceName": "MEAN-1",
    "resourceType": "Instance",
    "createdAt": 1571071699.901,
    "location": {
      "availabilityZone": "all",
      "regionName": "us-west-2"
    },
    "isTerminal": false,
    "operationDetails": "LoadBalancer-1",
    "operationType": "AttachInstancesToLoadBalancer",
    "status": "Started",
    "statusChangedAt": 1571071699.901
  }
]
}
```

- For API details, see [AttachInstancesToLoadBalancer](#) in *AWS CLI Command Reference*.

attach-load-balancer-tls-certificate

The following code example shows how to use `attach-load-balancer-tls-certificate`.

AWS CLI

To attach a TLS certificate to a load balancer

The following `attach-load-balancer-tls-certificate` example attaches the load balancer TLS certificate `Certificate2` to the load balancer `LoadBalancer-1`.

```
aws lightsail attach-load-balancer-tls-certificate \  
  --certificate-name Certificate2 \  
  --load-balancer-name LoadBalancer-1
```

Output:

```
{  
  "operations": [  
    {  
      "id": "cf1ad6e3-3cbb-4b8a-a7f2-3EXAMPLEa118",  
      "resourceName": "LoadBalancer-1",  
      "resourceType": "LoadBalancer",  
      "createdAt": 1571072255.416,  
      "location": {  
        "availabilityZone": "all",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": true,  
      "operationDetails": "Certificate2",  
      "operationType": "AttachLoadBalancerTlsCertificate",  
      "status": "Succeeded",  
      "statusChangedAt": 1571072255.416  
    },  
    {  
      "id": "dae1bcfb-d531-4c06-b4ea-bEXAMPLEc04e",  
      "resourceName": "Certificate2",  
      "resourceType": "LoadBalancerTlsCertificate",  
      "createdAt": 1571072255.416,  
      "location": {  
        "availabilityZone": "all",  
        "regionName": "us-west-2"  
      },  
    },  
  ],  
}
```

```

        "isTerminal": true,
        "operationDetails": "LoadBalancer-1",
        "operationType": "AttachLoadBalancerTlsCertificate",
        "status": "Succeeded",
        "statusChangedAt": 1571072255.416
    }
]
}

```

- For API details, see [AttachLoadBalancerTlsCertificate](#) in *AWS CLI Command Reference*.

attach-static-ip

The following code example shows how to use `attach-static-ip`.

AWS CLI

To attach a static IP to an instance

The following `attach-static-ip` example attaches static IP `StaticIp-1` to instance `MEAN-1`.

```

aws lightsail attach-static-ip \
  --static-ip-name StaticIp-1 \
  --instance-name MEAN-1

```

Output:

```

{
  "operations": [
    {
      "id": "45e6fa13-4808-4b8d-9292-bEXAMPLE20b2",
      "resourceName": "StaticIp-1",
      "resourceType": "StaticIp",
      "createdAt": 1571072569.375,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      },
      "isTerminal": true,
      "operationDetails": "MEAN-1",
      "operationType": "AttachStaticIp",
    }
  ]
}

```



```

        "status": "Succeeded",
        "statusChangedAt": 1571072569.375
    },
    {
        "id": "9ee09a17-863c-4e51-8a6d-3EXAMPLE5475",
        "resourceName": "MEAN-1",
        "resourceType": "Instance",
        "createdAt": 1571072569.376,
        "location": {
            "availabilityZone": "us-west-2a",
            "regionName": "us-west-2"
        },
        "isTerminal": true,
        "operationDetails": "StaticIp-1",
        "operationType": "AttachStaticIp",
        "status": "Succeeded",
        "statusChangedAt": 1571072569.376
    }
]
}

```

- For API details, see [AttachStaticIp](#) in *AWS CLI Command Reference*.

close-instance-public-ports

The following code example shows how to use `close-instance-public-ports`.

AWS CLI

To close firewall ports for an instance

The following `close-instance-public-ports` example closes TCP port 22 on instance MEAN-2.

```

aws lightsail close-instance-public-ports \
  --instance-name MEAN-2 \
  --port-info fromPort=22,protocol=TCP,toPort=22

```

Output:

```

{
  "operation": {

```

```
"id": "4f328636-1c96-4649-ae6d-1EXAMPLEf446",
"resourceName": "MEAN-2",
"resourceType": "Instance",
"createdAt": 1571072845.737,
"location": {
  "availabilityZone": "us-west-2a",
  "regionName": "us-west-2"
},
"isTerminal": true,
"operationDetails": "22/tcp",
"operationType": "CloseInstancePublicPorts",
"status": "Succeeded",
"statusChangedAt": 1571072845.737
}
}
```

- For API details, see [CloseInstancePublicPorts](#) in *AWS CLI Command Reference*.

copy-snapshot

The following code example shows how to use copy-snapshot.

AWS CLI

Example 1: To copy a snapshot within the same AWS Region

The following copy-snapshot example copies instance snapshot MEAN-1-1571075291 as instance snapshot MEAN-1-Copy within the same AWS Region us-west-2.

```
aws lightsail copy-snapshot \  
  --source-snapshot-name MEAN-1-1571075291 \  
  --target-snapshot-name MEAN-1-Copy \  
  --source-region us-west-2
```

Output:

```
{
  "operations": [
    {
      "id": "ced16fc1-f401-4556-8d82-1EXAMPLEb982",
      "resourceName": "MEAN-1-Copy",
      "resourceType": "InstanceSnapshot",
```

```

    "createdAt": 1571075581.498,
    "location": {
      "availabilityZone": "all",
      "regionName": "us-west-2"
    },
    "isTerminal": false,
    "operationDetails": "us-west-2:MEAN-1-1571075291",
    "operationType": "CopySnapshot",
    "status": "Started",
    "statusChangedAt": 1571075581.498
  }
]
}

```

For more information, see [Copying snapshots from one AWS Region to another in Amazon Lightsail](#) in the *Lightsail Dev Guide*.

Example 2: To copy a snapshot from one AWS Region to another

The following copy-snapshot example copies instance snapshot MEAN-1-1571075291 as instance snapshot MEAN-1-1571075291-Copy from AWS Region us-west-2 to us-east-1.

```

aws lightsail copy-snapshot \
  --source-snapshot-name MEAN-1-1571075291 \
  --target-snapshot-name MEAN-1-1571075291-Copy \
  --source-region us-west-2 \
  --region us-east-1

```

Output:

```

{
  "operations": [
    {
      "id": "91116b79-119c-4451-b44a-dEXAMPLEd97b",
      "resourceName": "MEAN-1-1571075291-Copy",
      "resourceType": "InstanceSnapshot",
      "createdAt": 1571075695.069,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-east-1"
      },
      "isTerminal": false,
      "operationDetails": "us-west-2:MEAN-1-1571075291",
    }
  ]
}

```

```

        "operationType": "CopySnapshot",
        "status": "Started",
        "statusChangedAt": 1571075695.069
    }
]
}

```

For more information, see [Copying snapshots from one AWS Region to another in Amazon Lightsail](#) in the *Lightsail Dev Guide*.

Example 3: To copy an automatic snapshot within the same AWS Region

The following copy-snapshot example copies automatic snapshot 2019-10-14 of instance WordPress-1 as a manual snapshot WordPress-1-10142019 in the AWS Region us-west-2.

```

aws lightsail copy-snapshot \
  --source-resource-name WordPress-1 \
  --restore-date 2019-10-14 \
  --target-snapshot-name WordPress-1-10142019 \
  --source-region us-west-2

```

Output:

```

{
  "operations": [
    {
      "id": "be3e6754-cd1d-48e6-ad9f-2EXAMPLE1805",
      "resourceName": "WordPress-1-10142019",
      "resourceType": "InstanceSnapshot",
      "createdAt": 1571082412.311,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      },
      "isTerminal": false,
      "operationDetails": "us-west-2:WordPress-1",
      "operationType": "CopySnapshot",
      "status": "Started",
      "statusChangedAt": 1571082412.311
    }
  ]
}

```

For more information, see [Keeping automatic snapshots of instances or disks in Amazon Lightsail](#) in the *Lightsail Dev Guide*.

Example 4: To copy an automatic snapshot from one AWS Region to another

The following copy-snapshot example copies automatic snapshot 2019-10-14 of instance WordPress-1 as a manual snapshot WordPress-1-10142019 from the AWS Region us-west-2 to us-east-1.

```
aws lightsail copy-snapshot \  
  --source-resource-name WordPress-1 \  
  --restore-date 2019-10-14 \  
  --target-snapshot-name WordPress-1-10142019 \  
  --source-region us-west-2 \  
  --region us-east-1
```

Output:

```
{  
  "operations": [  
    {  
      "id": "dffa128b-0b07-476e-b390-bEXAMPLE3775",  
      "resourceName": "WordPress-1-10142019",  
      "resourceType": "InstanceSnapshot",  
      "createdAt": 1571082493.422,  
      "location": {  
        "availabilityZone": "all",  
        "regionName": "us-east-1"  
      },  
      "isTerminal": false,  
      "operationDetails": "us-west-2:WordPress-1",  
      "operationType": "CopySnapshot",  
      "status": "Started",  
      "statusChangedAt": 1571082493.422  
    }  
  ]  
}
```

For more information, see [Keeping automatic snapshots of instances or disks in Amazon Lightsail](#) in the *Lightsail Dev Guide*.

- For API details, see [CopySnapshot](#) in *AWS CLI Command Reference*.

create-disk-from-snapshot

The following code example shows how to use `create-disk-from-snapshot`.

AWS CLI

To create a create a disk from a disk snapshot

The following `create-disk-from-snapshot` example creates a block storage disk named `Disk-2` from the specified block storage disk snapshot. The disk is created in the specified AWS Region and Availability Zone, with 32 GB of storage space.

```
aws lightsail create-disk-from-snapshot \  
  --disk-name Disk-2 \  
  --disk-snapshot-name Disk-1-1566839161 \  
  --availability-zone us-west-2a \  
  --size-in-gb 32
```

Output:

```
{  
  "operations": [  
    {  
      "id": "d42b605d-5ef1-4b4a-8791-7a3e8b66b5e7",  
      "resourceName": "Disk-2",  
      "resourceType": "Disk",  
      "createdAt": 1569624941.471,  
      "location": {  
        "availabilityZone": "us-west-2a",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": false,  
      "operationType": "CreateDiskFromSnapshot",  
      "status": "Started",  
      "statusChangedAt": 1569624941.791  
    }  
  ]  
}
```

For more information, see [Creating a block storage disk from a snapshot in Amazon Lightsail](#) in the *Lightsail Developer Guide*.

- For API details, see [CreateDiskFromSnapshot](#) in *AWS CLI Command Reference*.

create-disk-snapshot

The following code example shows how to use create-disk-snapshot.

AWS CLI

Example 1: To create a snapshot of a disk

The following create-disk-snapshot example creates a snapshot named DiskSnapshot-1 of the specified block storage disk.

```
aws lightsail create-disk-snapshot \  
  --disk-name Disk-1 \  
  --disk-snapshot-name DiskSnapshot-1
```

Output:

```
{  
  "operations": [  
    {  
      "id": "fa74c6d2-03a3-4f42-a7c7-792f124d534b",  
      "resourceName": "DiskSnapshot-1",  
      "resourceType": "DiskSnapshot",  
      "createdAt": 1569625129.739,  
      "location": {  
        "availabilityZone": "all",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": false,  
      "operationDetails": "Disk-1",  
      "operationType": "CreateDiskSnapshot",  
      "status": "Started",  
      "statusChangedAt": 1569625129.739  
    },  
    {  
      "id": "920a25df-185c-4528-87cd-7b85f5488c06",  
      "resourceName": "Disk-1",  
      "resourceType": "Disk",  
      "createdAt": 1569625129.739,  
      "location": {  
        "availabilityZone": "us-west-2a",  
        "regionName": "us-west-2"  
      },  
    },  
  ],  
}
```

```

        "isTerminal": false,
        "operationDetails": "DiskSnapshot-1",
        "operationType": "CreateDiskSnapshot",
        "status": "Started",
        "statusChangedAt": 1569625129.739
    }
]
}

```

Example 2: To create a snapshot of an instance's system disk

The following `create-disk-snapshot` example creates a snapshot of the specified instance's system disk.

```

aws lightsail create-disk-snapshot \
  --instance-name WordPress-1 \
  --disk-snapshot-name SystemDiskSnapshot-1

```

Output:

```

{
  "operations": [
    {
      "id": "f508cf1c-6597-42a6-a4c3-4aebd75af0d9",
      "resourceName": "SystemDiskSnapshot-1",
      "resourceType": "DiskSnapshot",
      "createdAt": 1569625294.685,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      },
      "isTerminal": false,
      "operationDetails": "WordPress-1",
      "operationType": "CreateDiskSnapshot",
      "status": "Started",
      "statusChangedAt": 1569625294.685
    },
    {
      "id": "0bb9f712-da3b-4d99-b508-3bf871d989e5",
      "resourceName": "WordPress-1",
      "resourceType": "Instance",
      "createdAt": 1569625294.685,
      "location": {

```



```

        "availabilityZone": "us-west-2a",
        "regionName": "us-west-2"
    },
    "isTerminal": false,
    "operationDetails": "SystemDiskSnapshot-1",
    "operationType": "CreateDiskSnapshot",
    "status": "Started",
    "statusChangedAt": 1569625294.685
}
]
}

```

For more information, see [Snapshots in Amazon Lightsail](#) and [Creating a snapshot of an instance root volume in Amazon Lightsail](#) in the *Lightsail Developer Guide*.

- For API details, see [CreateDiskSnapshot](#) in *AWS CLI Command Reference*.

create-disk

The following code example shows how to use `create-disk`.

AWS CLI

To create a block storage disk

The following `create-disk` example creates a block storage disk `Disk-1` in the specified AWS Region and Availability Zone, with 32 GB of storage space.

```

aws lightsail create-disk \
  --disk-name Disk-1 \
  --availability-zone us-west-2a \
  --size-in-gb 32

```

Output:

```

{
  "operations": [
    {
      "id": "1c85e2ec-86ba-4697-b936-77f4d3dc013a",
      "resourceName": "Disk-1",
      "resourceType": "Disk",
      "createdAt": 1569449220.36,
      "location": {

```

```

        "availabilityZone": "us-west-2a",
        "regionName": "us-west-2"
    },
    "isTerminal": false,
    "operationType": "CreateDisk",
    "status": "Started",
    "statusChangedAt": 1569449220.588
}
]
}

```

- For API details, see [CreateDisk](#) in *AWS CLI Command Reference*.

create-domain-entry

The following code example shows how to use `create-domain-entry`.

AWS CLI

To create a domain entry (DNS record)

The following `create-domain-entry` example creates a DNS record (A) for the apex of the specified domain that points to an instance's IP address.

Note: Lightsail's domain-related API operations are available in only the `us-east-1` Region. If your CLI profile is configured to use a different Region, you must include the `--region us-east-1` parameter or the command fails.

```

aws lightsail create-domain-entry \
  --region us-east-1 \
  --domain-name example.com \
  --domain-entry name=example.com,type=A,target=192.0.2.0

```

Output:

```

{
  "operation": {
    "id": "5be4494d-56f4-41fc-8730-693dcd0ef9e2",
    "resourceName": "example.com",
    "resourceType": "Domain",
    "createdAt": 1569865296.519,
    "location": {

```

```
        "availabilityZone": "all",
        "regionName": "global"
    },
    "isTerminal": true,
    "operationType": "CreateDomainEntry",
    "status": "Succeeded",
    "statusChangedAt": 1569865296.519
}
}
```

For more information, see [DNS in Amazon Lightsail](#) and [Creating a DNS zone to manage your domain's DNS records in Amazon Lightsail](#) in the *Lightsail Developer Guide*.

- For API details, see [CreateDomainEntry](#) in *AWS CLI Command Reference*.

create-domain

The following code example shows how to use create-domain.

AWS CLI

To create a domain (DNS zone)

The following create-domain example creates a DNS zone for the specified domain.

Note: Lightsail's domain-related API operations are available in only the us-east-1 Region. If your CLI profile is configured to use a different Region, you must include the `--region us-east-1` parameter or the command fails.

```
aws lightsail create-domain \
  --region us-east-1 \
  --domain-name example.com
```

Output:

```
{
  "operation": {
    "id": "64e522c8-9ae1-4c05-9b65-3f237324dc34",
    "resourceName": "example.com",
    "resourceType": "Domain",
    "createdAt": 1569864291.92,
    "location": {
      "availabilityZone": "all",
```

```
        "regionName": "global"
    },
    "isTerminal": true,
    "operationType": "CreateDomain",
    "status": "Succeeded",
    "statusChangedAt": 1569864292.109
}
}
```

For more information, see [DNS in Amazon Lightsail](#) and [Creating a DNS zone to manage your domain's DNS records in Amazon Lightsail](#) in the *Lightsail Developer Guide*.

- For API details, see [CreateDomain](#) in *AWS CLI Command Reference*.

create-instance-snapshot

The following code example shows how to use `create-instance-snapshot`.

AWS CLI

To create a snapshot of an instance

The following `create-instance-snapshot` example creates a snapshot from the specified instance.

```
aws lightsail create-instance-snapshot \
  --instance-name WordPress-1 \
  --instance-snapshot-name WordPress-Snapshot-1
```

Output:

```
{
  "operations": [
    {
      "id": "4c3db559-9dd0-41e7-89c0-2cb88c19786f",
      "resourceName": "WordPress-Snapshot-1",
      "resourceType": "InstanceSnapshot",
      "createdAt": 1569866438.48,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      },
      "isTerminal": false,
```

```
    "operationDetails": "WordPress-1",
    "operationType": "CreateInstanceSnapshot",
    "status": "Started",
    "statusChangedAt": 1569866438.48
  },
  {
    "id": "c04fdc45-2981-488c-88b5-d6d2fd759a6a",
    "resourceName": "WordPress-1",
    "resourceType": "Instance",
    "createdAt": 1569866438.48,
    "location": {
      "availabilityZone": "us-west-2a",
      "regionName": "us-west-2"
    },
    "isTerminal": false,
    "operationDetails": "WordPress-Snapshot-1",
    "operationType": "CreateInstanceSnapshot",
    "status": "Started",
    "statusChangedAt": 1569866438.48
  }
]
}
```

- For API details, see [CreateInstanceSnapshot](#) in *AWS CLI Command Reference*.

create-instances-from-snapshot

The following code example shows how to use `create-instances-from-snapshot`.

AWS CLI

To create an instance from a snapshot

The following `create-instances-from-snapshot` example creates an instance from the specified instance snapshot, in the specified AWS Region and Availability Zone, using the \$10 USD bundle.

Note: The bundle that you specify must be equal to or greater in specifications than the bundle of the original source instance used to create the snapshot.

```
aws lightsail create-instances-from-snapshot \
  --instance-snapshot-name WordPress-1-1569866208 \
  --instance-names WordPress-2 \
```

```
--availability-zone us-west-2a \  
--bundle-id medium_2_0
```

Output:

```
{  
  "operations": [  
    {  
      "id": "003f8271-b711-464d-b9b8-7f3806cb496e",  
      "resourceName": "WordPress-2",  
      "resourceType": "Instance",  
      "createdAt": 1569865914.908,  
      "location": {  
        "availabilityZone": "us-west-2a",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": false,  
      "operationType": "CreateInstancesFromSnapshot",  
      "status": "Started",  
      "statusChangedAt": 1569865914.908  
    }  
  ]  
}
```

- For API details, see [CreateInstancesFromSnapshot](#) in *AWS CLI Command Reference*.

create-instances

The following code example shows how to use create-instances.

AWS CLI

Example 1: To create a single instance

The following create-instances example creates an instance in the specified AWS Region and Availability Zone, using the WordPress blueprint, and the \$3.50 USD bundle.

```
aws lightsail create-instances \  
  --instance-names Instance-1 \  
  --availability-zone us-west-2a \  
  --blueprint-id wordpress_5_1_1_2 \  
  --bundle-id nano_2_0
```

Output:

```
{
  "operations": [
    {
      "id": "9a77158f-7be3-4d6d-8054-cf5ae2b720cc",
      "resourceName": "Instance-1",
      "resourceType": "Instance",
      "createdAt": 1569447986.061,
      "location": {
        "availabilityZone": "us-west-2a",
        "regionName": "us-west-2"
      },
      "isTerminal": false,
      "operationType": "CreateInstance",
      "status": "Started",
      "statusChangedAt": 1569447986.061
    }
  ]
}
```

Example 2: To create multiple instances at one time

The following `create-instances` example creates three instances in the specified AWS Region and Availability Zone, using the WordPress blueprint, and the \$3.50 USD bundle.

```
aws lightsail create-instances \
  --instance-names {"Instance1","Instance2","Instance3"} \
  --availability-zone us-west-2a \
  --blueprint-id wordpress_5_1_1_2 \
  --bundle-id nano_2_0
```

Output:

```
{
  "operations": [
    {
      "id": "5492f015-9d2e-48c6-8eea-b516840e6903",
      "resourceName": "Instance1",
      "resourceType": "Instance",
      "createdAt": 1569448780.054,
      "location": {
        "availabilityZone": "us-west-2a",
```

```
        "regionName": "us-west-2"
      },
      "isTerminal": false,
      "operationType": "CreateInstance",
      "status": "Started",
      "statusChangedAt": 1569448780.054
    },
    {
      "id": "c58b5f46-2676-44c8-b95c-3ad375898515",
      "resourceName": "Instance2",
      "resourceType": "Instance",
      "createdAt": 1569448780.054,
      "location": {
        "availabilityZone": "us-west-2a",
        "regionName": "us-west-2"
      },
      "isTerminal": false,
      "operationType": "CreateInstance",
      "status": "Started",
      "statusChangedAt": 1569448780.054
    },
    {
      "id": "a5ad8006-9bee-4499-9eb7-75e42e6f5882",
      "resourceName": "Instance3",
      "resourceType": "Instance",
      "createdAt": 1569448780.054,
      "location": {
        "availabilityZone": "us-west-2a",
        "regionName": "us-west-2"
      },
      "isTerminal": false,
      "operationType": "CreateInstance",
      "status": "Started",
      "statusChangedAt": 1569448780.054
    }
  ]
}
```

- For API details, see [CreateInstances](#) in *AWS CLI Command Reference*.

create-key-pair

The following code example shows how to use `create-key-pair`.

AWS CLI

To create a key pair

The following `create-key-pair` example creates a key pair that you can use to authenticate and connect to an instance.

```
aws lightsail create-key-pair \
  --key-pair-name MyPersonalKeyPair
```

The output provides the private key base64 value that you can use to authenticate to instances that use the created key pair. **Note:** Copy and paste the private key base64 value to a safe location because you cannot retrieve it later.

```
{
  "keyPair": {
    "name": "MyPersonalKeyPair",
    "arn": "arn:aws:lightsail:us-west-2:111122223333:KeyPair/55025c71-198f-403b-
b42f-a69433e724fb",
    "supportCode": "621291663362/MyPersonalKeyPair",
    "createdAt": 1569866556.567,
    "location": {
      "availabilityZone": "all",
      "regionName": "us-west-2"
    },
    "resourceType": "KeyPair"
  },
  "publicKeyBase64": "ssh-rsa ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAQCV0xUEwX96amPERH7K1bVT1tTF190mNk6o7m5YVhk9x10dMbDRbFvhtXvw4jz
+BUHgedGUXno6uF7agqxZN01kPLJBIVTW26SSYBJ0tE
+y804UyVsjrUqCaMXDhmfXpWuLMPwuXhwcKh7e8hwoTfkiX0E6Q1
+KqF/MiA3w6DCjEqvvdI07SiEZJFsuGNfYDDN3w60Re15MUhmn30Jdn4y/
A7NWb3IxL4pPfvE4rgFRKU8n1jp9kwRnLVMVB0WuGXk6n+H6M2f1 ",
  "privateKeyBase64": "-----BEGIN RSA PRIVATE KEY-----
EXAMPLETCCAfICCQD6m7oRw0uX0jANBgqhkIG9w0BAQUFADCBiDELMakGA1UEBhMC
\nVVMxCzAJBgNVBAgTAldBMRawDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6\nnb24xFDASBgNVBAwTC01BTSBD
\nBgqhkIG9w0BCQEWEG5vb251QGftYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
\nMTIwNDI0MjA0NTIxWjCBiDELMakGA1UEBhMCMVVMxCzAJBgNVBAgTAldBMRawDgYD
\nVQQHEwdTZWF0dGx1MQ8wDQEXAMPEwZBbWF6b24xFDASBgNVBAwTC01BTSBDb25z
\nb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWMxHzAdBgqhkIG9w0BCQEWEG5vb251QGft
\nYXpvbi5jb20wZG8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMEXAMPLE4GmWIWJ
\n21uUSfwfEvySwTc2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
\nrDHUdUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
```

```

\nIbb30hjZncvQAaREXAMPLEMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4\nnUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
\nFFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780EXAMPLELvJx79LjSTb
\nNYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=\n-----END RSA PRIVATE KEY-----",
  "operation": {
    "id": "67f984db-9994-45fe-ad38-59bafcaf82ef",
    "resourceName": "MyPersonalKeyPair",
    "resourceType": "KeyPair",
    "createdAt": 1569866556.567,
    "location": {
      "availabilityZone": "all",
      "regionName": "us-west-2"
    },
    "isTerminal": true,
    "operationType": "CreateKeyPair",
    "status": "Succeeded",
    "statusChangedAt": 1569866556.704
  }
}

```

- For API details, see [CreateKeyPair](#) in *AWS CLI Command Reference*.

create-load-balancer-tls-certificate

The following code example shows how to use `create-load-balancer-tls-certificate`.

AWS CLI

To create a TLS certificate for a load balancer

The following `create-load-balancer-tls-certificate` example creates a TLS certificate that is attached to the specified load balancer. The certificate created applies to the specified domains. **Note:** Only two certificates can be created for a load balancer.

```

aws lightsail create-load-balancer-tls-certificate \
  --certificate-alternative-names abc.example.com \
  --certificate-domain-name example.com \
  --certificate-name MySecondCertificate \
  --load-balancer-name MyFirstLoadBalancer

```

Output:

```
{
  "operations": [
    {
      "id": "be663aed-cb46-41e2-9b23-e2f747245bd4",
      "resourceName": "MySecondCertificate",
      "resourceType": "LoadBalancerTlsCertificate",
      "createdAt": 1569867364.971,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      },
      "isTerminal": true,
      "operationDetails": "MyFirstLoadBalancer",
      "operationType": "CreateLoadBalancerTlsCertificate",
      "status": "Succeeded",
      "statusChangedAt": 1569867365.219
    },
    {
      "id": "f3dfa930-969e-41cc-ac7d-337178716f6d",
      "resourceName": "MyFirstLoadBalancer",
      "resourceType": "LoadBalancer",
      "createdAt": 1569867364.971,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      },
      "isTerminal": true,
      "operationDetails": "MySecondCertificate",
      "operationType": "CreateLoadBalancerTlsCertificate",
      "status": "Succeeded",
      "statusChangedAt": 1569867365.219
    }
  ]
}
```

- For API details, see [CreateLoadBalancerTlsCertificate](#) in *AWS CLI Command Reference*.

create-load-balancer

The following code example shows how to use create-load-balancer.

AWS CLI

To create a load balancer

The following `create-load-balancer` example creates a load balancer with a TLS certificate. The TLS certificate applies to the specified domains, and routes traffic to instances on port 80.

```
aws lightsail create-load-balancer \  
  --certificate-alternative-names www.example.com test.example.com \  
  --certificate-domain-name example.com \  
  --certificate-name Certificate-1 \  
  --instance-port 80 \  
  --load-balancer-name LoadBalancer-1
```

Output:

```
{  
  "operations": [  
    {  
      "id": "cc7b920a-83d8-4762-a74e-9174fe1540be",  
      "resourceName": "LoadBalancer-1",  
      "resourceType": "LoadBalancer",  
      "createdAt": 1569867169.406,  
      "location": {  
        "availabilityZone": "all",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": false,  
      "operationType": "CreateLoadBalancer",  
      "status": "Started",  
      "statusChangedAt": 1569867169.406  
    },  
    {  
      "id": "658ed43b-f729-42f3-a8e4-3f8024d3c98d",  
      "resourceName": "LoadBalancer-1",  
      "resourceType": "LoadBalancerTlsCertificate",  
      "createdAt": 1569867170.193,  
      "location": {  
        "availabilityZone": "all",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": true,  
      "operationDetails": "LoadBalancer-1",
```

```

    "operationType": "CreateLoadBalancerTlsCertificate",
    "status": "Succeeded",
    "statusChangedAt": 1569867170.54
  },
  {
    "id": "4757a342-5181-4870-b1e0-227eebc35ab5",
    "resourceName": "LoadBalancer-1",
    "resourceType": "LoadBalancer",
    "createdAt": 1569867170.193,
    "location": {
      "availabilityZone": "all",
      "regionName": "us-west-2"
    },
    "isTerminal": true,
    "operationDetails": "Certificate-1",
    "operationType": "CreateLoadBalancerTlsCertificate",
    "status": "Succeeded",
    "statusChangedAt": 1569867170.54
  }
]
}

```

For more information, see [Lightsail load balancers](#) in the *Lightsail Developer Guide*.

- For API details, see [CreateLoadBalancer](#) in *AWS CLI Command Reference*.

create-relational-database-from-snapshot

The following code example shows how to use `create-relational-database-from-snapshot`.

AWS CLI

To create a managed database from a snapshot

The following `create-relational-database-from-snapshot` example creates a managed database from the specified snapshot in the specified AWS Region and Availability Zone, using the \$15 USD standard database bundle. **Note:** The bundle that you specify must be equal to or greater in specifications than the bundle of the original source database used to create the snapshot.

```

aws lightsail create-relational-database-from-snapshot \
  --relational-database-snapshot-name Database-0regon-1-1566839359 \

```

```
--relational-database-name Database-1 \  
--availability-zone us-west-2a \  
--relational-database-bundle-id micro_1_0 \  
--no-publicly-accessible
```

Output:

```
{  
  "operations": [  
    {  
      "id": "ad6d9193-9d5c-4ea1-97ae-8fe6de600b4c",  
      "resourceName": "Database-1",  
      "resourceType": "RelationalDatabase",  
      "createdAt": 1569867916.938,  
      "location": {  
        "availabilityZone": "us-west-2a",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": false,  
      "operationType": "CreateRelationalDatabaseFromSnapshot",  
      "status": "Started",  
      "statusChangedAt": 1569867918.643  
    }  
  ]  
}
```

- For API details, see [CreateRelationalDatabaseFromSnapshot](#) in *AWS CLI Command Reference*.

create-relational-database-snapshot

The following code example shows how to use `create-relational-database-snapshot`.

AWS CLI

To create a snapshot of a managed database

The following `create-relational-database-snapshot` example creates a snapshot of the specified managed database.

```
aws lightsail create-relational-database-snapshot \  
--relational-database-name Database1 \  
--relational-database-snapshot-name RelationalDatabaseSnapshot1
```

Output:

```
{
  "operations": [
    {
      "id": "853667fb-ea91-4c02-8d20-8fc5fd43b9eb",
      "resourceName": "RelationalDatabaseSnapshot1",
      "resourceType": "RelationalDatabaseSnapshot",
      "createdAt": 1569868074.645,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      },
      "isTerminal": false,
      "operationDetails": "Database1",
      "operationType": "CreateRelationalDatabaseSnapshot",
      "status": "Started",
      "statusChangedAt": 1569868074.645
    },
    {
      "id": "fbafa521-3cac-4be8-9773-1c143780b239",
      "resourceName": "Database1",
      "resourceType": "RelationalDatabase",
      "createdAt": 1569868074.645,
      "location": {
        "availabilityZone": "us-west-2a",
        "regionName": "us-west-2"
      },
      "isTerminal": false,
      "operationDetails": "RelationalDatabaseSnapshot1",
      "operationType": "CreateRelationalDatabaseSnapshot",
      "status": "Started",
      "statusChangedAt": 1569868074.645
    }
  ]
}
```

- For API details, see [CreateRelationalDatabaseSnapshot](#) in *AWS CLI Command Reference*.

create-relational-database

The following code example shows how to use `create-relational-database`.

AWS CLI

To create a managed database

The following `create-relational-database` example creates a managed database in the specified AWS Region and Availability Zone, using the MySQL 5.6 database engine (`mysql_5_6`), and the \$15 USD standard database bundle (`micro_1_0`). The managed database is pre-populated a master user name, and is not publicly accessible.

```
aws lightsail create-relational-database \  
  --relational-database-name Database-1 \  
  --availability-zone us-west-2a \  
  --relational-database-blueprint-id mysql_5_6 \  
  --relational-database-bundle-id micro_1_0 \  
  --master-database-name dbmaster \  
  --master-username user \  
  --no-publicly-accessible
```

Output:

```
{  
  "operations": [  
    {  
      "id": "b52bedee-73ed-4798-8d2a-9c12df89adcd",  
      "resourceName": "Database-1",  
      "resourceType": "RelationalDatabase",  
      "createdAt": 1569450017.244,  
      "location": {  
        "availabilityZone": "us-west-2a",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": false,  
      "operationType": "CreateRelationalDatabase",  
      "status": "Started",  
      "statusChangedAt": 1569450018.637  
    }  
  ]  
}
```

- For API details, see [CreateRelationalDatabase](#) in *AWS CLI Command Reference*.

delete-auto-snapshot

The following code example shows how to use delete-auto-snapshot.

AWS CLI

To delete an automatic snapshot

The following delete-auto-snapshot example deletes the automatic snapshot 2019-10-10 of instance WordPress-1.

```
aws lightsail delete-auto-snapshot \  
  --resource-name WordPress-1 \  
  --date 2019-10-10
```

Output:

```
{  
  "operations": [  
    {  
      "id": "31c36e09-3d52-46d5-b6d8-7EXAMPLE534a",  
      "resourceName": "WordPress-1",  
      "resourceType": "Instance",  
      "createdAt": 1571088141.501,  
      "location": {  
        "availabilityZone": "us-west-2",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": true,  
      "operationDetails": "DeleteAutoSnapshot-2019-10-10",  
      "operationType": "DeleteAutoSnapshot",  
      "status": "Succeeded"  
    }  
  ]  
}
```

For more information, see [Deleting automatic snapshots of instances or disks in Amazon Lightsail](#) in the *Lightsail Dev Guide*.

- For API details, see [DeleteAutoSnapshot](#) in *AWS CLI Command Reference*.

delete-disk-snapshot

The following code example shows how to use delete-disk-snapshot.

AWS CLI

To delete a snapshot of a block storage disk

The following delete-disk-snapshot example deletes the specified snapshot of a block storage disk

```
aws lightsail delete-disk-snapshot \  
  --disk-snapshot-name DiskSnapshot-1
```

Output:

```
{  
  "operations": [  
    {  
      "id": "d1e5766d-b81e-4595-ad5d-02afbcccfd5d",  
      "resourceName": "DiskSnapshot-1",  
      "resourceType": "DiskSnapshot",  
      "createdAt": 1569873552.79,  
      "location": {  
        "availabilityZone": "all",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": true,  
      "operationType": "DeleteDiskSnapshot",  
      "status": "Succeeded",  
      "statusChangedAt": 1569873552.79  
    }  
  ]  
}
```

- For API details, see [DeleteDiskSnapshot](#) in *AWS CLI Command Reference*.

delete-disk

The following code example shows how to use delete-disk.

AWS CLI

To delete a block storage disk

The following `delete-disk` example deletes the specified block storage disk.

```
aws lightsail delete-disk \  
  --disk-name Disk-1
```

Output:

```
{  
  "operations": [  
    {  
      "id": "6378c70f-4d75-4f7a-ab66-730fca0bb2fc",  
      "resourceName": "Disk-1",  
      "resourceType": "Disk",  
      "createdAt": 1569872887.864,  
      "location": {  
        "availabilityZone": "us-west-2a",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": true,  
      "operationType": "DeleteDisk",  
      "status": "Succeeded",  
      "statusChangedAt": 1569872887.864  
    }  
  ]  
}
```

- For API details, see [DeleteDisk](#) in *AWS CLI Command Reference*.

delete-domain-entry

The following code example shows how to use `delete-domain-entry`.

AWS CLI

To delete a domain entry (DNS record)

The following `delete-domain-entry` example deletes the specified domain entry from an existing domain.

Note: Lightsail's domain-related API operations are available in only the `us-east-1` Region. If your CLI profile is configured to use a different Region, you must include the `--region us-east-1` parameter or the command fails.

```
aws lightsail delete-domain-entry \  
  --region us-east-1 \  
  --domain-name example.com \  
  --domain-entry name=123.example.com,target=192.0.2.0,type=A
```

Output:

```
{  
  "operation": {  
    "id": "06eacd01-d785-420e-8daa-823150c7dca1",  
    "resourceName": "example.com ",  
    "resourceType": "Domain",  
    "createdAt": 1569874157.005,  
    "location": {  
      "availabilityZone": "all",  
      "regionName": "global"  
    },  
    "isTerminal": true,  
    "operationType": "DeleteDomainEntry",  
    "status": "Succeeded",  
    "statusChangedAt": 1569874157.005  
  }  
}
```

- For API details, see [DeleteDomainEntry](#) in *AWS CLI Command Reference*.

delete-domain

The following code example shows how to use `delete-domain`.

AWS CLI

To delete a domain (DNS zone)

The following `delete-domain` example deletes the specified domain and all of the entries in the domain (DNS records).

Note: Lightsail's domain-related API operations are available in only the `us-east-1` Region. If your CLI profile is configured to use a different Region, you must include the `--region us-east-1` parameter or the command fails.

```
aws lightsail delete-domain \  
  --region us-east-1 \  
  --domain-name example.com
```

Output:

```
{  
  "operation": {  
    "id": "fcef5265-5af1-4a46-a3d7-90b5e18b9b32",  
    "resourceName": "example.com",  
    "resourceType": "Domain",  
    "createdAt": 1569873788.13,  
    "location": {  
      "availabilityZone": "all",  
      "regionName": "global"  
    },  
    "isTerminal": true,  
    "operationType": "DeleteDomain",  
    "status": "Succeeded",  
    "statusChangedAt": 1569873788.13  
  }  
}
```

- For API details, see [DeleteDomain](#) in *AWS CLI Command Reference*.

delete-instance-snapshot

The following code example shows how to use `delete-instance-snapshot`.

AWS CLI

title

The following `delete-instance-snapshot` example deletes the specified snapshot of an instance.

```
aws lightsail delete-instance-snapshot \  
  --instance-id example-instance-id \  
  --snapshot-id example-snapshot-id
```

```
--instance-snapshot-name WordPress-1-Snapshot-1
```

Output:

```
{
  "operations": [
    {
      "id": "14dad182-976a-46c6-bfd4-9480482bf0ea",
      "resourceName": "WordPress-1-Snapshot-1",
      "resourceType": "InstanceSnapshot",
      "createdAt": 1569874524.562,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      },
      "isTerminal": true,
      "operationType": "DeleteInstanceSnapshot",
      "status": "Succeeded",
      "statusChangedAt": 1569874524.562
    }
  ]
}
```

- For API details, see [DeleteInstanceSnapshot](#) in *AWS CLI Command Reference*.

delete-instance

The following code example shows how to use `delete-instance`.

AWS CLI**To delete an instance**

The following `delete-instance` example deletes the specified instance.

```
aws lightsail delete-instance \
  --instance-name WordPress-1
```

Output:

```
{
  "operations": [
```

```
{
  "id": "d77345a3-8f80-4d2e-b47d-aaa622718df2",
  "resourceName": "Disk-1",
  "resourceType": "Disk",
  "createdAt": 1569874357.469,
  "location": {
    "availabilityZone": "us-west-2a",
    "regionName": "us-west-2"
  },
  "isTerminal": false,
  "operationDetails": "WordPress-1",
  "operationType": "DetachDisk",
  "status": "Started",
  "statusChangedAt": 1569874357.469
},
{
  "id": "708fa606-2bfd-4e48-a2c1-0b856585b5b1",
  "resourceName": "WordPress-1",
  "resourceType": "Instance",
  "createdAt": 1569874357.465,
  "location": {
    "availabilityZone": "us-west-2a",
    "regionName": "us-west-2"
  },
  "isTerminal": false,
  "operationDetails": "Disk-1",
  "operationType": "DetachDisk",
  "status": "Started",
  "statusChangedAt": 1569874357.465
},
{
  "id": "3187e823-8acb-405d-b098-fad5ceb17bec",
  "resourceName": "WordPress-1",
  "resourceType": "Instance",
  "createdAt": 1569874357.829,
  "location": {
    "availabilityZone": "us-west-2a",
    "regionName": "us-west-2"
  },
  "isTerminal": true,
  "operationType": "DeleteInstance",
  "status": "Succeeded",
  "statusChangedAt": 1569874357.829
}
```

```
]
}
```

- For API details, see [DeleteInstance](#) in *AWS CLI Command Reference*.

delete-key-pair

The following code example shows how to use `delete-key-pair`.

AWS CLI

To delete a key pair

The following `delete-key-pair` example deletes the specified key pair.

```
aws lightsail delete-key-pair \
  --key-pair-name MyPersonalKeyPair
```

Output:

```
{
  "operation": {
    "id": "81621463-df38-4810-b866-6e801a15abbf",
    "resourceName": "MyPersonalKeyPair",
    "resourceType": "KeyPair",
    "createdAt": 1569874626.466,
    "location": {
      "availabilityZone": "all",
      "regionName": "us-west-2"
    },
    "isTerminal": true,
    "operationType": "DeleteKeyPair",
    "status": "Succeeded",
    "statusChangedAt": 1569874626.685
  }
}
```

- For API details, see [DeleteKeyPair](#) in *AWS CLI Command Reference*.

delete-known-host-keys

The following code example shows how to use `delete-known-host-keys`.

AWS CLI

To delete known host keys from an instance

The following `delete-known-host-keys` example deletes the known host key from the specified instance.

```
aws lightsail delete-known-host-keys \  
  --instance-name Instance-1
```

Output:

```
{  
  "operations": [  
    {  
      "id": "c61afe9c-45a4-41e6-a97e-d212364da3f5",  
      "resourceName": "Instance-1",  
      "resourceType": "Instance",  
      "createdAt": 1569874760.201,  
      "location": {  
        "availabilityZone": "us-west-2a",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": true,  
      "operationType": "DeleteKnownHostKeys",  
      "status": "Succeeded",  
      "statusChangedAt": 1569874760.201  
    }  
  ]  
}
```

For more information, see [Troubleshooting connection issues with the Amazon Lightsail browser-based SSH or RDP client](#) in the *Lightsail Dev Guide*.

- For API details, see [DeleteKnownHostKeys](#) in *AWS CLI Command Reference*.

`delete-load-balancer-tls-certificate`

The following code example shows how to use `delete-load-balancer-tls-certificate`.

AWS CLI

To delete a TLS certificate for a load balancer

The following `delete-load-balancer-tls-certificate` example deletes the specific TLS certificate from the specified load balancer.

```
aws lightsail delete-load-balancer-tls-certificate \  
  --load-balancer-name MyFirstLoadBalancer \  
  --certificate-name MyFirstCertificate
```

Output:

```
{  
  "operations": [  
    {  
      "id": "50bec274-e45e-4caa-8a69-b763ef636583",  
      "resourceName": "MyFirstCertificate",  
      "resourceType": "LoadBalancerTlsCertificate",  
      "createdAt": 1569874989.48,  
      "location": {  
        "availabilityZone": "all",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": false,  
      "operationType": "DeleteLoadBalancerTlsCertificate",  
      "status": "Started",  
      "statusChangedAt": 1569874989.48  
    },  
    {  
      "id": "78c58cdc-a59a-4b27-8213-500638634a8f",  
      "resourceName": "MyFirstLoadBalancer",  
      "resourceType": "LoadBalancer",  
      "createdAt": 1569874989.48,  
      "location": {  
        "availabilityZone": "all",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": false,  
      "operationType": "DeleteLoadBalancerTlsCertificate",  
      "status": "Started",  
      "statusChangedAt": 1569874989.48  
    }  
  ]  
}
```

- For API details, see [DeleteLoadBalancerTlsCertificate](#) in *AWS CLI Command Reference*.

delete-load-balancer

The following code example shows how to use delete-load-balancer.

AWS CLI

To delete a load balancer

The following delete-load-balancer example deletes the specified load balancer and any associated TLS certificates.

```
aws lightsail delete-load-balancer \  
  --load-balancer-name MyFirstLoadBalancer
```

Output:

```
{  
  "operations": [  
    {  
      "id": "a8c968c7-72a3-4680-a714-af8f03eea535",  
      "resourceName": "MyFirstLoadBalancer",  
      "resourceType": "LoadBalancer",  
      "createdAt": 1569875092.125,  
      "location": {  
        "availabilityZone": "all",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": true,  
      "operationType": "DeleteLoadBalancer",  
      "status": "Succeeded",  
      "statusChangedAt": 1569875092.125  
    },  
    {  
      "id": "f91a29fc-8ce3-4e69-a227-ea70ca890bf5",  
      "resourceName": "MySecondCertificate",  
      "resourceType": "LoadBalancerTlsCertificate",  
      "createdAt": 1569875091.938,  
      "location": {  
        "availabilityZone": "all",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": false,  
      "operationType": "DeleteLoadBalancerTlsCertificate",
```

```

        "status": "Started",
        "statusChangedAt": 1569875091.938
    },
    {
        "id": "cf64c060-154b-4eb4-ba57-84e2e41563d6",
        "resourceName": "MyFirstLoadBalancer",
        "resourceType": "LoadBalancer",
        "createdAt": 1569875091.94,
        "location": {
            "availabilityZone": "all",
            "regionName": "us-west-2"
        },
        "isTerminal": false,
        "operationType": "DeleteLoadBalancerTlsCertificate",
        "status": "Started",
        "statusChangedAt": 1569875091.94
    }
]
}

```

For more information, see title in the *guide*.

- For API details, see [DeleteLoadBalancer](#) in *AWS CLI Command Reference*.

delete-relational-database-snapshot

The following code example shows how to use `delete-relational-database-snapshot`.

AWS CLI

To delete a snapshot of a managed database

The following `delete-relational-database-snapshot` example deletes the specified snapshot of a managed database.

```
aws lightsail delete-relational-database-snapshot \
  --relational-database-snapshot-name Database-Oregon-1-1566839359
```

Output:

```
{
  "operations": [
```

```

    {
      "id": "b99acae8-735b-4823-922f-30af580e3729",
      "resourceName": "Database-Oregon-1-1566839359",
      "resourceType": "RelationalDatabaseSnapshot",
      "createdAt": 1569875293.58,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      },
      "isTerminal": true,
      "operationType": "DeleteRelationalDatabaseSnapshot",
      "status": "Succeeded",
      "statusChangedAt": 1569875293.58
    }
  ]
}

```

- For API details, see [DeleteRelationalDatabaseSnapshot](#) in *AWS CLI Command Reference*.

delete-relational-database

The following code example shows how to use delete-relational-database.

AWS CLI

To delete a managed database

The following delete-relational-database example deletes the specified managed database.

```
aws lightsail delete-relational-database \
  --relational-database-name Database-1
```

Output:

```

{
  "operations": [
    {
      "id": "3b0c41c1-053d-46f0-92a3-14f76141dc86",
      "resourceName": "Database-1",
      "resourceType": "RelationalDatabase",

```

```

    "createdAt": 1569875210.999,
    "location": {
      "availabilityZone": "us-west-2a",
      "regionName": "us-west-2"
    },
    "isTerminal": false,
    "operationType": "DeleteRelationalDatabase",
    "status": "Started",
    "statusChangedAt": 1569875210.999
  },
  {
    "id": "01ddeae8-a87a-4a4b-a1f3-092c71bf9180",
    "resourceName": "Database-1",
    "resourceType": "RelationalDatabase",
    "createdAt": 1569875211.029,
    "location": {
      "availabilityZone": "us-west-2a",
      "regionName": "us-west-2"
    },
    "isTerminal": false,
    "operationDetails": "Database-1-FinalSnapshot-1569875210793",
    "operationType": "CreateRelationalDatabaseSnapshot",
    "status": "Started",
    "statusChangedAt": 1569875211.029
  },
  {
    "id": "74d73681-30e8-4532-974e-1f23cd3f9f73",
    "resourceName": "Database-1-FinalSnapshot-1569875210793",
    "resourceType": "RelationalDatabaseSnapshot",
    "createdAt": 1569875211.029,
    "location": {
      "availabilityZone": "all",
      "regionName": "us-west-2"
    },
    "isTerminal": false,
    "operationDetails": "Database-1",
    "operationType": "CreateRelationalDatabaseSnapshot",
    "status": "Started",
    "statusChangedAt": 1569875211.029
  }
]
}

```

- For API details, see [DeleteRelationalDatabase](#) in *AWS CLI Command Reference*.

detach-static-ip

The following code example shows how to use `detach-static-ip`.

AWS CLI

To detach a static IP from an instance

The following `detach-static-ip` example detaches static IP `StaticIp-1` from any attached instance.

```
aws lightsail detach-static-ip \  
  --static-ip-name StaticIp-1
```

Output:

```
{  
  "operations": [  
    {  
      "id": "2a43d8a3-9f2d-4fe7-bdd0-eEXAMPLE3cf3",  
      "resourceName": "StaticIp-1",  
      "resourceType": "StaticIp",  
      "createdAt": 1571088261.999,  
      "location": {  
        "availabilityZone": "all",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": true,  
      "operationDetails": "MEAN-1",  
      "operationType": "DetachStaticIp",  
      "status": "Succeeded",  
      "statusChangedAt": 1571088261.999  
    },  
    {  
      "id": "41a7d40c-74e8-4d2e-a837-cEXAMPLEf747",  
      "resourceName": "MEAN-1",  
      "resourceType": "Instance",  
      "createdAt": 1571088262.022,  
      "location": {  
        "availabilityZone": "us-west-2a",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": true,
```

```
        "operationDetails": "StaticIp-1",
        "operationType": "DetachStaticIp",
        "status": "Succeeded",
        "statusChangedAt": 1571088262.022
    }
]
```

- For API details, see [DetachStaticIp](#) in *AWS CLI Command Reference*.

get-active-names

The following code example shows how to use `get-active-names`.

AWS CLI

To get active resource names

The following `get-active-names` example returns the active resource names in the configured AWS Region.

```
aws lightsail get-active-names
```

Output:

```
{
  "activeNames": [
    "WordPress-1",
    "StaticIp-1",
    "MEAN-1",
    "Plesk_Hosting_Stack_on_Ubuntu-1"
  ]
}
```

- For API details, see [GetActiveNames](#) in *AWS CLI Command Reference*.

get-auto-snapshots

The following code example shows how to use `get-auto-snapshots`.

AWS CLI

To get the available automatic snapshots for an instance

The following `get-auto-snapshots` example returns the available automatic snapshots for instance `WordPress-1`.

```
aws lightsail get-auto-snapshots \  
  --resource-name WordPress-1
```

Output:

```
{  
  "resourceName": "WordPress-1",  
  "resourceType": "Instance",  
  "autoSnapshots": [  
    {  
      "date": "2019-10-14",  
      "createdAt": 1571033872.0,  
      "status": "Success",  
      "fromAttachedDisks": []  
    },  
    {  
      "date": "2019-10-13",  
      "createdAt": 1570947473.0,  
      "status": "Success",  
      "fromAttachedDisks": []  
    },  
    {  
      "date": "2019-10-12",  
      "createdAt": 1570861072.0,  
      "status": "Success",  
      "fromAttachedDisks": []  
    },  
    {  
      "date": "2019-10-11",  
      "createdAt": 1570774672.0,  
      "status": "Success",  
      "fromAttachedDisks": []  
    }  
  ]  
}
```

For more information, see [Keeping automatic snapshots of instances or disks in Amazon Lightsail](#) in the *Lightsail Dev Guide*.

- For API details, see [GetAutoSnapshots](#) in *AWS CLI Command Reference*.

get-blueprints

The following code example shows how to use `get-blueprints`.

AWS CLI

To get the blueprints for new instances

The following `get-blueprints` example displays details about all of the available blueprints that can be used to create new instances in Amazon Lightsail.

```
aws lightsail get-blueprints
```

Output:

```
{
  "blueprints": [
    {
      "blueprintId": "wordpress",
      "name": "WordPress",
      "group": "wordpress",
      "type": "app",
      "description": "Bitnami, the leaders in application packaging, and Automattic, the experts behind WordPress, have teamed up to offer this official WordPress image. This image is a pre-configured, ready-to-run image for running WordPress on Amazon Lightsail. WordPress is the world's most popular content management platform. Whether it's for an enterprise or small business website, or a personal or corporate blog, content authors can easily create content using its new Gutenberg editor, and developers can extend the base platform with additional features. Popular plugins like Jetpack, Akismet, All in One SEO Pack, WP Mail, Google Analytics for WordPress, and Amazon Polly are all pre-installed in this image. Let's Encrypt SSL certificates are supported through an auto-configuration script.",
      "isActive": true,
      "minPower": 0,
      "version": "5.2.2-3",
      "versionCode": "1",
      "productUrl": "https://aws.amazon.com/marketplace/pp/B00NN8Y43U",
```

```
    "licenseUrl": "https://d7umqicpi7263.cloudfront.net/eula/
product/7d426cb7-9522-4dd7-a56b-55dd8cc1c8d0/588fd495-6492-4610-b3e8-
d15ce864454c.txt",
    "platform": "LINUX_UNIX"
  },
  {
    "blueprintId": "lamp_7_1_28",
    "name": "LAMP (PHP 7)",
    "group": "lamp_7",
    "type": "app",
    "description": "LAMP with PHP 7.x certified by Bitnami greatly
simplifies the development and deployment of PHP applications. It includes the
latest versions of PHP 7.x, Apache and MySQL together with phpMyAdmin and popular
PHP frameworks Zend, Symfony, CodeIgniter, CakePHP, Smarty, and Laravel. Other pre-
configured components and PHP modules include FastCGI, ModSecurity, SQLite, Varnish,
ImageMagick, xDebug, Xcache, OpenLDAP, Memcache, OAuth, PEAR, PECL, APC, GD and
cURL. It is secure by default and supports multiple applications, each with its own
virtual host and project directory. Let's Encrypt SSL certificates are supported
through an auto-configuration script.",
    "isActive": true,
    "minPower": 0,
    "version": "7.1.28",
    "versionCode": "1",
    "productUrl": "https://aws.amazon.com/marketplace/pp/B072JNJZ5C",
    "licenseUrl": "https://d7umqicpi7263.cloudfront.net/eula/product/
cb6afd05-a3b2-4916-a3e6-bccd414f5f21/12ab56cc-6a8c-4977-9611-dcd770824aad.txt",
    "platform": "LINUX_UNIX"
  },
  {
    "blueprintId": "nodejs",
    "name": "Node.js",
    "group": "node",
    "type": "app",
    "description": "Node.js certified by Bitnami is a pre-configured, ready
to run image for Node.js on Amazon EC2. It includes the latest version of Node.js,
Apache, Python and Redis. The image supports multiple Node.js applications, each
with its own virtual host and project directory. It is configured for production
use and is secure by default, as all ports except HTTP, HTTPS and SSH ports are
closed. Let's Encrypt SSL certificates are supported through an auto-configuration
script. Developers benefit from instant access to a secure, update and consistent
Node.js environment without having to manually install and configure multiple
components and libraries.",
    "isActive": true,
    "minPower": 0,
```

```
        "version": "12.7.0",
        "versionCode": "1",
        "productUrl": "https://aws.amazon.com/marketplace/pp/B00NNZUAK0",
        "licenseUrl": "https://d7umqicpi7263.cloudfront.net/
eula/product/033793fe-951d-47d0-aa94-5fbd0afb3582/25f8fa66-c868-4d80-
adf8-4a2b602064ae.txt",
        "platform": "LINUX_UNIX"
    },
    ...
}
]
```

- For API details, see [GetBlueprints](#) in *AWS CLI Command Reference*.

get-bundles

The following code example shows how to use `get-bundles`.

AWS CLI

To get the bundles for new instances

The following `get-bundles` example displays details about all of the available bundles that can be used to create new instances in Amazon Lightsail.

```
aws lightsail get-bundles
```

Output:

```
{
  "bundles": [
    {
      "price": 3.5,
      "cpuCount": 1,
      "diskSizeInGb": 20,
      "bundleId": "nano_2_0",
      "instanceType": "nano",
      "isActive": true,
      "name": "Nano",
      "power": 300,
    }
  ]
}
```

```
    "ramSizeInGb": 0.5,
    "transferPerMonthInGb": 1024,
    "supportedPlatforms": [
      "LINUX_UNIX"
    ]
  },
  {
    "price": 5.0,
    "cpuCount": 1,
    "diskSizeInGb": 40,
    "bundleId": "micro_2_0",
    "instanceType": "micro",
    "isActive": true,
    "name": "Micro",
    "power": 500,
    "ramSizeInGb": 1.0,
    "transferPerMonthInGb": 2048,
    "supportedPlatforms": [
      "LINUX_UNIX"
    ]
  },
  {
    "price": 10.0,
    "cpuCount": 1,
    "diskSizeInGb": 60,
    "bundleId": "small_2_0",
    "instanceType": "small",
    "isActive": true,
    "name": "Small",
    "power": 1000,
    "ramSizeInGb": 2.0,
    "transferPerMonthInGb": 3072,
    "supportedPlatforms": [
      "LINUX_UNIX"
    ]
  },
  ...
}
]
```

- For API details, see [GetBundles](#) in *AWS CLI Command Reference*.

get-cloud-formation-stack-records

The following code example shows how to use `get-cloud-formation-stack-records`.

AWS CLI

To get the CloudFormation stack records and their associated stacks

The following `get-cloud-formation-stack-records` example displays details about the CloudFormation stack records and their associated stacks used to create Amazon EC2 resources from exported Amazon Lightsail snapshots.

```
aws lightsail get-cloud-formation-stack-records
```

Output:

```
{
  "cloudFormationStackRecords": [
    {
      "name": "CloudFormationStackRecord-588a4243-
e2d1-490d-8200-3a7513ecebdf",
      "arn": "arn:aws:lightsail:us-
west-2:111122223333:CloudFormationStackRecord/28d646ab-27bc-48d9-a422-1EXAMPLE6d37",
      "createdAt": 1565301666.586,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      },
      "resourceType": "CloudFormationStackRecord",
      "state": "Succeeded",
      "sourceInfo": [
        {
          "resourceType": "ExportSnapshotRecord",
          "name": "ExportSnapshotRecord-
e02f23d7-0453-4aa9-9c95-91aa01a141dd",
          "arn": "arn:aws:lightsail:us-
west-2:111122223333:ExportSnapshotRecord/f12b8792-f3ea-4d6f-b547-2EXAMPLE8796"
        }
      ],
      "destinationInfo": {
        "id": "arn:aws:cloudformation:us-west-2:111122223333:stack/
Lightsail-Stack-588a4243-e2d1-490d-8200-3EXAMPLEebdf/063203b0-
ba28-11e9-838b-0EXAMPLE8b00",
```

```
        "service": "Aws::CloudFormation::Stack"
      }
    ]
  }
```

- For API details, see [GetCloudFormationStackRecords](#) in *AWS CLI Command Reference*.

get-disk-snapshot

The following code example shows how to use `get-disk-snapshot`.

AWS CLI

To get information about a disk snapshot

The following `get-disk-snapshot` example displays details about the disk snapshot `Disk-1-1566839161`.

```
aws lightsail get-disk-snapshot \
  --disk-snapshot-name Disk-1-1566839161
```

Output:

```
{
  "diskSnapshot": {
    "name": "Disk-1-1566839161",
    "arn": "arn:aws:lightsail:us-west-2:111122223333:DiskSnapshot/
e2d0fa53-8ee0-41a0-8e56-0EXAMPLE1051",
    "supportCode": "6EXAMPLE3362/snap-0EXAMPLE06100d09",
    "createdAt": 1566839163.749,
    "location": {
      "availabilityZone": "all",
      "regionName": "us-west-2"
    },
    "resourceType": "DiskSnapshot",
    "tags": [],
    "sizeInGb": 8,
    "state": "completed",
    "progress": "100%",
    "fromDiskName": "Disk-1",
```

```
    "fromDiskArn": "arn:aws:lightsail:us-west-2:111122223333:Disk/
c21cfb0a-07f2-44ae-9a23-bEXAMPLE8096",
    "isFromAutoSnapshot": false
  }
}
```

For more information, see title in the *guide*.

- For API details, see [GetDiskSnapshot](#) in *AWS CLI Command Reference*.

get-disk-snapshots

The following code example shows how to use `get-disk-snapshots`.

AWS CLI

To get information about all disk snapshots

The following `get-disk-snapshots` example displays details about all of the disk snapshots in the configured AWS Region.

```
aws lightsail get-disk-snapshots
```

Output:

```
{
  "diskSnapshots": [
    {
      "name": "Disk-2-1571090588",
      "arn": "arn:aws:lightsail:us-
west-2:111122223333:DiskSnapshot/32e889a9-38d4-4687-9f21-eEXAMPLE7839",
      "supportCode": "6EXAMPLE3362/snap-0EXAMPLE1ca192a4",
      "createdAt": 1571090591.226,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      },
      "resourceType": "DiskSnapshot",
      "tags": [],
      "sizeInGb": 8,
      "state": "completed",
      "progress": "100%",
```



```

        "fromDiskName": "Disk-2",
        "fromDiskArn": "arn:aws:lightsail:us-
west-2:111122223333:Disk/6a343ff8-6341-422d-86e2-bEXAMPLE16c2",
        "isFromAutoSnapshot": false
    },
    {
        "name": "Disk-1-1566839161",
        "arn": "arn:aws:lightsail:us-west-2:111122223333:DiskSnapshot/
e2d0fa53-8ee0-41a0-8e56-0EXAMPLE1051",
        "supportCode": "6EXAMPLE3362/snap-0EXAMPLEe06100d09",
        "createdAt": 1566839163.749,
        "location": {
            "availabilityZone": "all",
            "regionName": "us-west-2"
        },
        "resourceType": "DiskSnapshot",
        "tags": [],
        "sizeInGb": 8,
        "state": "completed",
        "progress": "100%",
        "fromDiskName": "Disk-1",
        "fromDiskArn": "arn:aws:lightsail:us-west-2:111122223333:Disk/
c21cfb0a-07f2-44ae-9a23-bEXAMPLE8096",
        "isFromAutoSnapshot": false
    }
]
}

```

- For API details, see [GetDiskSnapshots](#) in *AWS CLI Command Reference*.

get-disk

The following code example shows how to use `get-disk`.

AWS CLI

To get information about a block storage disk

The following `get-disk` example displays details about the disk `Disk-1`.

```
aws lightsail get-disk \
  --disk-name Disk-1
```

Output:

```
{
  "disk": {
    "name": "Disk-1",
    "arn": "arn:aws:lightsail:us-west-2:111122223333:Disk/
c21cfb0a-07f2-44ae-9a23-bEXAMPLE8096",
    "supportCode": "6EXAMPLE3362/vol-0EXAMPLEf2f88b32f",
    "createdAt": 1566585439.587,
    "location": {
      "availabilityZone": "us-west-2a",
      "regionName": "us-west-2"
    },
    "resourceType": "Disk",
    "tags": [],
    "sizeInGb": 8,
    "isSystemDisk": false,
    "iops": 100,
    "path": "/dev/xvdf",
    "state": "in-use",
    "attachedTo": "WordPress_Multisite-1",
    "isAttached": true,
    "attachmentState": "attached"
  }
}
```

For more information, see title in the *guide*.

- For API details, see [GetDisk](#) in *AWS CLI Command Reference*.

get-disks

The following code example shows how to use `get-disks`.

AWS CLI

To get information about all block storage disks

The following `get-disks` example displays details about all of the disks in the configured AWS Region.

```
aws lightsail get-disks
```

Output:

```
{
  "disks": [
    {
      "name": "Disk-2",
      "arn": "arn:aws:lightsail:us-west-2:111122223333:Disk/6a343ff8-6341-422d-86e2-bEXAMPLE16c2",
      "supportCode": "6EXAMPLE3362/vol-0EXAMPLE929602087",
      "createdAt": 1571090461.634,
      "location": {
        "availabilityZone": "us-west-2a",
        "regionName": "us-west-2"
      },
      "resourceType": "Disk",
      "tags": [],
      "sizeInGb": 8,
      "isSystemDisk": false,
      "iops": 100,
      "state": "available",
      "isAttached": false,
      "attachmentState": "detached"
    },
    {
      "name": "Disk-1",
      "arn": "arn:aws:lightsail:us-west-2:111122223333:Disk/c21cfb0a-07f2-44ae-9a23-bEXAMPLE8096",
      "supportCode": "6EXAMPLE3362/vol-0EXAMPLEf2f88b32f",
      "createdAt": 1566585439.587,
      "location": {
        "availabilityZone": "us-west-2a",
        "regionName": "us-west-2"
      },
      "resourceType": "Disk",
      "tags": [],
      "sizeInGb": 8,
      "isSystemDisk": false,
      "iops": 100,
      "path": "/dev/xvdf",
      "state": "in-use",
      "attachedTo": "WordPress_Multisite-1",
      "isAttached": true,
      "attachmentState": "attached"
    }
  ]
}
```

```
]
}
```

- For API details, see [GetDisks](#) in *AWS CLI Command Reference*.

get-domain

The following code example shows how to use `get-domain`.

AWS CLI

To get information about a domain

The following `get-domain` example displays details about the domain `example.com`.

Note: Lightsail's domain-related API operations are available in only the `us-east-1` AWS Region. If your CLI profile is configured to use a different Region, you must include the `--region us-east-1` parameter or the command fails.

```
aws lightsail get-domain \
  --domain-name example.com \
  --region us-east-1
```

Output:

```
{
  "domain": {
    "name": "example.com",
    "arn":
"arn:aws:lightsail:global:111122223333:Domain/28cda903-3f15-44b2-9baf-3EXAMPLEb304",
    "supportCode": "6EXAMPLE3362//hostedzone/ZEXAMPLEONGSC1",
    "createdAt": 1570728588.6,
    "location": {
      "availabilityZone": "all",
      "regionName": "global"
    },
    "resourceType": "Domain",
    "tags": [],
    "domainEntries": [
      {
        "id": "-1682899164",
        "name": "example.com",
```

```
        "target": "192.0.2.0",
        "isAlias": false,
        "type": "A"
    },
    {
        "id": "1703104243",
        "name": "example.com",
        "target": "ns-137.awsdns-17.com",
        "isAlias": false,
        "type": "NS"
    },
    {
        "id": "-1038331153",
        "name": "example.com",
        "target": "ns-1710.awsdns-21.co.uk",
        "isAlias": false,
        "type": "NS"
    },
    {
        "id": "-2107289565",
        "name": "example.com",
        "target": "ns-692.awsdns-22.net",
        "isAlias": false,
        "type": "NS"
    },
    {
        "id": "1582095705",
        "name": "example.com",
        "target": "ns-1436.awsdns-51.org",
        "isAlias": false,
        "type": "NS"
    },
    {
        "id": "-1769796132",
        "name": "example.com",
        "target": "ns-1710.awsdns-21.co.uk. awsdns-hostmaster.amazon.com. 1
7200 900 1209600 86400",
        "isAlias": false,
        "type": "SOA"
    }
}
]
```

- For API details, see [GetDomain](#) in *AWS CLI Command Reference*.

get-domains

The following code example shows how to use `get-domains`.

AWS CLI

To get information about all domains

The following `get-domains` example displays details about all of the domains in the configured AWS Region.

Note: Lightsail's domain-related API operations are available in only the `us-east-1` AWS Region. If your CLI profile is configured to use a different Region, you must include the `--region us-east-1` parameter or the command fails.

```
aws lightsail get-domains \  
  --region us-east-1
```

Output:

```
{  
  "domains": [  
    {  
      "name": "example.com",  
      "arn":  
"arn:aws:lightsail:global:111122223333:Domain/28cda903-3f15-44b2-9baf-3EXAMPLEb304",  
      "supportCode": "6EXAMPLE3362//hostedzone/ZEXAMPLEONGSC1",  
      "createdAt": 1570728588.6,  
      "location": {  
        "availabilityZone": "all",  
        "regionName": "global"  
      },  
      "resourceType": "Domain",  
      "tags": [],  
      "domainEntries": [  
        {  
          "id": "-1682899164",  
          "name": "example.com",  
          "target": "192.0.2.0",  
          "isAlias": false,  
        }  
      ]  
    }  
  ]  
}
```

```
    "type": "A"
  },
  {
    "id": "1703104243",
    "name": "example.com",
    "target": "ns-137.awsdns-17.com",
    "isAlias": false,
    "type": "NS"
  },
  {
    "id": "-1038331153",
    "name": "example.com",
    "target": "ns-4567.awsdns-21.co.uk",
    "isAlias": false,
    "type": "NS"
  },
  {
    "id": "-2107289565",
    "name": "example.com",
    "target": "ns-333.awsdns-22.net",
    "isAlias": false,
    "type": "NS"
  },
  {
    "id": "1582095705",
    "name": "example.com",
    "target": "ns-1111.awsdns-51.org",
    "isAlias": false,
    "type": "NS"
  },
  {
    "id": "-1769796132",
    "name": "example.com",
    "target": "ns-1234.awsdns-21.co.uk. awsdns-
hostmaster.amazon.com. 1 7200 900 1209600 86400",
    "isAlias": false,
    "type": "SOA"
  },
  {
    "id": "1029454894",
    "name": "_dead6a124ede046a0319eb44a4eb3cbc.example.com",
    "target": "_be133b0a0899fb7b6bf79d9741d1a383.hkvuijqjoua.acm-
validations.aws",
    "isAlias": false,
```

```
        "type": "CNAME"
      }
    ]
  },
  {
    "name": "example.net",
    "arn": "arn:aws:lightsail:global:111122223333:Domain/9c9f0d70-
c92e-4753-86c2-6EXAMPLE029d",
    "supportCode": "6EXAMPLE3362//hostedzone/ZEXAMPLE5TPKMV",
    "createdAt": 1556661071.384,
    "location": {
      "availabilityZone": "all",
      "regionName": "global"
    },
    "resourceType": "Domain",
    "tags": [],
    "domainEntries": [
      {
        "id": "-766320943",
        "name": "example.net",
        "target": "192.0.2.2",
        "isAlias": false,
        "type": "A"
      },
      {
        "id": "-453913825",
        "name": "example.net",
        "target": "ns-123.awsdns-10.net",
        "isAlias": false,
        "type": "NS"
      },
      {
        "id": "1553601564",
        "name": "example.net",
        "target": "ns-4444.awsdns-47.co.uk",
        "isAlias": false,
        "type": "NS"
      },
      {
        "id": "1653797661",
        "name": "example.net",
        "target": "ns-7890.awsdns-61.org",
        "isAlias": false,
        "type": "NS"
      }
    ]
  }
]
```



```
    },
    {
      "id": "706414698",
      "name": "example.net",
      "target": "ns-123.awsdns-44.com",
      "isAlias": false,
      "type": "NS"
    },
    {
      "id": "337271745",
      "name": "example.net",
      "target": "ns-4444.awsdns-47.co.uk. awsdns-
hostmaster.amazon.com. 1 7200 900 1209600 86400",
      "isAlias": false,
      "type": "SOA"
    },
    {
      "id": "-1785431096",
      "name": "www.example.net",
      "target": "192.0.2.2",
      "isAlias": false,
      "type": "A"
    }
  ]
},
{
  "name": "example.org",
  "arn": "arn:aws:lightsail:global:111122223333:Domain/
f0f13ba3-3df0-4fdc-8ebb-1EXAMPLEf26e",
  "supportCode": "6EXAMPLE3362//hostedzone/ZEXAMPLEAF038",
  "createdAt": 1556661199.106,
  "location": {
    "availabilityZone": "all",
    "regionName": "global"
  },
  "resourceType": "Domain",
  "tags": [],
  "domainEntries": [
    {
      "id": "2065301345",
      "name": "example.org",
      "target": "192.0.2.4",
      "isAlias": false,
      "type": "A"
    }
  ]
}
```

```
    },
    {
      "id": "-447198516",
      "name": "example.org",
      "target": "ns-123.awsdns-45.com",
      "isAlias": false,
      "type": "NS"
    },
    {
      "id": "136463022",
      "name": "example.org",
      "target": "ns-9999.awsdns-15.co.uk",
      "isAlias": false,
      "type": "NS"
    },
    {
      "id": "1395941679",
      "name": "example.org",
      "target": "ns-555.awsdns-01.net",
      "isAlias": false,
      "type": "NS"
    },
    {
      "id": "872052569",
      "name": "example.org",
      "target": "ns-6543.awsdns-38.org",
      "isAlias": false,
      "type": "NS"
    },
    {
      "id": "1001949377",
      "name": "example.org",
      "target": "ns-1234.awsdns-15.co.uk. awsdns-
hostmaster.amazon.com. 1 7200 900 1209600 86400",
      "isAlias": false,
      "type": "SOA"
    },
    {
      "id": "1046191192",
      "name": "www.example.org",
      "target": "192.0.2.4",
      "isAlias": false,
      "type": "A"
    }
  }
```

```

    ]
  }
]
}

```

- For API details, see [GetDomains](#) in *AWS CLI Command Reference*.

get-export-snapshot-record

The following code example shows how to use `get-export-snapshot-record`.

AWS CLI

To get the records of snapshots exported to Amazon EC2

The following `get-export-snapshot-record` example displays details about Amazon Lightsail instance or disk snapshots exported to Amazon EC2.

```
aws lightsail get-export-snapshot-records
```

Output:

```

{
  "exportSnapshotRecords": [
    {
      "name": "ExportSnapshotRecord-d2da10ce-0b3c-4ae1-ab3a-2EXAMPLEa586",
      "arn": "arn:aws:lightsail:us-west-2:111122223333:ExportSnapshotRecord/076c7060-b0cc-4162-98f0-2EXAMPLEe28e",
      "createdAt": 1543534665.678,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      },
      "resourceType": "ExportSnapshotRecord",
      "state": "Succeeded",
      "sourceInfo": {
        "resourceType": "InstanceSnapshot",
        "createdAt": 1540339310.706,
        "name": "WordPress-512MB-0regon-1-1540339219",
        "arn": "arn:aws:lightsail:us-west-2:111122223333:InstanceSnapshot/5446f534-ed60-4c17-b4a5-bEXAMPLEf8b7",
        "fromResourceName": "WordPress-512MB-0regon-1",

```

```

        "fromResourceArn": "arn:aws:lightsail:us-
west-2:111122223333:Instance/4b8f1f24-e4d1-4cf3-88ff-cEXAMPLEa397",
        "instanceSnapshotInfo": {
            "fromBundleId": "nano_2_0",
            "fromBlueprintId": "wordpress_4_9_8",
            "fromDiskInfo": [
                {
                    "path": "/dev/sda1",
                    "sizeInGb": 20,
                    "isSystemDisk": true
                }
            ]
        }
    },
    "destinationInfo": {
        "id": "ami-0EXAMPLEc0d65058e",
        "service": "Aws::EC2::Image"
    }
},
{
    "name": "ExportSnapshotRecord-1c94e884-40ff-4fe1-9302-0EXAMPLE14c2",
    "arn": "arn:aws:lightsail:us-west-2:111122223333:ExportSnapshotRecord/
fb392ce8-6567-4013-9bfd-3EXAMPLE5b4c",
    "createdAt": 1543432110.2,
    "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
    },
    "resourceType": "ExportSnapshotRecord",
    "state": "Succeeded",
    "sourceInfo": {
        "resourceType": "InstanceSnapshot",
        "createdAt": 1540833603.545,
        "name": "LAMP_PHP_5-512MB-0regon-1-1540833565",
        "arn": "arn:aws:lightsail:us-
west-2:111122223333:InstanceSnapshot/82334399-b5f2-49ec-8382-0EXAMPLEe45f",
        "fromResourceName": "LAMP_PHP_5-512MB-0regon-1",
        "fromResourceArn": "arn:aws:lightsail:us-
west-2:111122223333:Instance/863b9f35-ab1e-4418-bdd2-1EXAMPLEbab2",
        "instanceSnapshotInfo": {
            "fromBundleId": "nano_2_0",
            "fromBlueprintId": "lamp_5_6_37_2",
            "fromDiskInfo": [
                {

```

```

        "path": "/dev/sda1",
        "sizeInGb": 20,
        "isSystemDisk": true
      }
    ]
  },
  "destinationInfo": {
    "id": "ami-0EXAMPLE7c5ec84e2",
    "service": "Aws::EC2::Image"
  }
}
]
}

```

- For API details, see [GetExportSnapshotRecord](#) in *AWS CLI Command Reference*.

get-instance-access-details

The following code example shows how to use `get-instance-access-details`.

AWS CLI

To get host key information for an instance

The following `get-instance-access-details` example displays host key information for instance `WordPress_Multisite-1`.

```
aws lightsail get-instance-access-details \
  --instance-name WordPress_Multisite-1
```

Output:

```
{
  "accessDetails": {
    "certKey": "ssh-rsa-cert-v01@openssh.com
AEXAMPLEEaC1yc2EtY2VydC12MDFAb3B1bnNzaC5jb20AAAAgNf076Dt3ppmPd0fPxZVMmS491aEAYYH9cHqAJ3fNML8
vEXAMPLE2eBWJyQvn7o1/
i0+s966h5sx8qUD791PB7q5UESd5VZGFtytrykfQJnjiwqe7EV5agzvjb1Lj26Fb37EKda9HVfCOu8pWbvky7Tyn9w29
+xFmQM9xVz0rXZmqx8uJidJpRgLCMTviofwQJU/
K1EXAMPLEAAAAAAAAABAAAALS00MzMzMDU4MzA4ODg1MTY2NjM4Onp6UWlndHk4UElRSG9Stit0TG5QSEE9PQAAAAAsAAA
+LiB+ozNbUA0cdNL9Y67x7qPv/R7XhTc21+2A+8+GuVpK/Kz9dqDMKNAEXAMPLE+YYN
```

```

+tiXm7Y80gziK+7iDB7xUuQ4vghmn4+qgz9mKwYgWvVe2+0XLuV7cnWPB7iU1HQg
+E3LUKrV4ZFw9pj7X2dFdNkFMxwWgI1ISWKimEXAMPLERhf1Rqc/
QH6TpWCvPfcx8uvwVqdwTfke/SfA5BCzbGGI1UmIUadh8nHcb5FamQ1hK7kECy47K/x9FMn/
KwmM7pCwJbSLDM07n9bnbvck6m8ZoB2N2YLMG5dW7BerEXAMPLERbqfdtyYJHHe11EyyEJs1fWNU3D5JIGlgzcPAV
+Z1bQyUCZXf0os1Sa+HE85f0/
FRq9SVSBSHrmbeb0frlPhgMzgSmqLeyhlbr6wwWIDbREXAMPLEJZ49H7RdQxdKyYrZPwvRgcr0qI2EL0tAajnpQQ8UZo
Aqter0xN5PhFL0J490WTacwCGRAjLhibAx7K1t/1ZXWo6c+ijq8c111327EXAMPLER/
e89GC89KcmKCxfGQniDAUGf8UqofIbq3Z0UgiAAYCVXc1I4L68NhVXyoWuQXPBRQSEXAMPLERm74tDL9tFN3c7tSe/
Oz0cTR+4sAAAIPAAAAB3NzaC1yc2EAAAIAQnG/
L0DqiSnLrWhEox4aHqMgd0m0oLLAYx60QH9F0TM9EXAMPLER961rzSCMon7ZgsWnNl00wZQgDG
+rtJ4N0B7H0Vwns4ynUFbzNQ3qFGGeE31KwX1L41vV1iSy7sDk8aI0LmrKJi1LE1Qc118uboR1woX0YEXAMPLERaUCeX
+10+WEXAMPLER6Y4U4ZvE2B3xyRdpvysb5TGFNTk5qPs1acnVkoL0GsZZXMPLGJnG40BpQLLtpj9sNMxAgZPCAUjhkqk
+nx0904NUZ2pTwbVSUaV1gm6pug9xbwN01Im21t34JeLlKTqxcJ6zzS8W0c0KKpAm5c4hWkseMbyutS2jav/4hiS
+BhrYgptzfwe5qRXEXAMPLERHzQr3YfGzYoBJ/
1LK3NHhx0ihhsfAYwMei0BFZT1F/7CT3IH4iitEkIgodio6/
Mw6UDqMPozyQCK11EA6LFhYCOZG9drWcoRa741M4kY9TP028Za8gDMh1WpkXLq9Gixon50HP8aM/
sEXAMPLER2+fnkw+1Bto05L6+VKoPlXaGqZ/fBYEXAMPLERAMQHjnLM1JYNvtEePhp+TNzXHzuixWf/
Ht04m0AVpXrzIDXaS102tXY=",
    "ipAddress": "192.0.2.0",
    "privateKey": "-----BEGIN RSA PRIVATE KEY-----
\nEXAMPLERBAAKCAQEA+AD3qeU2toBy505v7wnRLVo/tngVickL5+6Jf4tPrPeuoebM
\nfK1A+/ZTwe6uVBENEVRhbcra8pH0CZ44sKnuxFeWoM7425S49uhW9+xCnWvR1Xw
\njrvKvm75Mu08p/cNvfwugrBuaPB65DspgxNnofZWMVxpIpSq0SPWmSwQHV597d6C
\nrEXAMPLERo8hJmqz2KFQ09X7fB21BruGgr9aXiNPmWmovYKqWfmrnFvR7odFmDecq
\n5EXAMPLER9dyU1ZsrWhGby77eYrVaF10GNGQ8qy1HGUIScquZ9NDIL49n4mXbfsTH
\n0EXAMPLER12ZqsfLiYnSaUYCwjE74qH8ECVPytQIDAQABaoIBAHeZV9Z58JHAjifz
\nCEXAMPLEREqC3do0VDgXS1kKI92qNo4z2VcUEho878paCuVVXVHcCGgSnGeyIh2tN
\nMEXAMPLERsOhR427BhH3YLA+3Z5SIVnejbTgYPfLC37B8khTaYqkqMvdZiFVZK5qn
\nIEXAMPLERm93oF9eSZCjckLB/jGHsfb0eCDMP8BshHE2beuqzVMoK1Dx0nvoP3+Fp
\nAEXAMPLERsq6pDpCo9YVUX8g1u3Ro9cP12LXHDy+oVEY5KhbZQJ7VU1I72W0vppWW
\n0EXAMPLERkgY1q7p6qYtYcSgTEjz14gDiMfQ7SyHB3alkIoNONQ9ZPaWHyJvymeud
\noQTNuz0CgYEA/LFWNTEZrzdzdR1kJmyNRmAermU0B6utyNENChAlHGSHkB+11VSh
\nbEXAMPLERQo9ooUeW5Ux03YwacZLoDT1mwxw1PtC1+PNycZoLe1fE9UdARrdmGTob
\n8L7CPLSXp3xuR8VqSp2fnIc7hfiQs/NrPX9gm/E0rB0we0RkyDSzWScCgYEA+z/r
\niob+nJZq0Ybn0SuP6oMULP4vnWniWj8MIhUJU53LwSAM8DeJd0NKDdkui0d52aAL
\nVgn7nLo88rVWKhJwVc4tu/rNgZLcR3bP4+kL6zand0KQnMly0zNA2Ys26aa5udH1\nnqWl0WTt9WEm/
h10ndC1kn0MectrvsG17b38y5sMCgYEA54NiRGGz8oCPW6GN/FZA
\nKEXAMPLER5tw34GEH3Uxlcn3CejDaQmzc0ATwX4nIwRZDEqWyYZcS0btg1jhGiBD\nYEXAMPLERkc8Z71L/
agZEAaVCEog9FqfSqwB
+XTfoK8hQur74X1yCu9p6gof1q6k9\nneEXAMPLERchJcNN0g4ETIfMkCgYBdVORRHE4mqvWp0dzA7v66FdEz2YSkjAXKk
\naEXAMPLER8Z/8yBSmuBv1Qv03XA12my462uB92uzzGAuW
+1yBc2Kn1sXqYTy0y1z0\nngEXAMPLERBogjw4MqHKL1bPKMHyQU8/
q24PaYgzHPzy13w1H6pTYf1Xq1HdE2D6Vv\nYEXAMPLERgQC3i/
kVvhky/2XRwRVLC7J02Bg3QGTx38hpmDa5IuofKANjA+Wa3/zy\nbEXAMPLER6ytQgD9GN/YtBq+uh0

```

```
+ZzkvXPL+CWRi0ZRxpPwYDBBFU9Cw0AuWWG1L8\nwEXAMPLExM1cysRgcWB9RNgf3AuOpFd2i6XT/
riNsvvkpmJ+VooU8g==\n-----END RSA PRIVATE KEY-----\n",
  "protocol": "ssh",
  "instanceName": "WordPress_Multisite-1",
  "username": "bitnami",
  "hostKeys": [
    {
      "algorithm": "ssh-rsa",
      "publicKey":
"AEXAMPLEaC1yc2EAAAAADAQABAAABAQCoer9ieZTjQ3pXCHczuAYZFj1F7t
+uBkXuqeGMREx78pCvmS+DiEXAMPLEEuJ1Q8dcKhrQL4HpXbD9dosVCTaJnJwb4MQqsuSVFdHFzy3guP
+BKclWqtXJEXAMPLEsBGqZZlrIv6a9bTA0TCplZ8AD+hSRTaSXXqg6FT
+Qf16IktH0X1Ms7xIEXAMPLEmNtjCpzZiGXDHzytoMvUgwa8uHPp440g36EUu4VqQxoUHPJKoXvcQizyk3K8ym0hP0Tp
0t6y9HwvykEXAMPLEAfbKjbr42+u6+0Slkr4d339q2U1sTDytJhhs8HUel1wTfGRfp",
      "witnessedAt": 1570744377.699,
      "fingerprintSHA1": "SHA1:GEXAMPLEMoYgUg0ucadqU9Bt3Lk",
      "fingerprintSHA256": "SHA256:IEEXAMPLEcB5vgxnAUoJawbdZ
+MwELhIp6FUxuwq/LIU"
    },
    {
      "algorithm": "ssh-ed25519",
      "publicKey":
"AEXAMPLEaC11ZDI1NTE5AAAAIC1gwGPDfGa0NxEXAMPLEJX3UNap781QxHQmn8nzlrUv",
      "witnessedAt": 1570744377.697,
      "fingerprintSHA1": "SHA1:VEXAMPLE5ReqSmTgv03sSUw9toU",
      "fingerprintSHA256": "SHA256:0EXAMPLEdE6tI95k3TJpG
+qhJbAoknB0yz9nAEaDt3A"
    },
    {
      "algorithm": "ecdsa-sha2-nistp256",
      "publicKey":
"AEXAMPLEZHNhLXNoYTIItbmLzdHAyNTYAAAAIbmlzdHAyNTYAAABEXAMPLE9B4mZy8YSsZW7cixCDq5yHSAAxjJkDo5
+EnK1DCsYtUkxxEXAMPLE6V0WL2z63RTKa2AUPgd8irjxwI=",
      "witnessedAt": 1570744377.707,
      "fingerprintSHA1": "SHA1:UEXAMPLE0YCFxScf2G6tDg+7YG0",
      "fingerprintSHA256": "SHA256:wEXAMPLEQ9a/
iEXAMPLEhRufm6U9vFU4cpkMPHnBsNA"
    }
  ]
}
}
```

- For API details, see [GetInstanceAccessDetails](#) in *AWS CLI Command Reference*.

get-instance-metric-data

The following code example shows how to use `get-instance-metric-data`.

AWS CLI

To get metric data for an instance

The following `get-instance-metric-data` example returns the average percent of CPUUtilization every 7200 seconds (2 hours) between 1571342400 and 1571428800 for instance MEAN-1.

We recommend that you use a unix time converter to identify the start and end times.

```
aws lightsail get-instance-metric-data \  
  --instance-name MEAN-1 \  
  --metric-name CPUUtilization \  
  --period 7200 \  
  --start-time 1571342400 \  
  --end-time 1571428800 \  
  --unit Percent \  
  --statistics Average
```

Output:

```
{  
  "metricName": "CPUUtilization",  
  "metricData": [  
    {  
      "average": 0.26113718770120725,  
      "timestamp": 1571342400.0,  
      "unit": "Percent"  
    },  
    {  
      "average": 0.26861268928111953,  
      "timestamp": 1571392800.0,  
      "unit": "Percent"  
    },  
    {  
      "average": 0.28187475104748777,  
      "timestamp": 1571378400.0,  
      "unit": "Percent"  
    }  
  ],  
}
```



```
{
  "average": 0.2651936960458352,
  "timestamp": 1571421600.0,
  "unit": "Percent"
},
{
  "average": 0.2561856213712188,
  "timestamp": 1571371200.0,
  "unit": "Percent"
},
{
  "average": 0.3021383254607764,
  "timestamp": 1571356800.0,
  "unit": "Percent"
},
{
  "average": 0.2618381649223539,
  "timestamp": 1571407200.0,
  "unit": "Percent"
},
{
  "average": 0.26331929394825787,
  "timestamp": 1571400000.0,
  "unit": "Percent"
},
{
  "average": 0.2576348407007818,
  "timestamp": 1571385600.0,
  "unit": "Percent"
},
{
  "average": 0.2513008454658378,
  "timestamp": 1571364000.0,
  "unit": "Percent"
},
{
  "average": 0.26329974562758346,
  "timestamp": 1571414400.0,
  "unit": "Percent"
},
{
  "average": 0.2667092536656445,
  "timestamp": 1571349600.0,
  "unit": "Percent"
}
```

```
    }  
  ]  
}
```

- For API details, see [GetInstanceMetricData](#) in *AWS CLI Command Reference*.

get-instance-port-states

The following code example shows how to use `get-instance-port-states`.

AWS CLI

To get firewall information for an instance

The following `get-instance-port-states` example returns the firewall ports configured for instance MEAN-1.

```
aws lightsail get-instance-port-states \  
  --instance-name MEAN-1
```

Output:

```
{  
  "portStates": [  
    {  
      "fromPort": 80,  
      "toPort": 80,  
      "protocol": "tcp",  
      "state": "open"  
    },  
    {  
      "fromPort": 22,  
      "toPort": 22,  
      "protocol": "tcp",  
      "state": "open"  
    },  
    {  
      "fromPort": 443,  
      "toPort": 443,  
      "protocol": "tcp",  
      "state": "open"  
    }  
  ]  
}
```

```
]
}
```

- For API details, see [GetInstancePortStates](#) in *AWS CLI Command Reference*.

get-instance-snapshot

The following code example shows how to use `get-instance-snapshot`.

AWS CLI

To get information about a specified instance snapshot

The following `get-instance-snapshot` example displays details about the specified instance snapshot.

```
aws lightsail get-instance-snapshot \
  --instance-snapshot-name MEAN-1-1571419854
```

Output:

```
{
  "instanceSnapshot": {
    "name": "MEAN-1-1571419854",
    "arn": "arn:aws:lightsail:us-west-2:111122223333:InstanceSnapshot/
ac54700c-48a8-40fd-b065-2EXAMPLEac8f",
    "supportCode": "6EXAMPLE3362/ami-0EXAMPLE67a73020d",
    "createdAt": 1571419891.927,
    "location": {
      "availabilityZone": "all",
      "regionName": "us-west-2"
    },
    "resourceType": "InstanceSnapshot",
    "tags": [],
    "state": "available",
    "fromAttachedDisks": [],
    "fromInstanceName": "MEAN-1",
    "fromInstanceArn": "arn:aws:lightsail:us-west-2:111122223333:Instance/
bd470fc5-a68b-44c5-8dbc-8EXAMPLEebada",
    "fromBlueprintId": "mean_4_0_9",
    "fromBundleId": "medium_2_0",
    "isFromAutoSnapshot": false,
```

```
    "sizeInGb": 80
  }
}
```

- For API details, see [GetInstanceSnapshot](#) in *AWS CLI Command Reference*.

get-instance-snapshots

The following code example shows how to use `get-instance-snapshots`.

AWS CLI

To get information about all of your instance snapshots

The following `get-instance-snapshots` example displays details about all of the instance snapshots in the configured AWS Region.

```
aws lightsail get-instance-snapshots
```

Output:

```
{
  "instanceSnapshots": [
    {
      "name": "MEAN-1-1571421498",
      "arn": "arn:aws:lightsail:us-west-2:111122223333:InstanceSnapshot/a20e6ebe-b0ee-4ae4-a750-3EXAMPLEcb0c",
      "supportCode": "6EXAMPLE3362/ami-0EXAMPLEe33cabfa1",
      "createdAt": 1571421527.755,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      },
      "resourceType": "InstanceSnapshot",
      "tags": [
        {
          "key": "no_delete"
        }
      ],
      "state": "available",
      "fromAttachedDisks": [],
      "fromInstanceName": "MEAN-1",
    }
  ]
}
```

```

        "fromInstanceArn": "arn:aws:lightsail:us-
west-2:111122223333:Instance/1761aa0a-6038-4f25-8b94-2EXAMPLE19fd",
        "fromBlueprintId": "wordpress_5_1_1_2",
        "fromBundleId": "micro_2_0",
        "isFromAutoSnapshot": false,
        "sizeInGb": 40
    },
    {
        "name": "MEAN-1-1571419854",
        "arn": "arn:aws:lightsail:us-west-2:111122223333:InstanceSnapshot/
ac54700c-48a8-40fd-b065-2EXAMPLEac8f",
        "supportCode": "6EXAMPLE3362/ami-0EXAMPLE67a73020d",
        "createdAt": 1571419891.927,
        "location": {
            "availabilityZone": "all",
            "regionName": "us-west-2"
        },
        "resourceType": "InstanceSnapshot",
        "tags": [],
        "state": "available",
        "fromAttachedDisks": [],
        "fromInstanceName": "MEAN-1",
        "fromInstanceArn": "arn:aws:lightsail:us-west-2:111122223333:Instance/
bd470fc5-a68b-44c5-8dbc-8EXAMPLEbada",
        "fromBlueprintId": "mean_4_0_9",
        "fromBundleId": "medium_2_0",
        "isFromAutoSnapshot": false,
        "sizeInGb": 80
    }
]
}

```

- For API details, see [GetInstanceSnapshots](#) in *AWS CLI Command Reference*.

get-instance-state

The following code example shows how to use `get-instance-state`.

AWS CLI

To get information about the state of an instance

The following `get-instance-state` example returns the state of the specified instance.

```
aws lightsail get-instance-state \  
  --instance-name MEAN-1
```

Output:

```
{  
  "state": {  
    "code": 16,  
    "name": "running"  
  }  
}
```

- For API details, see [GetInstanceState](#) in *AWS CLI Command Reference*.

get-instance

The following code example shows how to use `get-instance`.

AWS CLI**To get information about an instance**

The following `get-instance` example displays details about the instance MEAN-1.

```
aws lightsail get-instance \  
  --instance-name MEAN-1
```

Output:

```
{  
  "instance": {  
    "name": "MEAN-1",  
    "arn": "arn:aws:lightsail:us-west-2:111122223333:Instance/bd470fc5-  
a68b-44c5-8dbc-EXAMPLE4bada",  
    "supportCode": "6EXAMPLE3362/i-05EXAMPLE407c97d3",  
    "createdAt": 1570635023.124,  
    "location": {  
      "availabilityZone": "us-west-2a",  
      "regionName": "us-west-2"  
    },  
    "resourceType": "Instance",  
    "tags": [],
```

```
"blueprintId": "mean_4_0_9",
"blueprintName": "MEAN",
"bundleId": "medium_2_0",
"isStaticIp": false,
"privateIpAddress": "192.0.2.0",
"publicIpAddress": "192.0.2.0",
"hardware": {
  "cpuCount": 2,
  "disks": [
    {
      "createdAt": 1570635023.124,
      "sizeInGb": 80,
      "isSystemDisk": true,
      "iops": 240,
      "path": "/dev/sda1",
      "attachedTo": "MEAN-1",
      "attachmentState": "attached"
    }
  ],
  "ramSizeInGb": 4.0
},
"networking": {
  "monthlyTransfer": {
    "gbPerMonthAllocated": 4096
  },
  "ports": [
    {
      "fromPort": 80,
      "toPort": 80,
      "protocol": "tcp",
      "accessFrom": "Anywhere (0.0.0.0/0)",
      "accessType": "public",
      "commonName": "",
      "accessDirection": "inbound"
    },
    {
      "fromPort": 22,
      "toPort": 22,
      "protocol": "tcp",
      "accessFrom": "Anywhere (0.0.0.0/0)",
      "accessType": "public",
      "commonName": "",
      "accessDirection": "inbound"
    }
  ],
}
```

```
        {
            "fromPort": 443,
            "toPort": 443,
            "protocol": "tcp",
            "accessFrom": "Anywhere (0.0.0.0/0)",
            "accessType": "public",
            "commonName": "",
            "accessDirection": "inbound"
        }
    ],
    "state": {
        "code": 16,
        "name": "running"
    },
    "username": "bitnami",
    "sshKeyName": "MyKey"
}
}
```

- For API details, see [GetInstance](#) in *AWS CLI Command Reference*.

get-instances

The following code example shows how to use `get-instances`.

AWS CLI

To get information about all instances

The following `get-instances` example displays details about all of the instances in the configured AWS Region.

```
aws lightsail get-instances
```

Output:

```
{
  "instances": [
    {
      "name": "Windows_Server_2016-1",
```



```
    "arn": "arn:aws:lightsail:us-
west-2:111122223333:Instance/0f44fbb9-8f55-4e47-a25e-EXAMPLE04763",
    "supportCode": "62EXAMPLE362/i-0bEXAMPLE71a686b9",
    "createdAt": 1571332358.665,
    "location": {
      "availabilityZone": "us-west-2a",
      "regionName": "us-west-2"
    },
    "resourceType": "Instance",
    "tags": [],
    "blueprintId": "windows_server_2016",
    "blueprintName": "Windows Server 2016",
    "bundleId": "small_win_2_0",
    "isStaticIp": false,
    "privateIpAddress": "192.0.2.0",
    "publicIpAddress": "192.0.2.0",
    "hardware": {
      "cpuCount": 1,
      "disks": [
        {
          "createdAt": 1571332358.665,
          "sizeInGb": 60,
          "isSystemDisk": true,
          "iops": 180,
          "path": "/dev/sda1",
          "attachedTo": "Windows_Server_2016-1",
          "attachmentState": "attached"
        },
        {
          "name": "my-disk-for-windows-server",
          "arn": "arn:aws:lightsail:us-
west-2:111122223333:Disk/4123a81c-484c-49ea-afea-5EXAMPLEda87",
          "supportCode": "6EXAMPLE3362/vol-0EXAMPLEb2b99ca3d",
          "createdAt": 1571355063.494,
          "location": {
            "availabilityZone": "us-west-2a",
            "regionName": "us-west-2"
          },
          "resourceType": "Disk",
          "tags": [],
          "sizeInGb": 128,
          "isSystemDisk": false,
          "iops": 384,
          "path": "/dev/xvdf",
```

```
        "state": "in-use",
        "attachedTo": "Windows_Server_2016-1",
        "isAttached": true,
        "attachmentState": "attached"
    }
],
"ramSizeInGb": 2.0
},
"networking": {
    "monthlyTransfer": {
        "gbPerMonthAllocated": 3072
    },
    "ports": [
        {
            "fromPort": 80,
            "toPort": 80,
            "protocol": "tcp",
            "accessFrom": "Anywhere (0.0.0.0/0)",
            "accessType": "public",
            "commonName": "",
            "accessDirection": "inbound"
        },
        {
            "fromPort": 22,
            "toPort": 22,
            "protocol": "tcp",
            "accessFrom": "Anywhere (0.0.0.0/0)",
            "accessType": "public",
            "commonName": "",
            "accessDirection": "inbound"
        },
        {
            "fromPort": 3389,
            "toPort": 3389,
            "protocol": "tcp",
            "accessFrom": "Anywhere (0.0.0.0/0)",
            "accessType": "public",
            "commonName": "",
            "accessDirection": "inbound"
        }
    ]
},
"state": {
    "code": 16,
```

```
        "name": "running"
    },
    "username": "Administrator",
    "sshKeyName": "LightsailDefaultKeyPair"
  },
  {
    "name": "MEAN-1",
    "arn": "arn:aws:lightsail:us-west-2:111122223333:Instance/bd470fc5-
a68b-44c5-8dbc-8EXAMPLEbada",
    "supportCode": "6EXAMPLE3362/i-0EXAMPLEa407c97d3",
    "createdAt": 1570635023.124,
    "location": {
      "availabilityZone": "us-west-2a",
      "regionName": "us-west-2"
    },
    "resourceType": "Instance",
    "tags": [],
    "blueprintId": "mean_4_0_9",
    "blueprintName": "MEAN",
    "bundleId": "medium_2_0",
    "isStaticIp": false,
    "privateIpAddress": "192.0.2.0",
    "publicIpAddress": "192.0.2.0",
    "hardware": {
      "cpuCount": 2,
      "disks": [
        {
          "name": "Disk-1",
          "arn": "arn:aws:lightsail:us-west-2:111122223333:Disk/
c21cfb0a-07f2-44ae-9a23-bEXAMPLE8096",
          "supportCode": "6EXAMPLE3362/vol-0EXAMPLEf2f88b32f",
          "createdAt": 1566585439.587,
          "location": {
            "availabilityZone": "us-west-2a",
            "regionName": "us-west-2"
          },
          "resourceType": "Disk",
          "tags": [
            {
              "key": "test"
            }
          ],
          "sizeInGb": 8,
          "isSystemDisk": false,
```

```
        "iops": 100,
        "path": "/dev/xvdf",
        "state": "in-use",
        "attachedTo": "MEAN-1",
        "isAttached": true,
        "attachmentState": "attached"
    },
    {
        "createdAt": 1570635023.124,
        "sizeInGb": 80,
        "isSystemDisk": true,
        "iops": 240,
        "path": "/dev/sda1",
        "attachedTo": "MEAN-1",
        "attachmentState": "attached"
    }
],
"ramSizeInGb": 4.0
},
"networking": {
    "monthlyTransfer": {
        "gbPerMonthAllocated": 4096
    },
    "ports": [
        {
            "fromPort": 80,
            "toPort": 80,
            "protocol": "tcp",
            "accessFrom": "Anywhere (0.0.0.0/0)",
            "accessType": "public",
            "commonName": "",
            "accessDirection": "inbound"
        },
        {
            "fromPort": 22,
            "toPort": 22,
            "protocol": "tcp",
            "accessFrom": "Anywhere (0.0.0.0/0)",
            "accessType": "public",
            "commonName": "",
            "accessDirection": "inbound"
        },
        {
            "fromPort": 443,
```

```

        "toPort": 443,
        "protocol": "tcp",
        "accessFrom": "Anywhere (0.0.0.0/0)",
        "accessType": "public",
        "commonName": "",
        "accessDirection": "inbound"
      }
    ]
  },
  "state": {
    "code": 16,
    "name": "running"
  },
  "username": "bitnami",
  "sshKeyName": "MyTestKey"
}
]
}

```

- For API details, see [GetInstances](#) in *AWS CLI Command Reference*.

get-key-pair

The following code example shows how to use `get-key-pair`.

AWS CLI

To get information about a key pair

The following `get-key-pair` example displays details about the specified key pair.

```
aws lightsail get-key-pair \
  --key-pair-name MyKey1
```

Output:

```
{
  "keyPair": {
    "name": "MyKey1",
    "arn": "arn:aws:lightsail:us-
west-2:111122223333:KeyPair/19a4efdf-3054-43d6-91fd-eEXAMPLE21bf",
    "supportCode": "6EXAMPLE3362/MyKey1",
  }
}
```

```
    "createdAt": 1571255026.975,
    "location": {
      "availabilityZone": "all",
      "regionName": "us-west-2"
    },
    "resourceType": "KeyPair",
    "tags": [],
    "fingerprint": "00:11:22:33:44:55:66:77:88:99:aa:bb:cc:dd:ee:ff:gg:hh:ii:jj"
  }
}
```

- For API details, see [GetKeyPair](#) in *AWS CLI Command Reference*.

get-key-pairs

The following code example shows how to use `get-key-pairs`.

AWS CLI

To get information about all key pairs

The following `get-key-pairs` example displays details about all of the key pairs in the configured AWS Region.

```
aws lightsail get-key-pairs
```

Output:

```
{
  "keyPairs": [
    {
      "name": "MyKey1",
      "arn": "arn:aws:lightsail:us-west-2:111122223333:KeyPair/19a4efdf-3054-43d6-91fd-eEXAMPLE21bf",
      "supportCode": "6EXAMPLE3362/MyKey1",
      "createdAt": 1571255026.975,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      },
      "resourceType": "KeyPair",
    }
  ]
}
```

```

        "tags": [],
        "fingerprint":
"00:11:22:33:44:55:66:77:88:99:aa:bb:cc:dd:ee:ff:gg:hh:ii:jj"
    }
]
}

```

- For API details, see [GetKeyPairs](#) in *AWS CLI Command Reference*.

get-load-balancer-tls-certificates

The following code example shows how to use `get-load-balancer-tls-certificates`.

AWS CLI

To get information about TLS certificates for a load balancer

The following `get-load-balancer-tls-certificates` example displays details about the TLS certificates for the specified load balancer.

```
aws lightsail get-load-balancer-tls-certificates \
  --load-balancer-name LoadBalancer-1
```

Output:

```

{
  "tlsCertificates": [
    {
      "name": "example-com",
      "arn": "arn:aws:lightsail:us-
west-2:111122223333:LoadBalancerTlsCertificate/d7bf4643-6a02-4cd4-b3c4-
fEXAMPLE9b4d",
      "supportCode": "6EXAMPLE3362/arn:aws:acm:us-
west-2:333322221111:certificate/9af8e32c-a54e-4a67-8c63-cEXAMPLEb314",
      "createdAt": 1571678025.3,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      },
      "resourceType": "LoadBalancerTlsCertificate",
      "loadBalancerName": "LoadBalancer-1",
    }
  ]
}

```

```

    "isAttached": false,
    "status": "ISSUED",
    "domainName": "example.com",
    "domainValidationRecords": [
      {
        "name": "_dEXAMPLE4ede046a0319eb44a4eb3cbc.example.com.",
        "type": "CNAME",
        "value": "_bEXAMPLE0899fb7b6bf79d9741d1a383.hkvuiqjoua.acm-
validations.aws.",
        "validationStatus": "SUCCESS",
        "domainName": "example.com"
      }
    ],
    "issuedAt": 1571678070.0,
    "issuer": "Amazon",
    "keyAlgorithm": "RSA-2048",
    "notAfter": 1605960000.0,
    "notBefore": 1571616000.0,
    "serial": "00:11:22:33:44:55:66:77:88:99:aa:bb:cc:dd:ee:ff",
    "signatureAlgorithm": "SHA256WITHRSA",
    "subject": "CN=example.com",
    "subjectAlternativeNames": [
      "example.com"
    ]
  }
]
}

```

- For API details, see [GetLoadBalancerTlsCertificates](#) in *AWS CLI Command Reference*.

get-load-balancer

The following code example shows how to use `get-load-balancer`.

AWS CLI

To get information about a load balancer

The following `get-load-balancer` example displays details about the specified load balancer.

```

aws lightsail get-load-balancer \
  --load-balancer-name LoadBalancer-1

```


Output:

```
{
  "loadBalancer": {
    "name": "LoadBalancer-1",
    "arn": "arn:aws:lightsail:us-west-2:111122223333:LoadBalancer/40486b2b-1ad0-4152-83e4-cEXAMPLE6f4b",
    "supportCode": "6EXAMPLE3362/arn:aws:elasticloadbalancing:us-west-2:333322221111:loadbalancer/app/bEXAMPLE128cb59d86f946a9395dd304/1EXAMPLE8dd9d77e",
    "createdAt": 1571677906.723,
    "location": {
      "availabilityZone": "all",
      "regionName": "us-west-2"
    },
    "resourceType": "LoadBalancer",
    "tags": [],
    "dnsName": "bEXAMPLE128cb59d86f946a9395dd304-1486911371.us-west-2.elb.amazonaws.com",
    "state": "active",
    "protocol": "HTTP",
    "publicPorts": [
      80
    ],
    "healthCheckPath": "/",
    "instancePort": 80,
    "instanceHealthSummary": [
      {
        "instanceName": "MEAN-3",
        "instanceHealth": "healthy"
      },
      {
        "instanceName": "MEAN-1",
        "instanceHealth": "healthy"
      },
      {
        "instanceName": "MEAN-2",
        "instanceHealth": "healthy"
      }
    ],
    "tlsCertificateSummaries": [
      {
        "name": "example-com",
        "isAttached": false
      }
    ]
  }
}
```

```
    }
  ],
  "configurationOptions": {
    "SessionStickinessEnabled": "false",
    "SessionStickiness_LB_CookieDurationSeconds": "86400"
  }
}
```

- For API details, see [GetLoadBalancer](#) in *AWS CLI Command Reference*.

get-load-balancers

The following code example shows how to use `get-load-balancers`.

AWS CLI

To get information about all load balancers

The following `get-load-balancers` example displays details about all of the load balancers in the configured AWS Region.

```
aws lightsail get-load-balancers
```

Output:

```
{
  "loadBalancers": [
    {
      "name": "LoadBalancer-1",
      "arn": "arn:aws:lightsail:us-west-2:111122223333:LoadBalancer/40486b2b-1ad0-4152-83e4-cEXAMPLE6f4b",
      "supportCode": "6EXAMPLE3362/arn:aws:elasticloadbalancing:us-west-2:333322221111:loadbalancer/app/bEXAMPLE128cb59d86f946a9395dd304/1EXAMPLE8dd9d77e",
      "createdAt": 1571677906.723,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      },
      "resourceType": "LoadBalancer",
    }
  ]
}
```

```
    "tags": [],
    "dnsName": "bEXAMPLE128cb59d86f946a9395dd304-1486911371.us-
west-2.elb.amazonaws.com",
    "state": "active",
    "protocol": "HTTP",
    "publicPorts": [
      80
    ],
    "healthCheckPath": "/",
    "instancePort": 80,
    "instanceHealthSummary": [
      {
        "instanceName": "MEAN-3",
        "instanceHealth": "healthy"
      },
      {
        "instanceName": "MEAN-1",
        "instanceHealth": "healthy"
      },
      {
        "instanceName": "MEAN-2",
        "instanceHealth": "healthy"
      }
    ],
    "tlsCertificateSummaries": [
      {
        "name": "example-com",
        "isAttached": false
      }
    ],
    "configurationOptions": {
      "SessionStickinessEnabled": "false",
      "SessionStickiness_LB_CookieDurationSeconds": "86400"
    }
  }
]
```

- For API details, see [GetLoadBalancers](#) in *AWS CLI Command Reference*.

get-operation

The following code example shows how to use get-operation.

AWS CLI

To get information about a single operation

The following `get-operation` example displays details about the specified operation.

```
aws lightsail get-operation \  
  --operation-id e5700e8a-daf2-4b49-bc01-3EXAMPLE910a
```

Output:

```
{  
  "operation": {  
    "id": "e5700e8a-daf2-4b49-bc01-3EXAMPLE910a",  
    "resourceName": "Instance-1",  
    "resourceType": "Instance",  
    "createdAt": 1571679872.404,  
    "location": {  
      "availabilityZone": "us-west-2a",  
      "regionName": "us-west-2"  
    },  
    "isTerminal": true,  
    "operationType": "CreateInstance",  
    "status": "Succeeded",  
    "statusChangedAt": 1571679890.304  
  }  
}
```

- For API details, see [GetOperation](#) in *AWS CLI Command Reference*.

get-operations-for-resource

The following code example shows how to use `get-operations-for-resource`.

AWS CLI

To get all operations for a resource

The following `get-operations-for-resource` example displays details about all operations for the specified resource.

```
aws lightsail get-operations-for-resource \  
  --resource-id e5700e8a-daf2-4b49-bc01-3EXAMPLE910a
```

```
--resource-name LoadBalancer-1
```

Output:

```
{
  "operations": [
    {
      "id": "e2973046-43f8-4252-a4b4-9EXAMPLE69ce",
      "resourceName": "LoadBalancer-1",
      "resourceType": "LoadBalancer",
      "createdAt": 1571678786.071,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      },
      "isTerminal": true,
      "operationDetails": "MEAN-1",
      "operationType": "DetachInstancesFromLoadBalancer",
      "status": "Succeeded",
      "statusChangedAt": 1571679087.57
    },
    {
      "id": "2d742a18-0e7f-48c8-9705-3EXAMPLEf98a",
      "resourceName": "LoadBalancer-1",
      "resourceType": "LoadBalancer",
      "createdAt": 1571678782.784,
      "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
      },
      "isTerminal": true,
      "operationDetails": "MEAN-1",
      "operationType": "AttachInstancesToLoadBalancer",
      "status": "Succeeded",
      "statusChangedAt": 1571678798.465
    },
    {
      "id": "6c700fcc-4246-40ab-952b-1EXAMPLEdac2",
      "resourceName": "LoadBalancer-1",
      "resourceType": "LoadBalancer",
      "createdAt": 1571678775.297,
      "location": {
        "availabilityZone": "all",

```

```

        "regionName": "us-west-2"
      },
      "isTerminal": true,
      "operationDetails": "MEAN-3",
      "operationType": "AttachInstancesToLoadBalancer",
      "status": "Succeeded",
      "statusChangedAt": 1571678842.806
    },
    ...
  ]
}

```

- For API details, see [GetOperationsForResource](#) in *AWS CLI Command Reference*.

get-operations

The following code example shows how to use get-operations.

AWS CLI

To get information about all operations

The following get-operations example displays details about all of the operations in the configured AWS Region.

```
aws lightsail get-operations
```

Output:

```

{
  "operations": [
    {
      "id": "e5700e8a-daf2-4b49-bc01-3EXAMPLE910a",
      "resourceName": "Instance-1",
      "resourceType": "Instance",
      "createdAt": 1571679872.404,
      "location": {
        "availabilityZone": "us-west-2a",
        "regionName": "us-west-2"
      }
    },

```

```
    "isTerminal": true,
    "operationType": "CreateInstance",
    "status": "Succeeded",
    "statusChangedAt": 1571679890.304
  },
  {
    "id": "701a3339-930e-4914-a9f9-7EXAMPLE68d7",
    "resourceName": "WordPress-1",
    "resourceType": "Instance",
    "createdAt": 1571678786.072,
    "location": {
      "availabilityZone": "all",
      "regionName": "us-west-2"
    },
    "isTerminal": true,
    "operationDetails": "LoadBalancer-1",
    "operationType": "DetachInstancesFromLoadBalancer",
    "status": "Succeeded",
    "statusChangedAt": 1571679086.399
  },
  {
    "id": "e2973046-43f8-4252-a4b4-9EXAMPLE69ce",
    "resourceName": "LoadBalancer-1",
    "resourceType": "LoadBalancer",
    "createdAt": 1571678786.071,
    "location": {
      "availabilityZone": "all",
      "regionName": "us-west-2"
    },
    "isTerminal": true,
    "operationDetails": "WordPress-1",
    "operationType": "DetachInstancesFromLoadBalancer",
    "status": "Succeeded",
    "statusChangedAt": 1571679087.57
  },
  ...
}
]
```

- For API details, see [GetOperations](#) in *AWS CLI Command Reference*.

get-regions

The following code example shows how to use `get-regions`.

AWS CLI

To get all AWS Regions for Amazon Lightsail

The following `get-regions` example displays details about all of the AWS Regions for Amazon Lightsail.

```
aws lightsail get-regions
```

Output:

```
{
  "regions": [
    {
      "continentCode": "NA",
      "description": "This region is recommended to serve users in the eastern United States",
      "displayName": "Virginia",
      "name": "us-east-1",
      "availabilityZones": [],
      "relationalDatabaseAvailabilityZones": []
    },
    {
      "continentCode": "NA",
      "description": "This region is recommended to serve users in the eastern United States",
      "displayName": "Ohio",
      "name": "us-east-2",
      "availabilityZones": [],
      "relationalDatabaseAvailabilityZones": []
    },
    {
      "continentCode": "NA",
      "description": "This region is recommended to serve users in the northwestern United States, Alaska, and western Canada",
      "displayName": "Oregon",
      "name": "us-west-2",
      "availabilityZones": [],
      "relationalDatabaseAvailabilityZones": []
    }
  ]
}
```



```
    },  
    ...  
  }  
]  
}
```

- For API details, see [GetRegions](#) in *AWS CLI Command Reference*.

get-relational-database-blueprints

The following code example shows how to use `get-relational-database-blueprints`.

AWS CLI

To get the blueprints for new relational databases

The following `get-relational-database-blueprints` example displays details about all of the available relational database blueprints that can be used to create new relational databases in Amazon Lightsail.

```
aws lightsail get-relational-database-blueprints
```

Output:

```
{  
  "blueprints": [  
    {  
      "blueprintId": "mysql_5_6",  
      "engine": "mysql",  
      "engineVersion": "5.6.44",  
      "engineDescription": "MySQL Community Edition",  
      "engineVersionDescription": "MySQL 5.6.44",  
      "isEngineDefault": false  
    },  
    {  
      "blueprintId": "mysql_5_7",  
      "engine": "mysql",  
      "engineVersion": "5.7.26",  
      "engineDescription": "MySQL Community Edition",  
      "engineVersionDescription": "MySQL 5.7.26",  
      "isEngineDefault": true  
    }  
  ]  
}
```

```
    },
    {
      "blueprintId": "mysql_8_0",
      "engine": "mysql",
      "engineVersion": "8.0.16",
      "engineDescription": "MySQL Community Edition",
      "engineVersionDescription": "MySQL 8.0.16",
      "isEngineDefault": false
    },
    {
      "blueprintId": "postgres_9_6",
      "engine": "postgres",
      "engineVersion": "9.6.15",
      "engineDescription": "PostgreSQL",
      "engineVersionDescription": "PostgreSQL 9.6.15-R1",
      "isEngineDefault": false
    },
    {
      "blueprintId": "postgres_10",
      "engine": "postgres",
      "engineVersion": "10.10",
      "engineDescription": "PostgreSQL",
      "engineVersionDescription": "PostgreSQL 10.10-R1",
      "isEngineDefault": false
    },
    {
      "blueprintId": "postgres_11",
      "engine": "postgres",
      "engineVersion": "11.5",
      "engineDescription": "PostgreSQL",
      "engineVersionDescription": "PostgreSQL 11.5-R1",
      "isEngineDefault": true
    }
  ]
}
```

- For API details, see [GetRelationalDatabaseBlueprints](#) in *AWS CLI Command Reference*.

get-relational-database-bundles

The following code example shows how to use `get-relational-database-bundles`.

AWS CLI

To get the bundles for new relational databases

The following `get-relational-database-bundles` example displays details about all of the available relational database bundles that can be used to create new relational databases in Amazon Lightsail. Note that the response does not include inactive bundles because the `--include-inactive` flag is not specified in the command. You cannot use inactive bundles to create new relational databases.

```
aws lightsail get-relational-database-bundles
```

Output:

```
{
  "bundles": [
    {
      "bundleId": "micro_2_0",
      "name": "Micro",
      "price": 15.0,
      "ramSizeInGb": 1.0,
      "diskSizeInGb": 40,
      "transferPerMonthInGb": 100,
      "cpuCount": 2,
      "isEncrypted": true,
      "isActive": true
    },
    {
      "bundleId": "micro_ha_2_0",
      "name": "Micro with High Availability",
      "price": 30.0,
      "ramSizeInGb": 1.0,
      "diskSizeInGb": 40,
      "transferPerMonthInGb": 100,
      "cpuCount": 2,
      "isEncrypted": true,
      "isActive": true
    },
    {
      "bundleId": "small_2_0",
      "name": "Small",
      "price": 30.0,
      "ramSizeInGb": 2.0,
```

```
    "diskSizeInGb": 80,
    "transferPerMonthInGb": 100,
    "cpuCount": 2,
    "isEncrypted": true,
    "isActive": true
  },
  {
    "bundleId": "small_ha_2_0",
    "name": "Small with High Availability",
    "price": 60.0,
    "ramSizeInGb": 2.0,
    "diskSizeInGb": 80,
    "transferPerMonthInGb": 100,
    "cpuCount": 2,
    "isEncrypted": true,
    "isActive": true
  },
  {
    "bundleId": "medium_2_0",
    "name": "Medium",
    "price": 60.0,
    "ramSizeInGb": 4.0,
    "diskSizeInGb": 120,
    "transferPerMonthInGb": 100,
    "cpuCount": 2,
    "isEncrypted": true,
    "isActive": true
  },
  {
    "bundleId": "medium_ha_2_0",
    "name": "Medium with High Availability",
    "price": 120.0,
    "ramSizeInGb": 4.0,
    "diskSizeInGb": 120,
    "transferPerMonthInGb": 100,
    "cpuCount": 2,
    "isEncrypted": true,
    "isActive": true
  },
  {
    "bundleId": "large_2_0",
    "name": "Large",
    "price": 115.0,
    "ramSizeInGb": 8.0,
```

```

        "diskSizeInGb": 240,
        "transferPerMonthInGb": 200,
        "cpuCount": 2,
        "isEncrypted": true,
        "isActive": true
    },
    {
        "bundleId": "large_ha_2_0",
        "name": "Large with High Availability",
        "price": 230.0,
        "ramSizeInGb": 8.0,
        "diskSizeInGb": 240,
        "transferPerMonthInGb": 200,
        "cpuCount": 2,
        "isEncrypted": true,
        "isActive": true
    }
]
}

```

For more information, see [Creating a database in Amazon Lightsail](#) in the *Amazon Lightsail Developer Guide*.

- For API details, see [GetRelationalDatabaseBundles](#) in *AWS CLI Command Reference*.

get-relational-database-events

The following code example shows how to use `get-relational-database-events`.

AWS CLI

To get the events for a relational database

The following `get-relational-database-events` example displays details about events in the last 17 hours (1020 minutes) for the specified relational database.

```

aws lightsail get-relational-database-events \
  --relational-database-name Database-1 \
  --duration-in-minutes 1020

```

Output:

```
{
```

```
"relationalDatabaseEvents": [  
  {  
    "resource": "Database-1",  
    "createdAt": 1571654146.553,  
    "message": "Backing up Relational Database",  
    "eventCategories": [  
      "backup"  
    ]  
  },  
  {  
    "resource": "Database-1",  
    "createdAt": 1571654249.98,  
    "message": "Finished Relational Database backup",  
    "eventCategories": [  
      "backup"  
    ]  
  }  
]
```

- For API details, see [GetRelationalDatabaseEvents](#) in *AWS CLI Command Reference*.

get-relational-database-log-events

The following code example shows how to use `get-relational-database-log-events`.

AWS CLI

To get log events for a relational database

The following `get-relational-database-log-events` example displays details about the specified log between 1570733176 and 1571597176 for relational database Database1. The information returned is configured to start from head.

We recommend that you use a unix time converter to identify the start and end times.

```
aws lightsail get-relational-database-log-events \  
  --relational-database-name Database1 \  
  --log-stream-name error \  
  --start-from-head \  
  --start-time 1570733176 \  
  --end-time 1571597176
```

Output:

```
{
  "resourceLogEvents": [
    {
      "createdAt": 1570820267.0,
      "message": "2019-10-11 18:57:47 20969 [Warning] IP address '192.0.2.0'
could not be resolved: Name or service not known"
    },
    {
      "createdAt": 1570860974.0,
      "message": "2019-10-12 06:16:14 20969 [Warning] IP address '8192.0.2.0'
could not be resolved: Temporary failure in name resolution"
    },
    {
      "createdAt": 1570860977.0,
      "message": "2019-10-12 06:16:17 20969 [Warning] IP address '192.0.2.0'
could not be resolved: Temporary failure in name resolution"
    },
    {
      "createdAt": 1570860979.0,
      "message": "2019-10-12 06:16:19 20969 [Warning] IP address '192.0.2.0'
could not be resolved: Temporary failure in name resolution"
    },
    {
      "createdAt": 1570860981.0,
      "message": "2019-10-12 06:16:21 20969 [Warning] IP address '192.0.2.0'
could not be resolved: Temporary failure in name resolution"
    },
    {
      "createdAt": 1570860982.0,
      "message": "2019-10-12 06:16:22 20969 [Warning] IP address '192.0.2.0'
could not be resolved: Temporary failure in name resolution"
    },
    {
      "createdAt": 1570860984.0,
      "message": "2019-10-12 06:16:24 20969 [Warning] IP address '192.0.2.0'
could not be resolved: Temporary failure in name resolution"
    },
    {
      "createdAt": 1570860986.0,
      "message": "2019-10-12 06:16:26 20969 [Warning] IP address '192.0.2.0'
could not be resolved: Temporary failure in name resolution"
    },
  ],
}
```

```

    ...
  }
],
"nextBackwardToken":
"eEXAMPLEZXJUZXh0IjoiZnRwb3F3cUpRS1Q5NndMYThxe1RUZ1FhR3JGc2dKWEEvM2kvajZMZzVWVWpqRDN0YjFXTj
"nextForwardToken":
"eEXAMPLEZXJUZXh0IjoiT09Lb0Z6ZFRJbHhaNEQ5N2tPbkkwRmwwNUxPZjFTbFFwUk1Qbz1SaWgvMwVXbEk4aG56VH
}

```

- For API details, see [GetRelationalDatabaseLogEvents](#) in *AWS CLI Command Reference*.

get-relational-database-log-streams

The following code example shows how to use `get-relational-database-log-streams`.

AWS CLI

To get the log streams for a relational database

The following `get-relational-database-log-streams` example returns all of the available log streams for the specified relational database.

```
aws lightsail get-relational-database-log-streams \
--relational-database-name Database1
```

Output:

```
{
  "logStreams": [
    "audit",
    "error",
    "general",
    "slowquery"
  ]
}
```

- For API details, see [GetRelationalDatabaseLogStreams](#) in *AWS CLI Command Reference*.

get-relational-database-master-user-password

The following code example shows how to use `get-relational-database-master-user-password`.

AWS CLI

To get the master user password for a relational database

The following `get-relational-database-master-user-password` example returns information about the master user password for the specified relational database.

```
aws lightsail get-relational-database-master-user-password \  
  --relational-database-name Database-1
```

Output:

```
{  
  "masterUserPassword": "VEXAMPLEec.9qvx,_t<)Wkf)kwboM,>2",  
  "createdAt": 1571259453.959  
}
```

- For API details, see [GetRelationalDatabaseMasterUserPassword](#) in *AWS CLI Command Reference*.

get-relational-database-metric-data

The following code example shows how to use `get-relational-database-metric-data`.

AWS CLI

To get metric data for a relational database

The following `get-relational-database-metric-data` example returns the count sum of the metric `DatabaseConnections` over the period of 24 hours (86400 seconds) between 1570733176 and 1571597176 for relational database `Database1`.

We recommend that you use a unix time converter to identify the start and end times.

```
aws lightsail get-relational-database-metric-data \  
  --relational-database-name Database1 \  
  --start-time 1570733176 \  
  --end-time 1571597176
```

```
--metric-name DatabaseConnections \  
--period 86400 \  
--start-time 1570733176 \  
--end-time 1571597176 \  
--unit Count \  
--statistics Sum
```

Output:

```
{  
  "metricName": "DatabaseConnections",  
  "metricData": [  
    {  
      "sum": 1.0,  
      "timestamp": 1571510760.0,  
      "unit": "Count"  
    },  
    {  
      "sum": 1.0,  
      "timestamp": 1570733160.0,  
      "unit": "Count"  
    },  
    {  
      "sum": 1.0,  
      "timestamp": 1570992360.0,  
      "unit": "Count"  
    },  
    {  
      "sum": 0.0,  
      "timestamp": 1571251560.0,  
      "unit": "Count"  
    },  
    {  
      "sum": 721.0,  
      "timestamp": 1570819560.0,  
      "unit": "Count"  
    },  
    {  
      "sum": 1.0,  
      "timestamp": 1571078760.0,  
      "unit": "Count"  
    },  
    {
```

```
    "sum": 2.0,  
    "timestamp": 1571337960.0,  
    "unit": "Count"  
  },  
  {  
    "sum": 684.0,  
    "timestamp": 1570905960.0,  
    "unit": "Count"  
  },  
  {  
    "sum": 0.0,  
    "timestamp": 1571165160.0,  
    "unit": "Count"  
  },  
  {  
    "sum": 1.0,  
    "timestamp": 1571424360.0,  
    "unit": "Count"  
  }  
]  
}
```

- For API details, see [GetRelationalDatabaseMetricData](#) in *AWS CLI Command Reference*.

get-relational-database-parameters

The following code example shows how to use `get-relational-database-parameters`.

AWS CLI

To get parameters for a relational database

The following `get-relational-database-parameters` example returns information about all of the available parameters for the specified relational database.

```
aws lightsail get-relational-database-parameters \  
  --relational-database-name Database-1
```

Output:

```
{  
  "parameters": [  
    {  
      "name": "db-instance-class",  
      "value": "db.m3.xlarge",  
      "unit": "Count",  
      "timestamp": 1571424360.0,  
      "sum": 1.0,  
      "unit": "Count"  
    },  
    {  
      "name": "db-instance-class",  
      "value": "db.m3.xlarge",  
      "unit": "Count",  
      "timestamp": 1571165160.0,  
      "sum": 0.0,  
      "unit": "Count"  
    },  
    {  
      "name": "db-instance-class",  
      "value": "db.m3.xlarge",  
      "unit": "Count",  
      "timestamp": 1570905960.0,  
      "sum": 684.0,  
      "unit": "Count"  
    },  
    {  
      "name": "db-instance-class",  
      "value": "db.m3.xlarge",  
      "unit": "Count",  
      "timestamp": 1571337960.0,  
      "sum": 2.0,  
      "unit": "Count"  
    }  
  ]  
}
```

```
{
  "allowedValues": "0,1",
  "applyMethod": "pending-reboot",
  "applyType": "dynamic",
  "dataType": "boolean",
  "description": "Automatically set all granted roles as active after the
user has authenticated successfully.",
  "isModifiable": true,
  "parameterName": "activate_all_roles_on_login",
  "parameterValue": "0"
},
{
  "allowedValues": "0,1",
  "applyMethod": "pending-reboot",
  "applyType": "static",
  "dataType": "boolean",
  "description": "Controls whether user-defined functions that have only
an xxx symbol for the main function can be loaded",
  "isModifiable": false,
  "parameterName": "allow-suspicious-udfs"
},
{
  "allowedValues": "0,1",
  "applyMethod": "pending-reboot",
  "applyType": "dynamic",
  "dataType": "boolean",
  "description": "Sets the autocommit mode",
  "isModifiable": true,
  "parameterName": "autocommit"
},
{
  "allowedValues": "0,1",
  "applyMethod": "pending-reboot",
  "applyType": "static",
  "dataType": "boolean",
  "description": "Controls whether the server autogenerates SSL key and
certificate files in the data directory, if they do not already exist.",
  "isModifiable": false,
  "parameterName": "auto_generate_certs"
},
...
}
]
```

```
}
```

For more information, see [Updating database parameters in Amazon Lightsail](#) in the *Lightsail Dev Guide*.

- For API details, see [GetRelationalDatabaseParameters](#) in *AWS CLI Command Reference*.

get-relational-database-snapshot

The following code example shows how to use `get-relational-database-snapshot`.

AWS CLI

To get information about a relational database snapshot

The following `get-relational-database-snapshot` example displays details about the specified relational database snapshot.

```
aws lightsail get-relational-database-snapshot \  
  --relational-database-snapshot-name Database-1-1571350042
```

Output:

```
{  
  "relationalDatabaseSnapshot": {  
    "name": "Database-1-1571350042",  
    "arn": "arn:aws:lightsail:us-west-2:111122223333:RelationalDatabaseSnapshot/0389bbad-4b85-4c3d-9EXAMPLEaee3643d2",  
    "supportCode": "6EXAMPLE3362/1s-8EXAMPLE2ba7ad041451946fafc2ad19cfbd9eb2",  
    "createdAt": 1571350046.238,  
    "location": {  
      "availabilityZone": "all",  
      "regionName": "us-west-2"  
    },  
    "resourceType": "RelationalDatabaseSnapshot",  
    "tags": [],  
    "engine": "mysql",  
    "engineVersion": "8.0.16",  
    "sizeInGb": 40,  
    "state": "available",  
    "fromRelationalDatabaseName": "Database-1",
```

```
    "fromRelationalDatabaseArn": "arn:aws:lightsail:us-  
west-2:111122223333:RelationalDatabase/7ea932b1-b85a-4bd5-9b3e-bEXAMPLE8cc4",  
    "fromRelationalDatabaseBundleId": "micro_1_0",  
    "fromRelationalDatabaseBlueprintId": "mysql_8_0"  
  }  
}
```

- For API details, see [GetRelationalDatabaseSnapshot](#) in *AWS CLI Command Reference*.

get-relational-database-snapshots

The following code example shows how to use `get-relational-database-snapshots`.

AWS CLI

To get information about all relational database snapshots

The following `get-relational-database-snapshots` example displays details about all of the relational database snapshots in the configured AWS Region.

```
aws lightsail get-relational-database-snapshots
```

Output:

```
{  
  "relationalDatabaseSnapshots": [  
    {  
      "name": "Database-1-1571350042",  
      "arn": "arn:aws:lightsail:us-  
west-2:111122223333:RelationalDatabaseSnapshot/0389bbad-4b85-4c3d-9861-6EXAMPLE43d2",  
      "supportCode": "6EXAMPLE3362/  
1s-8EXAMPLE2ba7ad041451946fafc2ad19cfbd9eb2",  
      "createdAt": 1571350046.238,  
      "location": {  
        "availabilityZone": "all",  
        "regionName": "us-west-2"  
      },  
      "resourceType": "RelationalDatabaseSnapshot",  
      "tags": [],  
      "engine": "mysql",  
      "engineVersion": "8.0.16",  
      "sizeInGb": 40,  
    }  
  ]  
}
```

```

    "state": "available",
    "fromRelationalDatabaseName": "Database-1",
    "fromRelationalDatabaseArn": "arn:aws:lightsail:us-
west-2:111122223333:RelationalDatabase/7ea932b1-b85a-4bd5-9b3e-bEXAMPLE8cc4",
    "fromRelationalDatabaseBundleId": "micro_1_0",
    "fromRelationalDatabaseBlueprintId": "mysql_8_0"
  },
  {
    "name": "Database1-Console",
    "arn": "arn:aws:lightsail:us-
west-2:111122223333:RelationalDatabaseSnapshot/8b94136e-06ec-4b1a-
a3fb-5EXAMPLEe1e9",
    "supportCode": "6EXAMPLE3362/
1s-9EXAMPLE14b000d34c8d1c432734e137612d5b5c",
    "createdAt": 1571249981.025,
    "location": {
      "availabilityZone": "all",
      "regionName": "us-west-2"
    },
    "resourceType": "RelationalDatabaseSnapshot",
    "tags": [
      {
        "key": "test"
      }
    ],
    "engine": "mysql",
    "engineVersion": "5.6.44",
    "sizeInGb": 40,
    "state": "available",
    "fromRelationalDatabaseName": "Database1",
    "fromRelationalDatabaseArn": "arn:aws:lightsail:us-
west-2:111122223333:RelationalDatabase/a6161cb7-4535-4f16-9dcf-8EXAMPLE3d4e",
    "fromRelationalDatabaseBundleId": "micro_1_0",
    "fromRelationalDatabaseBlueprintId": "mysql_5_6"
  }
]
}

```

- For API details, see [GetRelationalDatabaseSnapshots](#) in *AWS CLI Command Reference*.

get-relational-database

The following code example shows how to use `get-relational-database`.

AWS CLI

To get information about a relational database

The following `get-relational-database` example displays details about the specified relational database.

```
aws lightsail get-relational-database \  
  --relational-database-name Database-1
```

Output:

```
{  
  "relationalDatabase": {  
    "name": "Database-1",  
    "arn": "arn:aws:lightsail:us-  
west-2:111122223333:RelationalDatabase/7ea932b1-b85a-4bd5-9b3e-bEXAMPLE8cc4",  
    "supportCode": "6EXAMPLE3362/1s-9EXAMPLE8ad863723b62cc8901a8aa6e794ae0d2",  
    "createdAt": 1571259453.795,  
    "location": {  
      "availabilityZone": "us-west-2a",  
      "regionName": "us-west-2"  
    },  
    "resourceType": "RelationalDatabase",  
    "tags": [],  
    "relationalDatabaseBlueprintId": "mysql_8_0",  
    "relationalDatabaseBundleId": "micro_1_0",  
    "masterDatabaseName": "dbmaster",  
    "hardware": {  
      "cpuCount": 1,  
      "diskSizeInGb": 40,  
      "ramSizeInGb": 1.0  
    },  
    "state": "available",  
    "backupRetentionEnabled": false,  
    "pendingModifiedValues": {},  
    "engine": "mysql",  
    "engineVersion": "8.0.16",  
    "masterUsername": "dbmasteruser",  
    "parameterApplyStatus": "in-sync",  
    "preferredBackupWindow": "10:01-10:31",  
    "preferredMaintenanceWindow": "sat:11:14-sat:11:44",  
    "publiclyAccessible": true,  
  },  
}
```



```
    "masterEndpoint": {
      "port": 3306,
      "address": "1s-9EXAMPLE8ad863723b62ccEXAMPLEa6e794ae0d2.czowadgeezqi.us-
west-2.rds.amazonaws.com"
    },
    "pendingMaintenanceActions": []
  }
}
```

- For API details, see [GetRelationalDatabase](#) in *AWS CLI Command Reference*.

get-relational-databases

The following code example shows how to use `get-relational-databases`.

AWS CLI

To get information about all relational databases

The following `get-relational-databases` example displays details about all of the relational databases in the configured AWS Region.

```
aws lightsail get-relational-databases
```

Output:

```
{
  "relationalDatabases": [
    {
      "name": "MySQL",
      "arn": "arn:aws:lightsail:us-
west-2:111122223333:RelationalDatabase/8529020c-3ab9-4d51-92af-5EXAMPLE8979",
      "supportCode": "6EXAMPLE3362/
1s-3EXAMPLEa995d8c3b06b4501356e5f2f28e1aeba",
      "createdAt": 1554306019.155,
      "location": {
        "availabilityZone": "us-west-2a",
        "regionName": "us-west-2"
      },
      "resourceType": "RelationalDatabase",
      "tags": [],
      "relationalDatabaseBlueprintId": "mysql_8_0",
    }
  ]
}
```

```

    "relationalDatabaseBundleId": "micro_1_0",
    "masterDatabaseName": "dbmaster",
    "hardware": {
      "cpuCount": 1,
      "diskSizeInGb": 40,
      "ramSizeInGb": 1.0
    },
    "state": "available",
    "backupRetentionEnabled": true,
    "pendingModifiedValues": {},
    "engine": "mysql",
    "engineVersion": "8.0.15",
    "latestRestorableTime": 1571686200.0,
    "masterUsername": "dbmasteruser",
    "parameterApplyStatus": "in-sync",
    "preferredBackupWindow": "07:51-08:21",
    "preferredMaintenanceWindow": "tue:12:18-tue:12:48",
    "publiclyAccessible": true,
    "masterEndpoint": {
      "port": 3306,
      "address":
"ls-3EXAMPLEa995d8c3b06b4501356e5f2fEXAMPLEa.czowadgeezqi.us-
west-2.rds.amazonaws.com"
    },
    "pendingMaintenanceActions": []
  },
  {
    "name": "Postgres",
    "arn": "arn:aws:lightsail:us-west-2:111122223333:RelationalDatabase/
e9780b6b-d0ab-4af2-85f1-1EXAMPLEac68",
    "supportCode": "6EXAMPLE3362/
ls-3EXAMPLEb4ffffb5cec056220c734713e14bd5fcd",
    "createdAt": 1554306000.814,
    "location": {
      "availabilityZone": "us-west-2a",
      "regionName": "us-west-2"
    },
    "resourceType": "RelationalDatabase",
    "tags": [],
    "relationalDatabaseBlueprintId": "postgres_11",
    "relationalDatabaseBundleId": "micro_1_0",
    "masterDatabaseName": "dbmaster",
    "hardware": {
      "cpuCount": 1,

```

```

        "diskSizeInGb": 40,
        "ramSizeInGb": 1.0
    },
    "state": "available",
    "backupRetentionEnabled": true,
    "pendingModifiedValues": {},
    "engine": "postgres",
    "engineVersion": "11.1",
    "latestRestorableTime": 1571686339.0,
    "masterUsername": "dbmasteruser",
    "parameterApplyStatus": "in-sync",
    "preferredBackupWindow": "06:19-06:49",
    "preferredMaintenanceWindow": "sun:10:19-sun:10:49",
    "publiclyAccessible": false,
    "masterEndpoint": {
        "port": 5432,
        "address":
"ls-3EXAMPLEb4ffffb5cec056220c734713eEXAMPLEd.czowadgeezqi.us-
west-2.rds.amazonaws.com"
    },
    "pendingMaintenanceActions": []
}
]
}

```

- For API details, see [GetRelationalDatabases](#) in *AWS CLI Command Reference*.

get-static-ip

The following code example shows how to use `get-static-ip`.

AWS CLI

To get information about a static IP

The following `get-static-ip` example displays details about the specified static IP.

```
aws lightsail get-static-ip \
  --static-ip-name StaticIp-1
```

Output:

```
{
```

```
"staticIp": {
  "name": "StaticIp-1",
  "arn": "arn:aws:lightsail:us-
west-2:111122223333:StaticIp/2257cd76-1f0e-4ac0-82e2-2EXAMPLE23ad",
  "supportCode": "6EXAMPLE3362/192.0.2.0",
  "createdAt": 1571071325.076,
  "location": {
    "availabilityZone": "all",
    "regionName": "us-west-2"
  },
  "resourceType": "StaticIp",
  "ipAddress": "192.0.2.0",
  "isAttached": false
}
```

- For API details, see [GetStaticIp](#) in *AWS CLI Command Reference*.

get-static-ips

The following code example shows how to use `get-static-ips`.

AWS CLI

To get information about all static IPs

The following `get-static-ips` example displays details about all of the static IPs in the configured AWS Region.

```
aws lightsail get-static-ips
```

Output:

```
{
  "staticIps": [
    {
      "name": "StaticIp-1",
      "arn": "arn:aws:lightsail:us-
west-2:111122223333:StaticIp/2257cd76-1f0e-4ac0-8EXAMPLE16f9423ad",
      "supportCode": "6EXAMPLE3362/192.0.2.0",
      "createdAt": 1571071325.076,
      "location": {
```

```

        "availabilityZone": "all",
        "regionName": "us-west-2"
    },
    "resourceType": "StaticIp",
    "ipAddress": "192.0.2.0",
    "isAttached": false
},
{
    "name": "StaticIP-2",
    "arn": "arn:aws:lightsail:us-west-2:111122223333:StaticIp/c61edb40-
e5f0-4fd6-ae7c-8EXAMPLE19f8",
    "supportCode": "6EXAMPLE3362/192.0.2.2",
    "createdAt": 1568305385.681,
    "location": {
        "availabilityZone": "all",
        "regionName": "us-west-2"
    },
    "resourceType": "StaticIp",
    "ipAddress": "192.0.2.2",
    "attachedTo": "WordPress-1",
    "isAttached": true
}
]
}

```

- For API details, see [GetStaticIps](#) in *AWS CLI Command Reference*.

is-vpc-peered

The following code example shows how to use `is-vpc-peered`.

AWS CLI

To identify if your Amazon Lightsail virtual private cloud is peered

The following `is-vpc-peered` example returns the peering status of the Amazon Lightsail virtual private cloud (VPC) for the specified AWS Region.

```
aws lightsail is-vpc-peered \
  --region us-west-2
```

Output:

```
{
  "isPeered": true
}
```

- For API details, see [IsVpcPeered](#) in *AWS CLI Command Reference*.

open-instance-public-ports

The following code example shows how to use `open-instance-public-ports`.

AWS CLI

To open firewall ports for an instance

The following `open-instance-public-ports` example opens TCP port 22 on the specified instance.

```
aws lightsail open-instance-public-ports \
  --instance-name MEAN-2 \
  --port-info fromPort=22,protocol=TCP,toPort=22
```

Output:

```
{
  "operation": {
    "id": "719744f0-a022-46f2-9f11-6EXAMPLE4642",
    "resourceName": "MEAN-2",
    "resourceType": "Instance",
    "createdAt": 1571072906.849,
    "location": {
      "availabilityZone": "us-west-2a",
      "regionName": "us-west-2"
    },
    "isTerminal": true,
    "operationDetails": "22/tcp",
    "operationType": "OpenInstancePublicPorts",
    "status": "Succeeded",
    "statusChangedAt": 1571072906.849
  }
}
```

- For API details, see [OpenInstancePublicPorts](#) in *AWS CLI Command Reference*.

peer-vpc

The following code example shows how to use `peer-vpc`.

AWS CLI

To peer the Amazon Lightsail virtual private cloud

The following `peer-vpc` example peers the Amazon Lightsail virtual private cloud (VPC) for the specified AWS Region.

```
aws lightsail peer-vpc \  
  --region us-west-2
```

Output:

```
{  
  "operation": {  
    "id": "787e846a-54ac-497f-bce2-9EXAMPLE5d91",  
    "resourceName": "vpc-0EXAMPLEa5261efb3",  
    "resourceType": "PeeredVpc",  
    "createdAt": 1571694233.104,  
    "location": {  
      "availabilityZone": "all",  
      "regionName": "us-west-2"  
    },  
    "isTerminal": true,  
    "operationDetails": "vpc-e2b3eb9b",  
    "operationType": "PeeredVpc",  
    "status": "Succeeded",  
    "statusChangedAt": 1571694233.104  
  }  
}
```

- For API details, see [PeerVpc](#) in *AWS CLI Command Reference*.

reboot-instance

The following code example shows how to use `reboot-instance`.

AWS CLI

To reboot an instance

The following `reboot-instance` example reboots the specified instance.

```
aws lightsail reboot-instance \  
  --instance-name MEAN-1
```

Output:

```
{  
  "operations": [  
    {  
      "id": "2b679f1c-8b71-4bb4-8e97-8EXAMPLEed93",  
      "resourceName": "MEAN-1",  
      "resourceType": "Instance",  
      "createdAt": 1571694445.49,  
      "location": {  
        "availabilityZone": "us-west-2a",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": true,  
      "operationDetails": "",  
      "operationType": "RebootInstance",  
      "status": "Succeeded",  
      "statusChangedAt": 1571694445.49  
    }  
  ]  
}
```

- For API details, see [RebootInstance](#) in *AWS CLI Command Reference*.

reboot-relational-database

The following code example shows how to use `reboot-relational-database`.

AWS CLI

To reboot a relational database

The following `reboot-relational-database` example reboots the specified relational database.

```
aws lightsail reboot-relational-database \  
  --instance-name MEAN-1
```



```
--relational-database-name Database-1
```

Output:

```
{
  "operations": [
    {
      "id": "e4c980c0-3137-496c-9c91-1EXAMPLEdec2",
      "resourceName": "Database-1",
      "resourceType": "RelationalDatabase",
      "createdAt": 1571694532.91,
      "location": {
        "availabilityZone": "us-west-2a",
        "regionName": "us-west-2"
      },
      "isTerminal": false,
      "operationDetails": "",
      "operationType": "RebootRelationalDatabase",
      "status": "Started",
      "statusChangedAt": 1571694532.91
    }
  ]
}
```

- For API details, see [RebootRelationalDatabase](#) in *AWS CLI Command Reference*.

release-static-ip

The following code example shows how to use `release-static-ip`.

AWS CLI

To delete a static IP

The following `release-static-ip` example deletes the specified static IP.

```
aws lightsail release-static-ip \
  --static-ip-name StaticIp-1
```

Output:

```
{
```

```
"operations": [  
  {  
    "id": "e374c002-dc6d-4c7f-919f-2EXAMPLE13ce",  
    "resourceName": "StaticIp-1",  
    "resourceType": "StaticIp",  
    "createdAt": 1571694962.003,  
    "location": {  
      "availabilityZone": "all",  
      "regionName": "us-west-2"  
    },  
    "isTerminal": true,  
    "operationType": "ReleaseStaticIp",  
    "status": "Succeeded",  
    "statusChangedAt": 1571694962.003  
  }  
]  
}
```

- For API details, see [ReleaseStaticIp](#) in *AWS CLI Command Reference*.

start-instance

The following code example shows how to use `start-instance`.

AWS CLI

To start an instance

The following `start-instance` example starts the specified instance.

```
aws lightsail start-instance \  
  --instance-name WordPress-1
```

Output:

```
{  
  "operations": [  
    {  
      "id": "f88d2a93-7cea-4165-afce-2d688cb18f23",  
      "resourceName": "WordPress-1",  
      "resourceType": "Instance",  
      "createdAt": 1571695583.463,  
      "location": {  
        "availabilityZone": "us-east-1a",  
        "regionName": "us-east-1"  
      },  
      "isTerminal": true,  
      "operationType": "StartInstance",  
      "status": "InProgress",  
      "statusChangedAt": 1571695583.463  
    }  
  ]  
}
```

```
    "location": {
      "availabilityZone": "us-west-2a",
      "regionName": "us-west-2"
    },
    "isTerminal": false,
    "operationType": "StartInstance",
    "status": "Started",
    "statusChangedAt": 1571695583.463
  }
]
```

- For API details, see [StartInstance](#) in *AWS CLI Command Reference*.

start-relational-database

The following code example shows how to use `start-relational-database`.

AWS CLI

To start a relational database

The following `start-relational-database` example starts the specified relational database.

```
aws lightsail start-relational-database \  
  --relational-database-name Database-1
```

Output:

```
{  
  "operations": [  
    {  
      "id": "4d5294ec-a38a-4fda-9e37-aEXAMPLE0d24",  
      "resourceName": "Database-1",  
      "resourceType": "RelationalDatabase",  
      "createdAt": 1571695998.822,  
      "location": {  
        "availabilityZone": "us-west-2a",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": false,
```

```
        "operationType": "StartRelationalDatabase",
        "status": "Started",
        "statusChangedAt": 1571695998.822
      }
    ]
  }
}
```

- For API details, see [StartRelationalDatabase](#) in *AWS CLI Command Reference*.

stop-instance

The following code example shows how to use stop-instance.

AWS CLI

To stop an instance

The following stop-instance example stops the specified instance.

```
aws lightsail stop-instance \  
--instance-name WordPress-1
```

Output:

```
{
  "operations": [
    {
      "id": "265357e2-2943-4d51-888a-1EXAMPLE7585",
      "resourceName": "WordPress-1",
      "resourceType": "Instance",
      "createdAt": 1571695471.134,
      "location": {
        "availabilityZone": "us-west-2a",
        "regionName": "us-west-2"
      },
      "isTerminal": false,
      "operationType": "StopInstance",
      "status": "Started",
      "statusChangedAt": 1571695471.134
    }
  ]
}
```

- For API details, see [StopInstance](#) in *AWS CLI Command Reference*.

stop-relational-database

The following code example shows how to use `stop-relational-database`.

AWS CLI

To stop a relational database

The following `stop-relational-database` example stops the specified relational database.

```
aws lightsail stop-relational-database \  
  --relational-database-name Database-1
```

Output:

```
{  
  "operations": [  
    {  
      "id": "cc559c19-4adb-41e4-b75b-5EXAMPLE4e61",  
      "resourceName": "Database-1",  
      "resourceType": "RelationalDatabase",  
      "createdAt": 1571695526.29,  
      "location": {  
        "availabilityZone": "us-west-2a",  
        "regionName": "us-west-2"  
      },  
      "isTerminal": false,  
      "operationType": "StopRelationalDatabase",  
      "status": "Started",  
      "statusChangedAt": 1571695526.29  
    }  
  ]  
}
```

- For API details, see [StopRelationalDatabase](#) in *AWS CLI Command Reference*.

unpeer-vpc

The following code example shows how to use `unpeer-vpc`.

AWS CLI

To unpeer the Amazon Lightsail virtual private cloud

The following `unpeer-vpc` example unpeers the Amazon Lightsail virtual private cloud (VPC) for the specified AWS Region.

```
aws lightsail unpeer-vpc \  
  --region us-west-2
```

Output:

```
{  
  "operation": {  
    "id": "531aca64-7157-47ab-84c6-eEXAMPLEd898",  
    "resourceName": "vpc-0EXAMPLEa5261efb3",  
    "resourceType": "PeeredVpc",  
    "createdAt": 1571694109.945,  
    "location": {  
      "availabilityZone": "all",  
      "regionName": "us-west-2"  
    },  
    "isTerminal": true,  
    "operationDetails": "vpc-e2b3eb9b",  
    "operationType": "UnpeeredVpc",  
    "status": "Succeeded",  
    "statusChangedAt": 1571694109.945  
  }  
}
```

- For API details, see [UnpeerVpc](#) in *AWS CLI Command Reference*.

Macie examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Macie.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

describe-buckets

The following code example shows how to use `describe-buckets`.

AWS CLI

To query data about one or more S3 buckets that Amazon Macie monitors and analyzes for your account

The following `describe-buckets` example queries metadata for all S3 buckets whose names begin with MY-S3 and are in the current AWS Region.

```
aws macie2 describe-buckets \  
  --criteria '{"bucketName":{"prefix":"my-S3"}}'
```

Output:

```
{  
  "buckets": [  
    {  
      "accountId": "123456789012",  
      "allowsUnencryptedObjectUploads": "FALSE",  
      "bucketArn": "arn:aws:s3:::MY-S3-DOC-EXAMPLE-BUCKET1",  
      "bucketCreatedAt": "2020-05-18T19:54:00+00:00",  
      "bucketName": "MY-S3-DOC-EXAMPLE-BUCKET1",  
      "classifiableObjectCount": 13,  
      "classifiableSizeInBytes": 1592088,  
      "jobDetails": {  
        "isDefinedInJob": "TRUE",  
        "isMonitoredByJob": "TRUE",  
        "lastJobId": "08c81dc4a2f3377fae45c9ddaexample",
```

```
    "lastJobRunTime": "2021-04-26T14:55:30.270000+00:00"
  },
  "lastAutomatedDiscoveryTime": "2022-12-10T19:11:25.364000+00:00",
  "lastUpdated": "2022-12-13T07:33:06.337000+00:00",
  "objectCount": 13,
  "objectCountByEncryptionType": {
    "customerManaged": 0,
    "kmsManaged": 2,
    "s3Managed": 7,
    "unencrypted": 4,
    "unknown": 0
  },
  "publicAccess": {
    "effectivePermission": "NOT_PUBLIC",
    "permissionConfiguration": {
      "accountLevelPermissions": {
        "blockPublicAccess": {
          "blockPublicAcls": true,
          "blockPublicPolicy": true,
          "ignorePublicAcls": true,
          "restrictPublicBuckets": true
        }
      },
      "bucketLevelPermissions": {
        "accessControlList": {
          "allowsPublicReadAccess": false,
          "allowsPublicWriteAccess": false
        },
        "blockPublicAccess": {
          "blockPublicAcls": true,
          "blockPublicPolicy": true,
          "ignorePublicAcls": true,
          "restrictPublicBuckets": true
        },
        "bucketPolicy": {
          "allowsPublicReadAccess": false,
          "allowsPublicWriteAccess": false
        }
      }
    }
  },
  "region": "us-west-2",
  "replicationDetails": {
    "replicated": false,
```



```
        "replicatedExternally": false,
        "replicationAccounts": []
    },
    "sensitivityScore": 78,
    "serverSideEncryption": {
        "kmsMasterKeyId": null,
        "type": "NONE"
    },
    "sharedAccess": "NOT_SHARED",
    "sizeInBytes": 4549746,
    "sizeInBytesCompressed": 0,
    "tags": [
        {
            "key": "Division",
            "value": "HR"
        },
        {
            "key": "Team",
            "value": "Recruiting"
        }
    ],
    "unclassifiableObjectCount": {
        "fileType": 0,
        "storageClass": 0,
        "total": 0
    },
    "unclassifiableObjectSizeInBytes": {
        "fileType": 0,
        "storageClass": 0,
        "total": 0
    },
    "versioning": true
},
{
    "accountId": "123456789012",
    "allowsUnencryptedObjectUploads": "TRUE",
    "bucketArn": "arn:aws:s3:::MY-S3-DOC-EXAMPLE-BUCKET2",
    "bucketCreatedAt": "2020-11-25T18:24:38+00:00",
    "bucketName": "MY-S3-DOC-EXAMPLE-BUCKET2",
    "classifiableObjectCount": 8,
    "classifiableSizeInBytes": 133810,
    "jobDetails": {
        "isDefinedInJob": "TRUE",
        "isMonitoredByJob": "FALSE",
```

```
    "lastJobId": "188d4f6044d621771ef7d65f2example",
    "lastJobRunTime": "2021-04-09T19:37:11.511000+00:00"
  },
  "lastAutomatedDiscoveryTime": "2022-12-12T19:11:25.364000+00:00",
  "lastUpdated": "2022-12-13T07:33:06.337000+00:00",
  "objectCount": 8,
  "objectCountByEncryptionType": {
    "customerManaged": 0,
    "kmsManaged": 0,
    "s3Managed": 8,
    "unencrypted": 0,
    "unknown": 0
  },
  "publicAccess": {
    "effectivePermission": "NOT_PUBLIC",
    "permissionConfiguration": {
      "accountLevelPermissions": {
        "blockPublicAccess": {
          "blockPublicAcls": true,
          "blockPublicPolicy": true,
          "ignorePublicAcls": true,
          "restrictPublicBuckets": true
        }
      },
      "bucketLevelPermissions": {
        "accessControlList": {
          "allowsPublicReadAccess": false,
          "allowsPublicWriteAccess": false
        },
        "blockPublicAccess": {
          "blockPublicAcls": true,
          "blockPublicPolicy": true,
          "ignorePublicAcls": true,
          "restrictPublicBuckets": true
        },
        "bucketPolicy": {
          "allowsPublicReadAccess": false,
          "allowsPublicWriteAccess": false
        }
      }
    }
  },
  "region": "us-west-2",
  "replicationDetails": {
```

```
        "replicated": false,
        "replicatedExternally": false,
        "replicationAccounts": []
    },
    "sensitivityScore": 95,
    "serverSideEncryption": {
        "kmsMasterKeyId": null,
        "type": "AES256"
    },
    "sharedAccess": "EXTERNAL",
    "sizeInBytes": 175978,
    "sizeInBytesCompressed": 0,
    "tags": [
        {
            "key": "Division",
            "value": "HR"
        },
        {
            "key": "Team",
            "value": "Recruiting"
        }
    ],
    "unclassifiableObjectCount": {
        "fileType": 3,
        "storageClass": 0,
        "total": 3
    },
    "unclassifiableObjectSizeInBytes": {
        "fileType": 2999826,
        "storageClass": 0,
        "total": 2999826
    },
    "versioning": true
    }
]
```

For more information, see [Filtering your S3 bucket inventory](#) in the *Amazon Macie User Guide*.

- For API details, see [DescribeBuckets](#) in *AWS CLI Command Reference*.

Amazon Managed Grafana examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon Managed Grafana.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

list-workspaces

The following code example shows how to use `list-workspaces`.

AWS CLI

To list workspaces for the account in the Region specified by the user credential

The following `list-workspaces` example lists Grafana workspaces for the account's Region.

```
aws grafana list-workspaces
```

Output:

```
{
  "workspaces": [
    {
      "authentication": {
        "providers": [
```

```

        "AWS_SSO"
    ]
},
"created": "2022-04-04T16:20:21.796000-07:00",
"description": "to test tags",
"endpoint": "g-949e7b44df.grafana-workspace.us-east-1.amazonaws.com",
"grafanaVersion": "8.2",
"id": "g-949e7b44df",
"modified": "2022-04-04T16:20:21.796000-07:00",
"name": "testtag2",
"notificationDestinations": [
    "SNS"
],
"status": "ACTIVE"
},
{
    "authentication": {
        "providers": [
            "AWS_SSO"
        ]
    },
    "created": "2022-04-20T10:22:15.115000-07:00",
    "description": "ww",
    "endpoint": "g-bffa51ed1b.grafana-workspace.us-east-1.amazonaws.com",
    "grafanaVersion": "8.2",
    "id": "g-bffa51ed1b",
    "modified": "2022-04-20T10:22:15.115000-07:00",
    "name": "ww",
    "notificationDestinations": [
        "SNS"
    ],
    "status": "ACTIVE"
}
]
}

```

- For API details, see [ListWorkspaces](#) in *AWS CLI Command Reference*.

MediaConnect examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with MediaConnect.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-flow-outputs

The following code example shows how to use add-flow-outputs.

AWS CLI

To add outputs to a flow

The following add-flow-outputs example adds outputs to the specified flow.

```
aws mediaconnect add-flow-outputs \  
--flow-arn arn:aws:mediaconnect:us-  
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame \  
--outputs Description='NYC  
stream',Destination=192.0.2.12,Name=NYC,Port=3333,Protocol=rtp-  
fec,SmoothingLatency=100 Description='LA  
stream',Destination=203.0.113.9,Name=LA,Port=4444,Protocol=rtp-  
fec,SmoothingLatency=100
```

Output:

```
{  
  "Outputs": [  
    {  
      "Port": 3333,  
      "OutputArn": "arn:aws:mediaconnect:us-  
east-1:111122223333:output:2-3aBC45dEF67hiJ89-c34de5fG678h:NYC",
```

```

        "Name": "NYC",
        "Description": "NYC stream",
        "Destination": "192.0.2.12",
        "Transport": {
            "Protocol": "rtp-fec",
            "SmoothingLatency": 100
        }
    },
    {
        "Port": 4444,
        "OutputArn": "arn:aws:mediacconnect:us-
east-1:111122223333:output:2-987655dEF67hiJ89-c34de5fG678h:LA",
        "Name": "LA",
        "Description": "LA stream",
        "Destination": "203.0.113.9",
        "Transport": {
            "Protocol": "rtp-fec",
            "SmoothingLatency": 100
        }
    }
],
"FlowArn": "arn:aws:mediacconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame"
}

```

For more information, see [Adding Outputs to a Flow](#) in the *AWS Elemental MediaConnect User Guide*.

- For API details, see [AddFlowOutputs](#) in *AWS CLI Command Reference*.

create-flow

The following code example shows how to use create-flow.

AWS CLI

To create a flow

The following create-flow example creates a flow with the specified configuration.

```

aws mediacconnect create-flow \
  --availability-zone us-west-2c \
  --name ExampleFlow \

```

```
--source Description='Example source,
backup',IngestPort=1055,Name=BackupSource,Protocol=rtp,WhitelistCidr=10.24.34.0/23
```

Output:

```
{
  "Flow": {
    "FlowArn": "arn:aws:mediaconnect:us-
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:ExampleFlow",
    "AvailabilityZone": "us-west-2c",
    "EgressIp": "54.245.71.21",
    "Source": {
      "IngestPort": 1055,
      "SourceArn": "arn:aws:mediaconnect:us-
east-1:123456789012:source:2-3aBC45dEF67hiJ89-c34de5fG678h:BackupSource",
      "Transport": {
        "Protocol": "rtp",
        "MaxBitrate": 80000000
      },
      "Description": "Example source, backup",
      "IngestIp": "54.245.71.21",
      "WhitelistCidr": "10.24.34.0/23",
      "Name": "mySource"
    },
    "Entitlements": [],
    "Name": "ExampleFlow",
    "Outputs": [],
    "Status": "STANDBY",
    "Description": "Example source, backup"
  }
}
```

For more information, see [Creating a Flow](#) in the *AWS Elemental MediaConnect User Guide*.

- For API details, see [CreateFlow](#) in *AWS CLI Command Reference*.

delete-flow

The following code example shows how to use delete-flow.

AWS CLI

To delete a flow

The following `delete-flow` example deletes the specified flow.

```
aws mediaconnect delete-flow \  
  --flow-arn arn:aws:mediaconnect:us-  
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow
```

Output:

```
{  
  "FlowArn": "arn:aws:mediaconnect:us-  
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow",  
  "Status": "DELETING"  
}
```

For more information, see [Deleting a Flow](#) in the *AWS Elemental MediaConnect User Guide*.

- For API details, see [DeleteFlow](#) in *AWS CLI Command Reference*.

describe-flow

The following code example shows how to use `describe-flow`.

AWS CLI

To view the details of a flow

The following `describe-flow` example displays the specified flow's details, such as ARN, Availability Zone, status, source, entitlements, and outputs.

```
aws mediaconnect describe-flow \  
  --flow-arn arn:aws:mediaconnect:us-  
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow
```

Output:

```
{  
  "Flow": {  
    "EgressIp": "54.201.4.39",  
    "AvailabilityZone": "us-west-2c",  
    "Status": "ACTIVE",  
    "FlowArn": "arn:aws:mediaconnect:us-  
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow",
```

```
    "Entitlements": [
      {
        "EntitlementArn": "arn:aws:mediacconnect:us-
west-2:123456789012:entitlement:1-AaBb11CcDd22EeFf-34DE5fG12AbC:MyEntitlement",
        "Description": "Assign to this account",
        "Name": "MyEntitlement",
        "Subscribers": [
          "444455556666"
        ]
      }
    ],
    "Description": "NYC awards show",
    "Name": "AwardsShow",
    "Outputs": [
      {
        "Port": 2355,
        "Name": "NYC",
        "Transport": {
          "SmoothingLatency": 0,
          "Protocol": "rtp-fec"
        },
        "OutputArn": "arn:aws:mediacconnect:us-
east-1:123456789012:output:2-3aBC45dEF67hiJ89-c34de5fG678h:NYC",
        "Destination": "192.0.2.0"
      },
      {
        "Port": 3025,
        "Name": "LA",
        "Transport": {
          "SmoothingLatency": 0,
          "Protocol": "rtp-fec"
        },
        "OutputArn": "arn:aws:mediacconnect:us-
east-1:123456789012:output:2-987655dEF67hiJ89-c34de5fG678h:LA",
        "Destination": "192.0.2.0"
      }
    ],
    "Source": {
      "IngestIp": "54.201.4.39",
      "SourceArn": "arn:aws:mediacconnect:us-
east-1:123456789012:source:3-4aBC56dEF78hiJ90-4de5fG6Hi78Jk:ShowSource",
      "Transport": {
        "MaxBitrate": 80000000,
        "Protocol": "rtp"
```

```

    },
    "IngestPort": 1069,
    "Description": "Saturday night show",
    "Name": "ShowSource",
    "WhitelistCidr": "10.24.34.0/23"
  }
}
}

```

For more information, see [Viewing the Details of a Flow](#) in the *AWS Elemental MediaConnect User Guide*.

- For API details, see [DescribeFlow](#) in *AWS CLI Command Reference*.

grant-flow-entitlements

The following code example shows how to use grant-flow-entitlements.

AWS CLI

To grant an entitlement on a flow

The following grant-flow-entitlements example grants an entitlement to the specified existing flow to share your content with another AWS account.

```

aws mediaconnect grant-flow-entitlements \
  --flow-arn arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame \
  --entitlements Description='For
AnyCompany',Encryption={"Algorithm=aes128,KeyType=static-
key,RoleArn=arn:aws:iam::111122223333:role/MediaConnect-
ASM,SecretArn=arn:aws:secretsmanager:us-
west-2:111122223333:secret:mySecret1"},Name=AnyCompany_Entitlement,Subscribers=444455556666
Description='For Example Corp',Name=ExampleCorp,Subscribers=777788889999

```

Output:

```

{
  "Entitlements": [
    {
      "Name": "AnyCompany_Entitlement",

```

```

    "EntitlementArn": "arn:aws:mediacconnect:us-
west-2:111122223333:entitlement:1-11aa22bb11aa22bb-3333cccc4444:AnyCompany_Entitlement",
    "Subscribers": [
        "444455556666"
    ],
    "Description": "For AnyCompany",
    "Encryption": {
        "SecretArn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:mySecret1",
        "Algorithm": "aes128",
        "RoleArn": "arn:aws:iam::111122223333:role/MediaConnect-ASM",
        "KeyType": "static-key"
    }
},
{
    "Name": "ExampleCorp",
    "EntitlementArn": "arn:aws:mediacconnect:us-
west-2:111122223333:entitlement:1-3333cccc4444dddd-1111aaaa2222:ExampleCorp",
    "Subscribers": [
        "777788889999"
    ],
    "Description": "For Example Corp"
}
],
"FlowArn": "arn:aws:mediacconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame"
}

```

For more information, see [Granting an Entitlement on a Flow](#) in the *AWS Elemental MediaConnect User Guide*.

- For API details, see [GrantFlowEntitlements](#) in *AWS CLI Command Reference*.

list-entitlements

The following code example shows how to use `list-entitlements`.

AWS CLI

To view a list of entitlements

The following `list-entitlements` example displays a list of all entitlements that have been granted to the account.

```
aws mediacconnect list-entitlements
```

Output:

```
{
  "Entitlements": [
    {
      "EntitlementArn": "arn:aws:mediacconnect:us-
west-2:111122223333:entitlement:1-11aa22bb11aa22bb-3333cccc4444:MyEntitlement",
      "EntitlementName": "MyEntitlement"
    }
  ]
}
```

For more information, see [ListEntitlements](#) in the *AWS Elemental MediaConnect API Reference*.

- For API details, see [ListEntitlements](#) in *AWS CLI Command Reference*.

list-flows

The following code example shows how to use `list-flows`.

AWS CLI

To view a list of flows

The following `list-flows` example displays a list of flows.

```
aws mediacconnect list-flows
```

Output:

```
{
  "Flows": [
    {
      "Status": "STANDBY",
      "SourceType": "OWNED",
      "AvailabilityZone": "us-west-2a",
      "Description": "NYC awards show",
      "Name": "AwardsShow",

```

```

    "FlowArn": "arn:aws:mediacconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow"
  },
  {
    "Status": "STANDBY",
    "SourceType": "OWNED",
    "AvailabilityZone": "us-west-2c",
    "Description": "LA basketball game",
    "Name": "BasketballGame",
    "FlowArn": "arn:aws:mediacconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BasketballGame"
  }
]
}

```

For more information, see [Viewing a List of Flows](#) in the *AWS Elemental MediaConnect User Guide*.

- For API details, see [ListFlows](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags for a MediaConnect resource

The following `list-tags-for-resource` example displays the tag keys and values associated with the specified MediaConnect resource.

```

aws mediacconnect list-tags-for-resource \
  --resource-arn arn:aws:mediacconnect:us-
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BasketballGame

```

Output:

```

{
  "Tags": {
    "region": "west",
    "stage": "prod"
  }
}

```

```
}
```

For more information, see [ListTagsForResource](#), [TagResource](#), [UntagResource](#) in the *AWS Elemental MediaConnect API Reference*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

remove-flow-output

The following code example shows how to use `remove-flow-output`.

AWS CLI

To remove an output from a flow

The following `remove-flow-output` example removes an output from the specified flow.

```
aws mediaconnect remove-flow-output \
  --flow-arn arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame \
  --output-arn arn:aws:mediaconnect:us-
east-1:111122223333:output:2-3aBC45dEF67hiJ89-c34de5fG678h:NYC
```

Output:

```
{
  "FlowArn": "arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame",
  "OutputArn": "arn:aws:mediaconnect:us-
east-1:111122223333:output:2-3aBC45dEF67hiJ89-c34de5fG678h:NYC"
}
```

For more information, see [Removing Outputs from a Flow](#) in the *AWS Elemental MediaConnect User Guide*.

- For API details, see [RemoveFlowOutput](#) in *AWS CLI Command Reference*.

revoke-flow-entitlement

The following code example shows how to use `revoke-flow-entitlement`.

AWS CLI

To revoke an entitlement

The following `revoke-flow-entitlement` example revokes an entitlement on the specified flow.

```
aws mediaconnect revoke-flow-entitlement \  
  --flow-arn arn:aws:mediaconnect:us-  
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame \  
  --entitlement-arn arn:aws:mediaconnect:us-  
west-2:111122223333:entitlement:1-11aa22bb11aa22bb-3333cccc4444:AnyCompany_Entitlement
```

Output:

```
{  
  "FlowArn": "arn:aws:mediaconnect:us-  
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame",  
  "EntitlementArn": "arn:aws:mediaconnect:us-  
west-2:111122223333:entitlement:1-11aa22bb11aa22bb-3333cccc4444:AnyCompany_Entitlement"  
}
```

For more information, see [Revoking an Entitlement](#) in the *AWS Elemental MediaConnect User Guide*.

- For API details, see [RevokeFlowEntitlement](#) in *AWS CLI Command Reference*.

start-flow

The following code example shows how to use `start-flow`.

AWS CLI

To start a flow

The following `start-flow` example starts the specified flow.

```
aws mediaconnect start-flow \  
  --flow-arn arn:aws:mediaconnect:us-  
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow
```


This command produces no output. Output:

```
{
  "FlowArn": "arn:aws:mediacconnect:us-
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow",
  "Status": "STARTING"
}
```

For more information, see [Starting a Flow](#) in the *AWS Elemental MediaConnect User Guide*.

- For API details, see [StartFlow](#) in *AWS CLI Command Reference*.

stop-flow

The following code example shows how to use stop-flow.

AWS CLI

To stop a flow

The following stop-flow example stops the specified flow.

```
aws mediacconnect stop-flow \
  --flow-arn arn:aws:mediacconnect:us-
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow
```

Output:

```
{
  "Status": "STOPPING",
  "FlowArn": "arn:aws:mediacconnect:us-
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow"
}
```

For more information, see [Stopping a Flow](#) in the *AWS Elemental MediaConnect User Guide*.

- For API details, see [StopFlow](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To add tags to a MediaConnect resource

The following `tag-resource` example adds a tag with a key name and value to the specified MediaConnect resource.

```
aws mediaconnect tag-resource \  
  --resource-arn arn:aws:mediaconnect:us-  
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BasketballGame  
  --tags region=west
```

This command produces no output.

For more information, see [ListTagsForResource](#), [TagResource](#), [UntagResource](#) in the *AWS Elemental MediaConnect API Reference*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from a MediaConnect resource

The following `untag-resource` example remove the tag with the specified key name and its associated value from a MediaConnect resource.

```
aws mediaconnect untag-resource \  
  --resource-arn arn:aws:mediaconnect:us-  
east-1:123456789012:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BasketballGame \  
  --tag-keys region
```

This command produces no output.

For more information, see [ListTagsForResource](#), [TagResource](#), [UntagResource](#) in the *AWS Elemental MediaConnect API Reference*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-flow-entitlement

The following code example shows how to use update-flow-entitlement.

AWS CLI

To update an entitlement

The following update-flow-entitlement example updates the specified entitlement with a new description and subscriber.

```
aws mediaconnect update-flow-entitlement \
  --flow-arn arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame \
  --entitlement-arn arn:aws:mediaconnect:us-
west-2:111122223333:entitlement:1-11aa22bb11aa22bb-3333cccc4444:AnyCompany_Entitlement
\
  --description 'For AnyCompany Affiliate' \
  --subscribers 777788889999
```

Output:

```
{
  "FlowArn": "arn:aws:mediaconnect:us-
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame",
  "Entitlement": {
    "Name": "AnyCompany_Entitlement",
    "Description": "For AnyCompany Affiliate",
    "EntitlementArn": "arn:aws:mediaconnect:us-
west-2:111122223333:entitlement:1-11aa22bb11aa22bb-3333cccc4444:AnyCompany_Entitlement",
    "Encryption": {
      "KeyType": "static-key",
      "Algorithm": "aes128",
      "RoleArn": "arn:aws:iam::111122223333:role/MediaConnect-ASM",
      "SecretArn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:mySecret1"
    },
    "Subscribers": [
      "777788889999"
    ]
  }
}
```

For more information, see [Updating an Entitlement](#) in the *AWS Elemental MediaConnect User Guide*.

- For API details, see [UpdateFlowEntitlement](#) in *AWS CLI Command Reference*.

update-flow-output

The following code example shows how to use `update-flow-output`.

AWS CLI

To update an output on a flow

The following `update-flow-output` example update an output on the specified flow.

```
aws mediaconnect update-flow-output \  
  --flow-arn arn:aws:mediaconnect:us-  
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame \  
  --output-arn arn:aws:mediaconnect:us-  
east-1:111122223333:output:2-3aBC45dEF67hiJ89-c34de5fG678h:NYC \  
  --port 3331
```

Output:

```
{  
  "FlowArn": "arn:aws:mediaconnect:us-  
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:BaseballGame",  
  "Output": {  
    "Name": "NYC",  
    "Port": 3331,  
    "Description": "NYC stream",  
    "Transport": {  
      "Protocol": "rtsp-fec",  
      "SmoothingLatency": 100  
    },  
    "OutputArn": "arn:aws:mediaconnect:us-  
east-1:111122223333:output:2-3aBC45dEF67hiJ89-c34de5fG678h:NYC",  
    "Destination": "192.0.2.12"  
  }  
}
```

For more information, see [Updating Outputs on a Flow](#) in the *AWS Elemental MediaConnect User Guide*.

- For API details, see [UpdateFlowOutput](#) in *AWS CLI Command Reference*.

update-flow-source

The following code example shows how to use `update-flow-source`.

AWS CLI

To update the source of an existing flow

The following `update-flow-source` example updates the source of an existing flow.

```
aws mediaconnect update-flow-source \  
  --flow-arn arn:aws:mediaconnect:us-  
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow \  
  --source-arn arn:aws:mediaconnect:us-  
east-1:111122223333:source:3-4aBC56dEF78hiJ90-4de5fG6Hi78Jk:ShowSource \  
  --description 'Friday night show' \  
  --ingest-port 3344 \  
  --protocol rtp-fec \  
  --whitelist-cidr 10.24.34.0/23
```

Output:

```
{  
  "FlowArn": "arn:aws:mediaconnect:us-  
east-1:111122223333:flow:1-23aBC45dEF67hiJ8-12AbC34DE5fG:AwardsShow",  
  "Source": {  
    "IngestIp": "34.210.136.56",  
    "WhitelistCidr": "10.24.34.0/23",  
    "Transport": {  
      "Protocol": "rtp-fec"  
    },  
    "IngestPort": 3344,  
    "Name": "ShowSource",  
    "Description": "Friday night show",  
    "SourceArn": "arn:aws:mediaconnect:us-  
east-1:111122223333:source:3-4aBC56dEF78hiJ90-4de5fG6Hi78Jk:ShowSource"  
  }  
}
```

For more information, see [Updating the Source of a Flow](#) in the *AWS Elemental MediaConnect User Guide*.

- For API details, see [UpdateFlowSource](#) in *AWS CLI Command Reference*.

MediaConvert examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with MediaConvert.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

cancel-job

The following code example shows how to use `cancel-job`.

AWS CLI

To cancel a job that is in a queue

The following `cancel-job` example cancels the job with ID `1234567891234-abc123`. You can't cancel a job that the service has started processing.

```
aws mediaconvert cancel-job \  
  --endpoint-url https://abcd1234.mediaconvert.region-name-1.amazonaws.com \  
  --region region-name-1 \  
  --id 1234567891234-abc123
```

To get your account-specific endpoint, use `describe-endpoints`, or send the command without the endpoint. The service returns an error and your endpoint.

For more information, see [Working with AWS Elemental MediaConvert Jobs](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [CancelJob](#) in *AWS CLI Command Reference*.

create-job-template

The following code example shows how to use `create-job-template`.

AWS CLI

To create a job template

The following `create-job-template` example creates a job template with the transcoding settings that are specified in the file `job-template.json` that resides on your system.

```
aws mediaconvert create-job-template \  
  --endpoint-url https://abcd1234.mediaconvert.region-name-1.amazonaws.com \  
  --region region-name-1 \  
  --name JobTemplate1 \  
  --cli-input-json file://~/job-template.json
```

If you create your job template JSON file by using `get-job-template` and then modifying the file, remove the `JobTemplate` object, but keep the `Settings` child object inside it. Also, make sure to remove the following key-value pairs: `LastUpdated`, `Arn`, `Type`, and `CreatedAt`. You can specify the category, description, name, and queue either in the JSON file or at the command line.

To get your account-specific endpoint, use `describe-endpoints`, or send the command without the endpoint. The service returns an error and your endpoint.

If your request is successful, the service returns the JSON specification for the job template that you created.

For more information, see [Working with AWS Elemental MediaConvert Job Templates](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [CreateJobTemplate](#) in *AWS CLI Command Reference*.

create-job

The following code example shows how to use `create-job`.

AWS CLI

To create a job

The following `create-job` example creates a transcoding job with the settings that are specified in a file `job.json` that resides on the system that you send the command from. This JSON job specification might specify each setting individually, reference a job template, or reference output presets.

```
aws mediaconvert create-job \  
  --endpoint-url https://abcd1234.mediaconvert.region-name-1.amazonaws.com \  
  --region region-name-1 \  
  --cli-input-json file://~/job.json
```

You can use the AWS Elemental MediaConvert console to generate the JSON job specification by choosing your job settings, and then choosing **Show job JSON** at the bottom of the **Job** section.

To get your account-specific endpoint, use `describe-endpoints`, or send the command without the endpoint. The service returns an error and your endpoint.

If your request is successful, the service returns the JSON job specification that you sent with your request.

For more information, see [Working with AWS Elemental MediaConvert Jobs](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [CreateJob](#) in *AWS CLI Command Reference*.

create-preset

The following code example shows how to use `create-preset`.

AWS CLI

To create a custom output preset

The following `create-preset` example creates a custom output preset based on the output settings that are specified in the file `preset.json`. You can specify the category, description, and name either in the JSON file or at the command line.

```
aws mediaconvert create-preset \  
  --endpoint-url https://abcd1234.mediaconvert.region-name-1.amazonaws.com \  
  --region region-name-1 \  
  --cli-input-json file://~/preset.json
```

If you create your preset JSON file by using `get-preset` and then modifying the output file, ensure that you remove the following key-value pairs: `LastUpdated`, `Arn`, `Type`, and `CreatedAt`.

To get your account-specific endpoint, use `describe-endpoints`, or send the command without the endpoint. The service returns an error and your endpoint.

For more information, see [Working with AWS Elemental MediaConvert Output Presets](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [CreatePreset](#) in *AWS CLI Command Reference*.

create-queue

The following code example shows how to use `create-queue`.

AWS CLI

To create a custom queue

The following `create-queue` example creates a custom transcoding queue.

```
aws mediaconvert create-queue \  
  --endpoint-url https://abcd1234.mediaconvert.region-name-1.amazonaws.com \  
  --region region-name-1 \  
  --name Queue1 \  
  --description "Keep this queue empty unless job is urgent."
```

To get your account-specific endpoint, use `describe-endpoints`, or send the command without the endpoint. The service returns an error and your endpoint.

Output:

```
{
  "Queue": {
    "Status": "ACTIVE",
    "Name": "Queue1",
    "LastUpdated": 1518034928,
    "Arn": "arn:aws:mediaconvert:region-name-1:012345678998:queues/Queue1",
    "Type": "CUSTOM",
    "CreatedAt": 1518034928,
    "Description": "Keep this queue empty unless job is urgent."
  }
}
```

For more information, see [Working with AWS Elemental MediaConvert Queues](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [CreateQueue](#) in *AWS CLI Command Reference*.

delete-job-template

The following code example shows how to use `delete-job-template`.

AWS CLI

To delete a job template

The following `delete-job-template` example deletes the specified custom job template.

```
aws mediaconvert delete-job-template \
  --name "DASH Streaming" \
  --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

This command produces no output. Run `aws mediaconvert list-job-templates` to confirm that your template was deleted.

For more information, see [Working with AWS Elemental MediaConvert Job Templates](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [DeleteJobTemplate](#) in *AWS CLI Command Reference*.

delete-preset

The following code example shows how to use `delete-preset`.

AWS CLI

To delete a custom on-demand queue

The following `delete-preset` example deletes the specified custom preset.

```
aws mediaconvert delete-preset \  
  --name SimpleMP4 \  
  --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

This command produces no output. Run `aws mediaconvert list-presets` to confirm that your preset was deleted.

For more information, see [Working with AWS Elemental MediaConvert Output Presets](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [DeletePreset](#) in *AWS CLI Command Reference*.

delete-queue

The following code example shows how to use `delete-queue`.

AWS CLI

To delete a custom on-demand queue

The following `delete-queue` example deletes the specified custom on-demand queue.

You can't delete your default queue. You can't delete a reserved queue that has an active pricing plan or that contains unprocessed jobs.

```
aws mediaconvert delete-queue \  
  --name Customer1 \  
  --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

This command produces no output. Run `aws mediaconvert list-queues` to confirm that your queue was deleted.

For more information, see [Working with AWS Elemental MediaConvert Queues](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [DeleteQueue](#) in *AWS CLI Command Reference*.

describe-endpoints

The following code example shows how to use `describe-endpoints`.

AWS CLI

To get your account-specific endpoint

The following `describe-endpoints` example retrieves the endpoint that you need to send any other request to the service.

```
aws mediaconvert describe-endpoints
```

Output:

```
{
  "Endpoints": [
    {
      "Url": "https://abcd1234.mediaconvert.region-name-1.amazonaws.com"
    }
  ]
}
```

For more information, see [Getting Started with MediaConvert Using the API](#) in the *AWS Elemental MediaConvert API Reference*.

- For API details, see [DescribeEndpoints](#) in *AWS CLI Command Reference*.

get-job-template

The following code example shows how to use `get-job-template`.

AWS CLI

To get details for a job template

The following `get-job-template` example displays the JSON definition of the specified custom job template.

```
aws mediaconvert get-job-template \
  --name "DASH Streaming" \
  --endpoint-url https://abcd1234.mediaconvert.us-east-1.amazonaws.com
```

Output:

```
{
  "JobTemplate": {
    "StatusUpdateInterval": "SECONDS_60",
    "LastUpdated": 1568652998,
    "Description": "Create a DASH streaming ABR stack",
    "CreatedAt": 1568652998,
    "Priority": 0,
    "Name": "DASH Streaming",
    "Settings": {
      ...<truncatedforbrevity>...
    },
    "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:jobTemplates/DASH
Streaming",
    "Type": "CUSTOM"
  }
}
```

For more information, see [Working with AWS Elemental MediaConvert Job Templates](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [GetJobTemplate](#) in *AWS CLI Command Reference*.

get-job

The following code example shows how to use `get-job`.

AWS CLI

To get details for a particular job

The following example requests the information for the job with ID `1234567890987-1ab2c3`, which in this example ended in an error.

```
aws mediaconvert get-job \
  --endpoint-url https://abcd1234.mediaconvert.region-name-1.amazonaws.com \
  --region region-name-1 \
  --id 1234567890987-1ab2c3
```

To get your account-specific endpoint, use `describe-endpoints`, or send the command without the endpoint. The service returns an error and your endpoint.

If your request is successful, the service returns a JSON file with job information, including job settings, any returned errors, and other job data, as follows:

```
{
  "Job": {
    "Status": "ERROR",
    "Queue": "arn:aws:mediaconvert:region-name-1:012345678998:queues/Queue1",
    "Settings": {
      ...<truncated for brevity>...
    },
    "ErrorMessage": "Unable to open input file [s3://my-input-bucket/file-
name.mp4]: [Failed probe/open: [Failed to read data: AssumeRole failed]]",
    "ErrorCode": 1434,
    "Role": "arn:aws:iam::012345678998:role/MediaConvertServiceRole",
    "Arn": "arn:aws:mediaconvert:us-
west-1:012345678998:jobs/1234567890987-1ab2c3",
    "UserMetadata": {},
    "Timing": {
      "FinishTime": 1517442131,
      "SubmitTime": 1517442103,
      "StartTime": 1517442104
    },
    "Id": "1234567890987-1ab2c3",
    "CreatedAt": 1517442103
  }
}
```

For more information, see [Working with AWS Elemental MediaConvert Jobs](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [GetJob](#) in *AWS CLI Command Reference*.

get-preset

The following code example shows how to use `get-preset`.

AWS CLI

To get details for a particular preset

The following `get-preset` example requests the JSON definition of the specified custom preset.

```
aws mediaconvert get-preset \  
  --name SimpleMP4 \  
  --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

Output:

```
{  
  "Preset": {  
    "Description": "Creates basic MP4 file. No filtering or preprocessing.",  
    "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:presets/SimpleMP4",  
    "LastUpdated": 1568843141,  
    "Name": "SimpleMP4",  
    "Settings": {  
      "ContainerSettings": {  
        "Mp4Settings": {  
          "FreeSpaceBox": "EXCLUDE",  
          "CslgAtom": "INCLUDE",  
          "MoovPlacement": "PROGRESSIVE_DOWNLOAD"  
        },  
        "Container": "MP4"  
      },  
      "AudioDescriptions": [  
        {  
          "LanguageCodeControl": "FOLLOW_INPUT",  
          "AudioTypeControl": "FOLLOW_INPUT",  
          "CodecSettings": {  
            "AacSettings": {  
              "RawFormat": "NONE",  
              "CodecProfile": "LC",  
              "AudioDescriptionBroadcasterMix": "NORMAL",  
              "SampleRate": 48000,  
              "Bitrate": 96000,  
              "RateControlMode": "CBR",  
              "Specification": "MPEG4",  
              "CodingMode": "CODING_MODE_2_0"  
            },  
            "Codec": "AAC"  
          }  
        }  
      ],  
      "VideoDescription": {  
        "RespondToAfd": "NONE",  
        "TimecodeInsertion": "DISABLED",
```

```
"Sharpness": 50,
"ColorMetadata": "INSERT",
"CodecSettings": {
  "H264Settings": {
    "FramerateControl": "INITIALIZE_FROM_SOURCE",
    "SpatialAdaptiveQuantization": "ENABLED",
    "Softness": 0,
    "Telecine": "NONE",
    "CodecLevel": "AUTO",
    "QualityTuningLevel": "SINGLE_PASS",
    "UnregisteredSeiTimecode": "DISABLED",
    "Slices": 1,
    "Syntax": "DEFAULT",
    "GopClosedCadence": 1,
    "AdaptiveQuantization": "HIGH",
    "EntropyEncoding": "CABAC",
    "InterlaceMode": "PROGRESSIVE",
    "ParControl": "INITIALIZE_FROM_SOURCE",
    "NumberBFramesBetweenReferenceFrames": 2,
    "GopSizeUnits": "FRAMES",
    "RepeatPps": "DISABLED",
    "CodecProfile": "MAIN",
    "FieldEncoding": "PAFF",
    "GopSize": 90.0,
    "SlowPal": "DISABLED",
    "SceneChangeDetect": "ENABLED",
    "GopBReference": "DISABLED",
    "RateControlMode": "CBR",
    "FramerateConversionAlgorithm": "DUPLICATE_DROP",
    "FlickerAdaptiveQuantization": "DISABLED",
    "DynamicSubGop": "STATIC",
    "MinIInterval": 0,
    "TemporalAdaptiveQuantization": "ENABLED",
    "Bitrate": 400000,
    "NumberReferenceFrames": 3
  },
  "Codec": "H_264"
},
"AfdSignaling": "NONE",
"AntiAlias": "ENABLED",
"ScalingBehavior": "DEFAULT",
"DropFrameTimecode": "ENABLED"
},
},
```



```
    "Type": "CUSTOM",
    "CreatedAt": 1568841521
  }
}
```

For more information, see [Working with AWS Elemental MediaConvert Output Presets](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [GetPreset](#) in *AWS CLI Command Reference*.

get-queue

The following code example shows how to use `get-queue`.

AWS CLI

To get details for a queue

The following `get-queue` example retrieves the details of the specified custom queue.

```
aws mediaconvert get-queue \
  --name Customer1 \
  --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

Output:

```
{
  "Queue": {
    "LastUpdated": 1526428502,
    "Type": "CUSTOM",
    "SubmittedJobsCount": 0,
    "Status": "ACTIVE",
    "PricingPlan": "ON_DEMAND",
    "CreatedAt": 1526428502,
    "ProgressingJobsCount": 0,
    "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:queues/Customer1",
    "Name": "Customer1"
  }
}
```

For more information, see [Working with AWS Elemental MediaConvert Queues](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [GetQueue](#) in *AWS CLI Command Reference*.

list-job-templates

The following code example shows how to use `list-job-templates`.

AWS CLI

Example 1: To list your custom job templates

The following `list-job-templates` example lists all custom job templates in the current Region. To list the system job templates, see the next example.

```
aws mediaconvert list-job-templates \  
  --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

Output:

```
{  
  "JobTemplates": [  
    {  
      "Description": "Create a DASH streaming ABR stack",  
      "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:jobTemplates/DASH  
Streaming",  
      "Name": "DASH Streaming",  
      "LastUpdated": 1568653007,  
      "Priority": 0,  
      "Settings": {  
        ...<truncatedforbrevity>...  
      },  
      "Type": "CUSTOM",  
      "StatusUpdateInterval": "SECONDS_60",  
      "CreatedAt": 1568653007  
    },  
    {  
      "Description": "Create a high-res file",  
      "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:jobTemplates/File",  
      "Name": "File",  
      "LastUpdated": 1568653007,  
      "Priority": 0,  
      "Settings": {  
        ...<truncatedforbrevity>...  
      },  
    }  
  ]  
}
```

```

        "Type": "CUSTOM",
        "StatusUpdateInterval": "SECONDS_60",
        "CreatedAt": 1568653023
    }
]
}

```

Example 2: To list the MediaConvert system job templates

The following `list-job-templates` example lists all system job templates.

```

aws mediaconvert list-job-templates \
  --endpoint-url https://abcd1234.mediaconvert.us-east-1.amazonaws.com \
  --list-by SYSTEM

```

Output:

```

{
  "JobTemplates": [
    {
      "CreatedAt": 1568321779,
      "Arn": "arn:aws:mediaconvert:us-east-1:123456789012:jobTemplates/System-
Generic_Mp4_Hev1_Avc_Aac_Sdr_Qvbr",
      "Name": "System-Generic_Mp4_Hev1_Avc_Aac_Sdr_Qvbr",
      "Description": "GENERIC, MP4, AVC + HEV1(HEVC,SDR), AAC, SDR, QVBR",
      "Category": "GENERIC",
      "Settings": {
        "AdAvailOffset": 0,
        "OutputGroups": [
          {
            "Outputs": [
              {
                "Extension": "mp4",
                "Preset": "System-
Generic_Hd_Mp4_Avc_Aac_16x9_Sdr_1280x720p_30Hz_5Mbps_Qvbr_Vq9",
                "NameModifier":
                "_Generic_Hd_Mp4_Avc_Aac_16x9_Sdr_1280x720p_30Hz_5000Kbps_Qvbr_Vq9"
              },
              {
                "Extension": "mp4",
                "Preset": "System-
Generic_Hd_Mp4_Avc_Aac_16x9_Sdr_1920x1080p_30Hz_10Mbps_Qvbr_Vq9",

```

```

        "NameModifier":
        "_Generic_Hd_Mp4_Avc_Aac_16x9_Sdr_1920x1080p_30Hz_10000Kbps_Qvbr_Vq9"
        },
        {
            "Extension": "mp4",
            "Preset": "System-
Generic_Sd_Mp4_Avc_Aac_16x9_Sdr_640x360p_30Hz_0.8Mbps_Qvbr_Vq7",
            "NameModifier":
            "_Generic_Sd_Mp4_Avc_Aac_16x9_Sdr_640x360p_30Hz_800Kbps_Qvbr_Vq7"
            },
            {
                "Extension": "mp4",
                "Preset": "System-
Generic_Hd_Mp4_Hev1_Aac_16x9_Sdr_1280x720p_30Hz_4Mbps_Qvbr_Vq9",
                "NameModifier":
                "_Generic_Hd_Mp4_Hev1_Aac_16x9_Sdr_1280x720p_30Hz_4000Kbps_Qvbr_Vq9"
                },
                {
                    "Extension": "mp4",
                    "Preset": "System-
Generic_Hd_Mp4_Hev1_Aac_16x9_Sdr_1920x1080p_30Hz_8Mbps_Qvbr_Vq9",
                    "NameModifier":
                    "_Generic_Hd_Mp4_Hev1_Aac_16x9_Sdr_1920x1080p_30Hz_8000Kbps_Qvbr_Vq9"
                    },
                    {
                        "Extension": "mp4",
                        "Preset": "System-
Generic_Uhd_Mp4_Hev1_Aac_16x9_Sdr_3840x2160p_30Hz_12Mbps_Qvbr_Vq9",
                        "NameModifier":
                        "_Generic_Uhd_Mp4_Hev1_Aac_16x9_Sdr_3840x2160p_30Hz_12000Kbps_Qvbr_Vq9"
                    }
                ],
                "OutputGroupSettings": {
                    "FileGroupSettings": {

                    },
                    "Type": "FILE_GROUP_SETTINGS"
                },
                "Name": "File Group"
            }
        ]
    },
    "Type": "SYSTEM",
    "LastUpdated": 1568321779

```

```
    },  
    ...<truncatedforbrevity>...  
  ]  
}
```

For more information, see [Working with AWS Elemental MediaConvert Job Templates](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [ListJobTemplates](#) in *AWS CLI Command Reference*.

list-jobs

The following code example shows how to use `list-jobs`.

AWS CLI

To get details for all jobs in a region

The following example requests the information for all of your jobs in the specified region.

```
aws mediaconvert list-jobs \  
  --endpoint-url https://abcd1234.mediaconvert.region-name-1.amazonaws.com \  
  --region region-name-1
```

To get your account-specific endpoint, use `describe-endpoints`, or send the command without the endpoint. The service returns an error and your endpoint.

For more information, see [Working with AWS Elemental MediaConvert Jobs](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [ListJobs](#) in *AWS CLI Command Reference*.

list-presets

The following code example shows how to use `list-presets`.

AWS CLI

Example 1: To list your custom output presets

The following `list-presets` example lists your custom output presets. To list the system presets, see the next example.

```
aws mediaconvert list-presets \  
  --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

Output:

```
{  
  "Presets": [  
    {  
      "Name": "SimpleMP4",  
      "CreatedAt": 1568841521,  
      "Settings": {  
        .....  
      },  
      "Arn": "arn:aws:mediaconvert:us-east-1:003235472598:presets/SimpleMP4",  
      "Type": "CUSTOM",  
      "LastUpdated": 1568843141,  
      "Description": "Creates basic MP4 file. No filtering or preprocessing."  
    },  
    {  
      "Name": "SimpleTS",  
      "CreatedAt": 1568843113,  
      "Settings": {  
        ... truncated for brevity ...  
      },  
      "Arn": "arn:aws:mediaconvert:us-east-1:003235472598:presets/SimpleTS",  
      "Type": "CUSTOM",  
      "LastUpdated": 1568843113,  
      "Description": "Create a basic transport stream."  
    }  
  ]  
}
```

Example 2: To list the system output presets

The following `list-presets` example lists the available MediaConvert system presets. To list your custom presets, see the previous example.

```
aws mediaconvert list-presets \  
  --list-by SYSTEM \  
  --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

Output:

```

{
  "Presets": [
    {
      "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:presets/System-
      Avc_16x9_1080p_29_97fps_8500kbps",
      "Name": "System-Avc_16x9_1080p_29_97fps_8500kbps",
      "CreatedAt": 1568321789,
      "Description": "Wifi, 1920x1080, 16:9, 29.97fps, 8500kbps",
      "LastUpdated": 1568321789,
      "Type": "SYSTEM",
      "Category": "HLS",
      "Settings": {
        ...<output settings removed for brevity>...
      }
    },
    ...<list of presets shortened for brevity>...

    {
      "Arn": "arn:aws:mediaconvert:us-east-1:123456789012:presets/System-
      Xdcam_HD_1080i_29_97fps_35mpbs",
      "Name": "System-Xdcam_HD_1080i_29_97fps_35mpbs",
      "CreatedAt": 1568321790,
      "Description": "XDCAM MPEG HD, 1920x1080i, 29.97fps, 35mpbs",
      "LastUpdated": 1568321790,
      "Type": "SYSTEM",
      "Category": "MXF",
      "Settings": {
        ...<output settings removed for brevity>...
      }
    }
  ]
}

```

For more information, see [Working with AWS Elemental MediaConvert Output Presets](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [ListPresets](#) in *AWS CLI Command Reference*.

list-queues

The following code example shows how to use `list-queues`.

AWS CLI

To list your queues

The following `list-queues` example lists all of your MediaConvert queues.

```
aws mediaconvert list-queues \  
  --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

Output:

```
{  
  "Queues": [  
    {  
      "PricingPlan": "ON_DEMAND",  
      "Type": "SYSTEM",  
      "Status": "ACTIVE",  
      "CreatedAt": 1503451595,  
      "Name": "Default",  
      "SubmittedJobsCount": 0,  
      "ProgressingJobsCount": 0,  
      "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:queues/Default",  
      "LastUpdated": 1534549158  
    },  
    {  
      "PricingPlan": "ON_DEMAND",  
      "Type": "CUSTOM",  
      "Status": "ACTIVE",  
      "CreatedAt": 1537460025,  
      "Name": "Customer1",  
      "SubmittedJobsCount": 0,  
      "Description": "Jobs we run for our cusotmer.",  
      "ProgressingJobsCount": 0,  
      "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:queues/Customer1",  
      "LastUpdated": 1537460025  
    },  
    {  
      "ProgressingJobsCount": 0,  
      "Status": "ACTIVE",  
      "Name": "transcode-library",  
      "SubmittedJobsCount": 0,  
      "LastUpdated": 1564066204,  
      "ReservationPlan": {
```



```

        "Status": "ACTIVE",
        "ReservedSlots": 1,
        "PurchasedAt": 1564066203,
        "Commitment": "ONE_YEAR",
        "ExpiresAt": 1595688603,
        "RenewalType": "EXPIRE"
    },
    "PricingPlan": "RESERVED",
    "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:queues/transcode-
library",
    "Type": "CUSTOM",
    "CreatedAt": 1564066204
  }
]
}

```

For more information, see [Working with AWS Elemental MediaConvert Queues](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [ListQueues](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags on a MediaConvert queue, job template, or output preset

The following `list-tags-for-resource` example lists the tags on the specified output preset.

```

aws mediaconvert list-tags-for-resource \
  --arn arn:aws:mediaconvert:us-west-2:123456789012:presets/SimpleMP4 \
  --endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com

```

Output:

```

{
  "ResourceTags": {
    "Tags": {
      "customer": "zippyVideo"
    }
  },

```

```

    "Arn": "arn:aws:mediaconvert:us-west-2:123456789012:presets/SimpleMP4"
  }
}

```

For more information, see [Tagging AWS Elemental MediaConvert Queues, Job Templates, and Output Presets](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

update-job-template

The following code example shows how to use `update-job-template`.

AWS CLI

To change a job template

The following `update-job-template` example replaces the JSON definition of the specified custom job template with the JSON definition in the provided file.

```
aws mediaconvert update-job-template --name File1 --endpoint-url https://
abcd1234.mediaconvert.us-west-2.amazonaws.com --cli-input-json file://~/job-template-
update.json
```

Contents of `job-template-update.json`:

```

{
  "Description": "A simple job template that generates a single file output.",
  "Queue": "arn:aws:mediaconvert:us-east-1:012345678998:queues/Default",
  "Name": "SimpleFile",
  "Settings": {
    "OutputGroups": [
      {
        "Name": "File Group",
        "Outputs": [
          {
            "ContainerSettings": {
              "Container": "MP4",
              "Mp4Settings": {
                "CslgAtom": "INCLUDE",
                "FreeSpaceBox": "EXCLUDE",
                "MoovPlacement": "PROGRESSIVE_DOWNLOAD"
              }
            }
          }
        ]
      }
    ]
  }
}

```

```
    },
    "VideoDescription": {
      "ScalingBehavior": "DEFAULT",
      "TimecodeInsertion": "DISABLED",
      "AntiAlias": "ENABLED",
      "Sharpness": 50,
      "CodecSettings": {
        "Codec": "H_264",
        "H264Settings": {
          "InterlaceMode": "PROGRESSIVE",
          "NumberReferenceFrames": 3,
          "Syntax": "DEFAULT",
          "Softness": 0,
          "GopClosedCadence": 1,
          "GopSize": 90,
          "Slices": 1,
          "GopBReference": "DISABLED",
          "SlowPal": "DISABLED",
          "SpatialAdaptiveQuantization": "ENABLED",
          "TemporalAdaptiveQuantization": "ENABLED",
          "FlickerAdaptiveQuantization": "DISABLED",
          "EntropyEncoding": "CABAC",
          "Bitrate": 400000,
          "FramerateControl": "INITIALIZE_FROM_SOURCE",
          "RateControlMode": "CBR",
          "CodecProfile": "MAIN",
          "Telecine": "NONE",
          "MinIInterval": 0,
          "AdaptiveQuantization": "HIGH",
          "CodecLevel": "AUTO",
          "FieldEncoding": "PAFF",
          "SceneChangeDetect": "ENABLED",
          "QualityTuningLevel": "SINGLE_PASS",
          "FramerateConversionAlgorithm": "DUPLICATE_DROP",
          "UnregisteredSeiTimecode": "DISABLED",
          "GopSizeUnits": "FRAMES",
          "ParControl": "INITIALIZE_FROM_SOURCE",
          "NumberBFramesBetweenReferenceFrames": 2,
          "RepeatPps": "DISABLED",
          "DynamicSubGop": "STATIC"
        }
      }
    },
    "AfdSignaling": "NONE",
    "DropFrameTimecode": "ENABLED",
```

```

        "RespondToAfd": "NONE",
        "ColorMetadata": "INSERT"
    },
    "AudioDescriptions": [
        {
            "AudioTypeControl": "FOLLOW_INPUT",
            "CodecSettings": {
                "Codec": "AAC",
                "AacSettings": {
                    "AudioDescriptionBroadcasterMix": "NORMAL",
                    "Bitrate": 96000,
                    "RateControlMode": "CBR",
                    "CodecProfile": "LC",
                    "CodingMode": "CODING_MODE_2_0",
                    "RawFormat": "NONE",
                    "SampleRate": 48000,
                    "Specification": "MPEG4"
                }
            },
            "LanguageCodeControl": "FOLLOW_INPUT"
        }
    ]
}
],
"OutputGroupSettings": {
    "Type": "FILE_GROUP_SETTINGS",
    "FileGroupSettings": {}
}
},
"AdAvailOffset": 0
},
"StatusUpdateInterval": "SECONDS_60",
"Priority": 0
}

```

The system returns the JSON payload that you send with your request, even when the request results in an error. Therefore, the JSON returned is not necessarily the new definition of the job template.

Because the JSON payload can be long, you might need to scroll up to see any error messages.

For more information, see [Working with AWS Elemental MediaConvert Job Templates](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [UpdateJobTemplate](#) in *AWS CLI Command Reference*.

update-preset

The following code example shows how to use update-preset.

AWS CLI

To change a preset

The following update-preset example replaces the description for the specified preset.

```
aws mediaconvert update-preset \  
--name Customer1 \  
--description "New description text." \  
--endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

This command produces no output. Output:

```
{  
  "Preset": {  
    "Arn": "arn:aws:mediaconvert:us-east-1:003235472598:presets/SimpleMP4",  
    "Settings": {  
      ...<output settings removed for brevity>...  
    },  
    "Type": "CUSTOM",  
    "LastUpdated": 1568938411,  
    "Description": "New description text.",  
    "Name": "SimpleMP4",  
    "CreatedAt": 1568938240  
  }  
}
```

For more information, see [Working with AWS Elemental MediaConvert Output Presets](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [UpdatePreset](#) in *AWS CLI Command Reference*.

update-queue

The following code example shows how to use `update-queue`.

AWS CLI

To change a queue

The following `update-queue` example pauses the specified queue, by changing its status to `PAUSED`.

```
aws mediaconvert update-queue \  
--name Customer1 \  
--status PAUSED \  
--endpoint-url https://abcd1234.mediaconvert.us-west-2.amazonaws.com
```

Output:

```
{  
  "Queue": {  
    "LastUpdated": 1568839845,  
    "Status": "PAUSED",  
    "ProgressingJobsCount": 0,  
    "CreatedAt": 1526428516,  
    "Arn": "arn:aws:mediaconvert:us-west-1:123456789012:queues/Customer1",  
    "Name": "Customer1",  
    "SubmittedJobsCount": 0,  
    "PricingPlan": "ON_DEMAND",  
    "Type": "CUSTOM"  
  }  
}
```

For more information, see [Working with AWS Elemental MediaConvert Queues](#) in the *AWS Elemental MediaConvert User Guide*.

- For API details, see [UpdateQueue](#) in *AWS CLI Command Reference*.

MediaLive examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with MediaLive.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-channel

The following code example shows how to use `create-channel`.

AWS CLI

To create a channel

The following `create-channel` example creates a channel by passing in a JSON file that contains the parameters that you want to specify.

The channel in this example ingests an HLS PULL input that connects to a source that contains video, audio, and embedded captions. The channel creates one HLS output group with an Akamai server as the destination. The output group contains two outputs: one for the H.265 video and AAC audio, and one for the Web-VTT captions, in English only.

The JSON for this example channel includes the minimum required parameters for a channel that uses an HLS PULL input and that produces an HLS output group with Akamai as the destination. The JSON contains these main sections:

`InputAttachments`, which specifies one source for the audio, and one source for the captions. It does not specify a video selector, which means that MediaLive extracts the first video it finds in the source. `Destinations`, which contains the two IP addresses (URLs) for the single output group in this channel. These addresses require passwords. `EncoderSettings`, which contains subsections. `AudioDescriptions`, which specifies that the channel contains one audio output asset, which uses the source from `InputAttachments`, and produces audio in

AAC format.CaptionDescriptions, which specifies that the channel contains one captions output asset, which uses the source from InputAttachments, and produces captions in Web-VTT format.VideoDescriptions, which specifies that the channel contains one video output asset, with the specified resolution.OutputGroups, which specifies the output groups. In this example there is one group named Akamai. The connection is made using HLS PUT. The output group contains two outputs. One output is for the video asset (named Video_high) and the audio asset (named Audio_EN). One output is for the captions asset (named WebVTT_EN).

In this example, some of the parameters contain no value or contain nested empty parameters. For example, OutputSettings for the Video_and_audio output contains several nested parameters that end at an empty parameter M3u8Settings. This parameter must be included, but you can omit one, several, or all its children, which means that the child will take its default value or be null.

All the parameters that apply to this example channel but that aren't specified in this file will either take the default value, be set to null, or take a unique value generated by MediaLive.

```
aws medialive create-channel \  
  --cli-input-json file://channel-in-hls-out-hls-akamai.json
```

Contents of channel-in-hls-out-hls-akamai.json:

```
{  
  "Name": "News_West",  
  "RoleArn": "arn:aws:iam::111122223333:role/MediaLiveAccessRole",  
  "InputAttachments": [  
    {  
      "InputAttachmentName": "local_news",  
      "InputId": "1234567",  
      "InputSettings": {  
        "AudioSelectors": [  
          {  
            "Name": "English-Audio",  
            "SelectorSettings": {  
              "AudioLanguageSelection": {  
                "LanguageCode": "EN"  
              }  
            }  
          }  
        ],  
        "CaptionSelectors": [  

```



```
        {
            "LanguageCode": "ENE",
            "Name": "English_embedded"
        }
    ]
}
],
"Destinations": [
    {
        "Id": "akamai-server-west",
        "Settings": [
            {
                "PasswordParam": "/medialive/examplecorp1",
                "Url": "http://203.0.113.55/news/news_west",
                "Username": "examplecorp"
            },
            {
                "PasswordParam": "/medialive/examplecorp2",
                "Url": "http://203.0.113.82/news/news_west",
                "Username": "examplecorp"
            }
        ]
    }
],
"EncoderSettings": {
    "AudioDescriptions": [
        {
            "AudioSelectorName": "English-Audio",
            "CodecSettings": {
                "AacSettings": {}
            },
            "Name": "Audio_EN"
        }
    ],
    "CaptionDescriptions": [
        {
            "CaptionSelectorName": "English_embedded",
            "DestinationSettings": {
                "WebvttDestinationSettings": {}
            },
            "Name": "WebVTT_EN"
        }
    ]
},
```

```
"VideoDescriptions": [
  {
    "Height": 720,
    "Name": "Video_high",
    "Width": 1280
  }
],
"OutputGroups": [
  {
    "Name": "Akamai",
    "OutputGroupSettings": {
      "HlsGroupSettings": {
        "Destination": {
          "DestinationRefId": "akamai-server-west"
        },
        "HlsCdnSettings": {
          "HlsBasicPutSettings": {}
        }
      }
    },
    "Outputs": [
      {
        "AudioDescriptionNames": [
          "Audio_EN"
        ],
        "OutputName": "Video_and_audio",
        "OutputSettings": {
          "HlsOutputSettings": {
            "HlsSettings": {
              "StandardHlsSettings": {
                "M3u8Settings": {}
              }
            },
            "NameModifier": "_1"
          }
        },
        "VideoDescriptionName": "Video_high"
      },
      {
        "CaptionDescriptionNames": [
          "WebVTT_EN"
        ],
        "OutputName": "Captions-WebVTT",
        "OutputSettings": {
```

```

        "HlsOutputSettings": {
            "HlsSettings": {
                "StandardHlsSettings": {
                    "M3u8Settings": {}
                }
            },
            "NameModifier": "_2"
        }
    ]
}
],
"TimecodeConfig": {
    "Source": "EMBEDDED"
}
}
}

```

Output:

The output repeats back the contents of the JSON file, plus the following values. All parameters are ordered alphabetically.

ARN for the channel. The last part of the ARN is the unique channel ID. `EgressEndpoints` is blank in this example channel because it used only for PUSH inputs. When it applies it shows the addresses on MediaLive that content is pushed to. `OutputGroups`, `Outputs`. These show all the parameters for the output group and outputs, including those that you didn't include but that are relevant to this channel. The parameters might be empty (perhaps indicating the parameter or feature is disabled in this channel configuration) or might show the default value that will apply. `LogLevel` is set to the default (DISABLED). `Tags` is set to the default (null). `PipelinesRunningCount` and `State` show the current status of the channel.

For more information, see [Creating a Channel from Scratch](#) in the *AWS Elemental MediaLive User Guide*.

- For API details, see [CreateChannel](#) in *AWS CLI Command Reference*.

create-input

The following code example shows how to use `create-input`.

AWS CLI

To create an input

The following `create-input` example creates an HLS PULL input by passing in a JSON file that contains the parameters that apply to this type of input. The JSON for this example input specifies two sources (addresses) to the input, in order to support redundancy in the ingest. These addresses require passwords.

```
aws medialive create-input \  
  --cli-input-json file://input-hls-pull-news.json
```

Contents of `input-hls-pull-news.json`:

```
{  
  "Name": "local_news",  
  "RequestId": "cli000059",  
  "Sources": [  
    {  
      "Url": "https://203.0.113.13/newschannel/anytownusa.m3u8",  
      "Username": "examplecorp",  
      "PasswordParam": "/medialive/examplecorp1"  
    },  
    {  
      "Url": "https://198.51.100.54/fillervideos/oceanwaves.mp4",  
      "Username": "examplecorp",  
      "PasswordParam": "examplecorp2"  
    }  
  ],  
  "Type": "URL_PULL"  
}
```

Output:

The output repeats back the contents of the JSON file, plus the following values. All parameters are ordered alphabetically.

`Arn` for the input. The last part of the ARN is the unique input ID. `Attached Channels`, which is always empty for a newly created input. `Destinations`, which is empty in this example because it is used only with a PUSH input. `Id` for the input, the same as the ID in the ARN. `MediaConnectFlows`, which is empty in this example because it is used only with an input

of type `MediaConnect.SecurityGroups`, which is empty in this example because it is used only with a `PUSH` input. `State` of this input. `Tags`, which is empty (the default for this parameter).

For more information, see [Creating an Input](#) in the *AWS Elemental MediaLive User Guide*.

- For API details, see [CreateInput](#) in *AWS CLI Command Reference*.

MediaPackage examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with MediaPackage.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

`create-channel`

The following code example shows how to use `create-channel`.

AWS CLI

To create a channel

The following `create-channel` command creates a channel named `sportschannel` in the current account.

```
aws mediapackage create-channel --id sportschannel
```

Output:

```
{
  "Arn": "arn:aws:mediapackage:us-
west-2:111222333:channels/6d345804ec3f46c9b454a91d4a80d0e0",
  "HlsIngest": {
    "IngestEndpoints": [
      {
        "Id": "6d345804ec3f46c9b454a91d4a80d0e0",
        "Password": "generatedwebdavpassword1",
        "Url": "https://f31c86aed53b815a.mediapackage.us-
west-2.amazonaws.com/in/
v2/6d345804ec3f46c9b454a91d4a80d0e0/6d345804ec3f46c9b454a91d4a80d0e0/channel",
        "Username": "generatedwebdavusername1"
      },
      {
        "Id": "2daa32878af24803b24183727211b8ff",
        "Password": "generatedwebdavpassword2",
        "Url": "https://6ebbe7e04c4b0afa.mediapackage.us-
west-2.amazonaws.com/in/
v2/6d345804ec3f46c9b454a91d4a80d0e0/2daa32878af24803b24183727211b8ff/channel",
        "Username": "generatedwebdavusername2"
      }
    ]
  },
  "Id": "sportschannel",
  "Tags": {
    "region": "west"
  }
}
```

For more information, see [Creating a Channel](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [CreateChannel](#) in *AWS CLI Command Reference*.

create-origin-endpoint

The following code example shows how to use `create-origin-endpoint`.

AWS CLI

To create an origin endpoint

The following `create-origin-endpoint` command creates an origin endpoint named `cmf sports` with the package settings provided in a JSON file and specified endpoint settings.

```
aws mediapackage create-origin-endpoint \  
  --channel-id sportschannel \  
  --id cmafsports \  
  --cmf-package file:///file/path/cmafpkg.json --description "cmf output of  
sports" \  
  --id cmaf_sports \  
  --manifest-name sports_channel \  
  --startover-window-seconds 300 \  
  --tags region=west,media=sports \  
  --time-delay-seconds 10
```

Output:

```
{  
  "Arn": "arn:aws:mediapackage:us-  
west-2:111222333:origin_endpoints/1dc6718be36f4f34bb9cd86bc50925e6",  
  "ChannelId": "sportschannel",  
  "CmafPackage": {  
    "HlsManifests": [  
      {  
        "AdMarkers": "PASSTHROUGH",  
        "Id": "cmaf_sports_endpoint",  
        "IncludeIframeOnlyStream": true,  
        "ManifestName": "index",  
        "PlaylistType": "EVENT",  
        "PlaylistWindowSeconds": 300,  
        "ProgramDateTimeIntervalSeconds": 300,  
        "Url": "https://c4af3793bf76b33c.mediapackage.us-  
west-2.amazonaws.com/out/v1/1dc6718be36f4f34bb9cd86bc50925e6/cmaf_sports_endpoint/  
index.m3u8"  
      }  
    ],  
    "SegmentDurationSeconds": 2,  
    "SegmentPrefix": "sportschannel"  
  },  
  "Description": "cmf output of sports",  
  "Id": "cmaf_sports",  
  "ManifestName": "sports_channel",  
  "StartoverWindowSeconds": 300,  
  "Tags": {  
    "region": "west",  
    "media": "sports"  
  },  
}
```

```
"TimeDelaySeconds": 10,  
"Url": "",  
"Whitelist": []  
}
```

For more information, see [Creating an Endpoint](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [CreateOriginEndpoint](#) in *AWS CLI Command Reference*.

delete-channel

The following code example shows how to use `delete-channel`.

AWS CLI

To delete a channel

The following `delete-channel` command deletes the channel named `test`.

```
aws mediapackage delete-channel \  
  --id test
```

This command produces no output.

For more information, see [Deleting a Channel](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [DeleteChannel](#) in *AWS CLI Command Reference*.

delete-origin-endpoint

The following code example shows how to use `delete-origin-endpoint`.

AWS CLI

To delete an origin endpoint

The following `delete-origin-endpoint` command deletes the origin endpoint named `tester2`.

```
aws mediapackage delete-origin-endpoint \  
  --id tester2
```


For more information, see [Deleting an Endpoint](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [DeleteOriginEndpoint](#) in *AWS CLI Command Reference*.

describe-channel

The following code example shows how to use `describe-channel`.

AWS CLI

To describe a channel

The following `describe-channel` command displays all of the details of the channel named `test`.

```
aws mediapackage describe-channel \  
  --id test
```

Output:

```
{  
  "Arn": "arn:aws:mediapackage:us-  
west-2:111222333:channels/584797f1740548c389a273585dd22a63",  
  "HlsIngest": {  
    "IngestEndpoints": [  
      {  
        "Id": "584797f1740548c389a273585dd22a63",  
        "Password": "webdavgeneratedpassword1",  
        "Url": "https://9be9c4405c474882.mediapackage.us-  
west-2.amazonaws.com/in/  
v2/584797f1740548c389a273585dd22a63/584797f1740548c389a273585dd22a63/channel",  
        "Username": "webdavgeneratedusername1"  
      },  
      {  
        "Id": "7d187c8616fd455f88aaa5a9fcf74442",  
        "Password": "webdavgeneratedpassword2",  
        "Url": "https://7bf454c57220328d.mediapackage.us-  
west-2.amazonaws.com/in/  
v2/584797f1740548c389a273585dd22a63/7d187c8616fd455f88aaa5a9fcf74442/channel",  
        "Username": "webdavgeneratedusername2"  
      }  
    ]  
  },  
}
```

```
"Id": "test",
"Tags": {}
}
```

For more information, see [Viewing Channel Details](https://docs.aws.amazon.com/mediapackage/latest/ug/channels-view.html) in the *AWS Elemental MediaPackage User Guide*

- For API details, see [DescribeChannel](#) in *AWS CLI Command Reference*.

describe-origin-endpoint

The following code example shows how to use describe-origin-endpoint.

AWS CLI

To describe an origin endpoint

The following describe-origin-endpoint command displays all of the details of the origin endpoint named cmaf_sports.

```
aws mediapackage describe-origin-endpoint \
  --id cmaf_sports
```

Output:

```
{
  "Arn": "arn:aws:mediapackage:us-
west-2:111222333:origin_endpoints/1dc6718be36f4f34bb9cd86bc50925e6",
  "ChannelId": "sportschannel",
  "CmafPackage": {
    "HlsManifests": [
      {
        "AdMarkers": "NONE",
        "Id": "cmaf_sports_endpoint",
        "IncludeIframeOnlyStream": false,
        "PlaylistType": "EVENT",
        "PlaylistWindowSeconds": 60,
        "ProgramDateTimeIntervalSeconds": 0,
        "Url": "https://c4af3793bf76b33c.mediapackage.us-
west-2.amazonaws.com/out/v1/1dc6718be36f4f34bb9cd86bc50925e6/cmef_sports_endpoint/
index.m3u8"
      }
    ]
  }
}
```

```
    ],
    "SegmentDurationSeconds": 2,
    "SegmentPrefix": "sportschannel"
  },
  "Id": "cmf_sports",
  "ManifestName": "index",
  "StartoverWindowSeconds": 0,
  "Tags": {
    "region": "west",
    "media": "sports"
  },
  "TimeDelaySeconds": 0,
  "Url": "",
  "Whitelist": []
}
```

For more information, see [Viewing a Single Endpoint](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [DescribeOriginEndpoint](#) in *AWS CLI Command Reference*.

list-channels

The following code example shows how to use `list-channels`.

AWS CLI

To list all channels

The following `list-channels` command lists all of the channels that are configured on the current AWS account.

```
aws mediapackage list-channels
```

Output:

```
{
  "Channels": [
    {
      "Arn": "arn:aws:mediapackage:us-
west-2:111222333:channels/584797f1740548c389a273585dd22a63",
      "HlsIngest": {
        "IngestEndpoints": [
```

```

        {
            "Id": "584797f1740548c389a273585dd22a63",
            "Password": "webdavgeneratedpassword1",
            "Url": "https://9be9c4405c474882.mediapackage.us-
west-2.amazonaws.com/in/
v2/584797f1740548c389a273585dd22a63/584797f1740548c389a273585dd22a63/channel",
            "Username": "webdavgeneratedusername1"
        },
        {
            "Id": "7d187c8616fd455f88aaa5a9fcf74442",
            "Password": "webdavgeneratedpassword2",
            "Url": "https://7bf454c57220328d.mediapackage.us-
west-2.amazonaws.com/in/
v2/584797f1740548c389a273585dd22a63/7d187c8616fd455f88aaa5a9fcf74442/channel",
            "Username": "webdavgeneratedusername2"
        }
    ]
},
    "Id": "test",
    "Tags": {}
}
]
}

```

For more information, see [Viewing Channel Details](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [ListChannels](#) in *AWS CLI Command Reference*.

list-origin-endpoints

The following code example shows how to use `list-origin-endpoints`.

AWS CLI

To list all origin-endpoints on a channel

The following `list-origin-endpoints` command lists all of the origin endpoints that are configured on the channel named `test`.

```
aws mediapackage list-origin-endpoints \
  --channel-id test
```

Output:

```
{
  "OriginEndpoints": [
    {
      "Arn": "arn:aws:mediapackage:us-
west-2:111222333:origin_endpoints/247cff871f2845d3805129be22f2c0a2",
      "ChannelId": "test",
      "DashPackage": {
        "ManifestLayout": "FULL",
        "ManifestWindowSeconds": 60,
        "MinBufferTimeSeconds": 30,
        "MinUpdatePeriodSeconds": 15,
        "PeriodTriggers": [],
        "Profile": "NONE",
        "SegmentDurationSeconds": 2,
        "SegmentTemplateFormat": "NUMBER_WITH_TIMELINE",
        "StreamSelection": {
          "MaxVideoBitsPerSecond": 2147483647,
          "MinVideoBitsPerSecond": 0,
          "StreamOrder": "ORIGINAL"
        },
        "SuggestedPresentationDelaySeconds": 25
      },
      "Id": "tester2",
      "ManifestName": "index",
      "StartoverWindowSeconds": 0,
      "Tags": {},
      "TimeDelaySeconds": 0,
      "Url": "https://8343f7014c0ea438.mediapackage.us-west-2.amazonaws.com/
out/v1/247cff871f2845d3805129be22f2c0a2/index.mpd",
      "Whitelist": []
    },
    {
      "Arn": "arn:aws:mediapackage:us-
west-2:111222333:origin_endpoints/869e237f851549e9bcf10e3bc2830839",
      "ChannelId": "test",
      "HlsPackage": {
        "AdMarkers": "NONE",
        "IncludeIframeOnlyStream": false,
        "PlaylistType": "EVENT",
        "PlaylistWindowSeconds": 60,
        "ProgramDateTimeIntervalSeconds": 0,
        "SegmentDurationSeconds": 6,

```

```

        "StreamSelection": {
            "MaxVideoBitsPerSecond": 2147483647,
            "MinVideoBitsPerSecond": 0,
            "StreamOrder": "ORIGINAL"
        },
        "UseAudioRenditionGroup": false
    },
    "Id": "tester",
    "ManifestName": "index",
    "StartoverWindowSeconds": 0,
    "Tags": {},
    "TimeDelaySeconds": 0,
    "Url": "https://8343f7014c0ea438.mediapackage.us-west-2.amazonaws.com/
out/v1/869e237f851549e9bcf10e3bc2830839/index.m3u8",
    "Whitelist": []
}
]
}

```

For more information, see [Viewing all Endpoints Associated with a Channel](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [ListOriginEndpoints](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags assigned to a resource

The following `list-tags-for-resource` command lists the tags that are assigned to the specified resource.

```

aws mediapackage list-tags-for-resource \
  --resource-arn arn:aws:mediapackage:us-
west-2:111222333:channels/6d345804ec3f46c9b454a91d4a80d0e0

```

Output:

```
{
```

```
"Tags": {
  "region": "west"
}
```

For more information, see [Tagging Resources in AWS Elemental MediaPackage](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

rotate-ingest-endpoint-credentials

The following code example shows how to use `rotate-ingest-endpoint-credentials`.

AWS CLI

To rotate ingest credentials

The following `rotate-ingest-endpoint-credentials` command rotates the WebDAV username and password for the specified ingest endpoint.

```
aws mediapackage rotate-ingest-endpoint-credentials \
  --id test \
  --ingest-endpoint-id 584797f1740548c389a273585dd22a63
```

Output:

```
{
  "Arn": "arn:aws:mediapackage:us-west-2:111222333:channels/584797f1740548c389a273585dd22a63",
  "HlsIngest": {
    "IngestEndpoints": [
      {
        "Id": "584797f1740548c389a273585dd22a63",
        "Password": "webdavregeneratedpassword1",
        "Url": "https://9be9c4405c474882.mediapackage.us-west-2.amazonaws.com/in/v2/584797f1740548c389a273585dd22a63/584797f1740548c389a273585dd22a63/channel",
        "Username": "webdavregeneratedusername1"
      },
      {
        "Id": "7d187c8616fd455f88aaa5a9fcf74442",
        "Password": "webdavgeneratedpassword2",

```

```
        "Url": "https://7bf454c57220328d.mediapackage.us-  
west-2.amazonaws.com/in/  
v2/584797f1740548c389a273585dd22a63/7d187c8616fd455f88aaa5a9fcf74442/channel",  
        "Username": "webdavgeneratedusername2"  
    }  
]  
,  
"Id": "test",  
"Tags": {}  
}
```

For more information, see [Rotating Credentials on an Input URL](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [RotateIngestEndpointCredentials](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add a tag to a resource

The following `tag-resource` commands adds a `region=west` key and value pair to the specified resource.

```
aws mediapackage tag-resource \  
  --resource-arn arn:aws:mediapackage:us-  
west-2:111222333:channels/6d345804ec3f46c9b454a91d4a80d0e0 \  
  --tags region=west
```

This command produces no output.

For more information, see [Tagging Resources in AWS Elemental MediaPackage](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag from a resource

The following `untag-resource` command removes the tag with the key `region` from the specified channel.

```
aws mediapackage untag-resource \  
  --resource-arn arn:aws:mediapackage:us-  
west-2:111222333:channels/6d345804ec3f46c9b454a91d4a80d0e0 \  
  --tag-keys region
```

For more information, see [Tagging Resources in AWS Elemental MediaPackage](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-channel

The following code example shows how to use `update-channel`.

AWS CLI

To update a channel

The following `update-channel` command updates the channel named `sportschannel` to include the description `24x7 sports`.

```
aws mediapackage update-channel \  
  --id sportschannel \  
  --description "24x7 sports"
```

Output:

```
{  
  "Arn": "arn:aws:mediapackage:us-  
west-2:111222333:channels/6d345804ec3f46c9b454a91d4a80d0e0",  
  "Description": "24x7 sports",  
  "HlsIngest": {  
    "IngestEndpoints": [  
      {  
        "Id": "6d345804ec3f46c9b454a91d4a80d0e0",  
        "Password": "generatedwebdavpassword1",
```

```

        "Url": "https://f31c86aed53b815a.mediapackage.us-
west-2.amazonaws.com/in/
v2/6d345804ec3f46c9b454a91d4a80d0e0/6d345804ec3f46c9b454a91d4a80d0e0/channel",
        "Username": "generatedwebdavusername1"
    },
    {
        "Id": "2daa32878af24803b24183727211b8ff",
        "Password": "generatedwebdavpassword2",
        "Url": "https://6ebbe7e04c4b0afa.mediapackage.us-
west-2.amazonaws.com/in/
v2/6d345804ec3f46c9b454a91d4a80d0e0/2daa32878af24803b24183727211b8ff/channel",
        "Username": "generatedwebdavusername2"
    }
]
},
"Id": "sportschannel",
"Tags": {}
}

```

For more information, see [Editing a Channel](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [UpdateChannel](#) in *AWS CLI Command Reference*.

update-origin-endpoint

The following code example shows how to use update-origin-endpoint.

AWS CLI

To update an origin endpoint

The following update-origin-endpoint command updates the origin endpoint named `cmf_sports`. It changes the time delay to 0 seconds.

```

aws mediapackage update-origin-endpoint \
  --id cmf_sports \
  --time-delay-seconds 0

```

Output:

```

{
  "Arn": "arn:aws:mediapackage:us-
west-2:111222333:origin_endpoints/1dc6718be36f4f34bb9cd86bc50925e6",

```

```
"ChannelId": "sportschannel",
"CmafPackage": {
  "HlsManifests": [
    {
      "AdMarkers": "NONE",
      "Id": "cmaf_sports_endpoint",
      "IncludeIframeOnlyStream": false,
      "PlaylistType": "EVENT",
      "PlaylistWindowSeconds": 60,
      "ProgramDateTimeIntervalSeconds": 0,
      "Url": "https://c4af3793bf76b33c.mediapackage.us-
west-2.amazonaws.com/out/v1/1dc6718be36f4f34bb9cd86bc50925e6/cmaf_sports_endpoint/
index.m3u8"
    }
  ],
  "SegmentDurationSeconds": 2,
  "SegmentPrefix": "sportschannel"
},
"Id": "cmaf_sports",
"ManifestName": "index",
"StartoverWindowSeconds": 0,
"Tags": {
  "region": "west",
  "media": "sports"
},
"TimeDelaySeconds": 0,
"Url": "",
"Whitelist": []
}
```

For more information, see [Editing an Endpoint](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [UpdateOriginEndpoint](#) in *AWS CLI Command Reference*.

MediaPackage VOD examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with MediaPackage VOD.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-asset

The following code example shows how to use `create-asset`.

AWS CLI

To create an asset

The following `create-asset` example creates an asset named `Chicken_Asset` in the current AWS account. The asset ingests the file `30sec_chicken.smil` to `MediaPackage`.

```
aws mediapackage-vod create-asset \  
  --id chicken_asset \  
  --packaging-group-id hls_chicken_gp \  
  --source-role-arn arn:aws:iam::111122223333:role/EMP_Vod \  
  --source-arn arn:aws:s3::111122223333:video-bucket/A/30sec_chicken.smil
```

Output:

```
{  
  "Arn": "arn:aws:mediapackage-vod:us-west-2:111122223333:assets/chicken_asset",  
  "Id": "chicken_asset",  
  "PackagingGroupId": "hls_chicken_gp",  
  "SourceArn": "arn:aws:s3::111122223333:video-bucket/A/30sec_chicken.smil",  
  "SourceRoleArn": "arn:aws:iam::111122223333:role/EMP_Vod",  
  "EgressEndpoints": [  
    {  
      "PackagingConfigurationId": "New_config_1",  
      "Url": "https://c75ea2668ab49d02bca7ae10ef31c59e.egress.mediapackage-  
vod.us-west-2.amazonaws.com/out/"
```

```
v1/6644b55df1744261ab3732a8e5cdaf07/904b06a58c7645e08d57d40d064216ac/
f5b2e633ff4942228095d164c10074f3/index.m3u8"
    },
    {
      "PackagingConfigurationId": "new_hls",
      "Url": " https://c75ea2668ab49d02bca7ae10ef31c59e.egress.mediapackage-
vod.us-west-2.amazonaws.com/out/v1/6644b55df1744261ab3732a8e5cdaf07/
fe8f1f00a80e424cb4f8da4095835e9e/7370ec57432343af816332356d2bd5c6/string.m3u8"
    }
  ]
}
```

For more information, see [Ingest an Asset](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [CreateAsset](#) in *AWS CLI Command Reference*.

create-packaging-configuration

The following code example shows how to use `create-packaging-configuration`.

AWS CLI

To create a packaging configuration

The following `create-packaging-configuration` example creates a packaging configuration named `new_hls` in the packaging group named `hls_chicken`. This example uses a file on disk named `hls_pc.json` to provide the details.

```
aws mediapackage-vod create-packaging-configuration \
  --id new_hls \
  --packaging-group-id hls_chicken \
  --hls-package file://hls_pc.json
```

Contents of `hls_pc.json`:

```
{
  "HlsManifests": [
    {
      "AdMarkers": "NONE",
      "IncludeIframeOnlyStream": false,
      "ManifestName": "string",
      "ProgramDateTimeIntervalSeconds": 60,
```

```

        "RepeatExtXKey":true,
        "StreamSelection":{
            "MaxVideoBitsPerSecond":1000,
            "MinVideoBitsPerSecond":0,
            "StreamOrder":"ORIGINAL"
        }
    ],
    "SegmentDurationSeconds":6,
    "UseAudioRenditionGroup":false
}

```

Output:

```

{
  "Arn":"arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-configurations/new_hls",
  "Id":"new_hls",
  "PackagingGroupId":"hls_chicken",
  "HlsManifests":{
    "SegmentDurationSeconds":6,
    "UseAudioRenditionGroup":false,
    "HlsMarkers":[
      {
        "AdMarkers":"NONE",
        "IncludeIframeOnlyStream":false,
        "ManifestName":"string",
        "ProgramDateTimeIntervalSeconds":60,
        "RepeatExtXKey":true,
        "StreamSelection":{
          "MaxVideoBitsPerSecond":1000,
          "MinVideoBitsPerSecond":0,
          "StreamOrder":"ORIGINAL"
        }
      }
    ]
  }
}

```

For more information, see [Creating a Packaging Configuration](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [CreatePackagingConfiguration](#) in *AWS CLI Command Reference*.

create-packaging-group

The following code example shows how to use `create-packaging-group`.

AWS CLI

To create a packaging group

The following `create-packaging-group` example lists all of the packaging groups that are configured in the current AWS account.

```
aws mediapackage-vod create-packaging-group \  
  --id hls_chicken
```

Output:

```
{  
  "Arn": "arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-groups/  
hls_chicken",  
  "Id": "hls_chicken"  
}
```

For more information, see [Creating a Packaging Group](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [CreatePackagingGroup](#) in *AWS CLI Command Reference*.

delete-asset

The following code example shows how to use `delete-asset`.

AWS CLI

To delete an asset

The following `delete-asset` example deletes the asset named `30sec_chicken`.

```
aws mediapackage-vod delete-asset \  
  --id 30sec_chicken
```

This command produces no output.

For more information, see [Deleting an Asset](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [DeleteAsset](#) in *AWS CLI Command Reference*.

delete-packaging-configuration

The following code example shows how to use `delete-packaging-configuration`.

AWS CLI

To delete a packaging configuration

The following `delete-packaging-configuration` example deletes the packaging configuration named CMAF.

```
aws mediapackage-vod delete-packaging-configuration \  
  --id CMAF
```

This command produces no output.

For more information, see [Deleting a Packaging Configuration](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [DeletePackagingConfiguration](#) in *AWS CLI Command Reference*.

delete-packaging-group

The following code example shows how to use `delete-packaging-group`.

AWS CLI

To delete a packaging group

The following `delete-packaging-group` example deletes the packaging group named Dash_widevine.

```
aws mediapackage-vod delete-packaging-group \  
  --id Dash_widevine
```

This command produces no output.

For more information, see [Deleting a Packaging Group](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [DeletePackagingGroup](#) in *AWS CLI Command Reference*.

describe-asset

The following code example shows how to use describe-asset.

AWS CLI

To describe an asset

The following describe-asset example displays all of the details of the asset named 30sec_chicken.

```
aws mediapackage-vod describe-asset \  
  --id 30sec_chicken
```

Output:

```
{  
  "Arn": "arn:aws:mediapackage-vod:us-west-2:111122223333:assets/30sec_chicken",  
  "Id": "30sec_chicken",  
  "PackagingGroupId": "Packaging_group_1",  
  "SourceArn": "arn:aws:s3::111122223333:video-bucket/A/30sec_chicken.smil",  
  "SourceRoleArn": "arn:aws:iam::111122223333:role/EMP_Vod",  
  "EgressEndpoints": [  
    {  
      "PackagingConfigurationId": "DASH",  
      "Url": "https://a5f46a44118ba3e3724ef39ef532e701.egress.mediapackage-  
vod.us-west-2.amazonaws.com/out/v1/  
aad7962c569946119c2d5a691be5663c/66c25aff456d463aae0855172b3beb27/4ddfda6da17c4c279a1b8401cb  
index.mpd"  
    },  
    {  
      "PackagingConfigurationId": "HLS",  
      "Url": "https://a5f46a44118ba3e3724ef39ef532e701.egress.mediapackage-  
vod.us-west-2.amazonaws.com/out/v1/  
aad7962c569946119c2d5a691be5663c/6e5bf286a3414254a2bf0d22ae148d7e/06b5875b4d004c3cbdc4da2dc4  
index.m3u8"  
    },  
    {  
      "PackagingConfigurationId": "CMAF",  
      "Url": "https://a5f46a44118ba3e3724ef39ef532e701.egress.mediapackage-  
vod.us-west-2.amazonaws.com/out/v1/  
aad7962c569946119c2d5a691be5663c/628fb5d8d89e4702958b020af27fde0e/05eb062214064238ad6330a443  
index.m3u8"  }  
]
```

```
    }  
  ]  
}
```

For more information, see [Viewing Asset Details](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [DescribeAsset](#) in *AWS CLI Command Reference*.

describe-packaging-configuration

The following code example shows how to use `describe-packaging-configuration`.

AWS CLI

To describe a packaging configuration

The following `describe-packaging-configuration` example displays all of the details of the packaging configuration named DASH.

```
aws mediapackage-vod describe-packaging-configuration \  
  --id DASH
```

Output:

```
{  
  "Arn": "arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-configurations/  
DASH",  
  "Id": "DASH",  
  "PackagingGroupId": "Packaging_group_1",  
  "DashPackage": [  
    {  
      "SegmentDurationSeconds": "2"  
    },  
    {  
      "DashManifests": {  
        "ManifestName": "index",  
        "MinBufferTimeSeconds": "30",  
        "Profile": "NONE"  
      }  
    }  
  ]  
}
```

For more information, see [Viewing Packaging Configuration Details](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [DescribePackagingConfiguration](#) in *AWS CLI Command Reference*.

describe-packaging-group

The following code example shows how to use `describe-packaging-group`.

AWS CLI

To describe a packaging group

The following `describe-packaging-group` example displays all of the details of the packaging group named `Packaging_group_1`.

```
aws mediapackage-vod describe-packaging-group \
  --id Packaging_group_1
```

Output:

```
{
  "Arn": "arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-groups/
Packaging_group_1",
  "Id": "Packaging_group_1"
}
```

For more information, see [Viewing Packaging Group Details](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [DescribePackagingGroup](#) in *AWS CLI Command Reference*.

list-assets

The following code example shows how to use `list-assets`.

AWS CLI

To list all assets

The following `list-assets` example lists all of the assets that are configured in the current AWS account.

```
aws mediapackage-vod list-assets
```

Output:

```
{
  "Assets": [
    {
      "Arn": "arn:aws:mediapackage-vod:us-west-2:111122223333:assets/30sec_chicken",
      "Id": "30sec_chicken",
      "PackagingGroupId": "Packaging_group_1",
      "SourceArn": "arn:aws:s3::111122223333:video-bucket/A/30sec_chicken.smil",
      "SourceRoleArn": "arn:aws:iam::111122223333:role/EMP_Vod"
    }
  ]
}
```

For more information, see [Viewing Asset Details](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [ListAssets](#) in *AWS CLI Command Reference*.

list-packaging-configurations

The following code example shows how to use `list-packaging-configurations`.

AWS CLI

To list all packaging configurations

The following `list-packaging-configurations` example lists all of the packaging configurations that are configured on the packaging group named `Packaging_group_1`.

```
aws mediapackage-vod list-packaging-configurations \
  --packaging-group-id Packaging_group_1
```

Output:

```
{
  "PackagingConfigurations": [
    {
      "Arn": "arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-configurations/CMAF",
      "Id": "CMAF",
      "PackagingGroupId": "Packaging_group_1",
    }
  ]
}
```

```

    "CmafPackage":[
      {
        "SegmentDurationSeconds":"2"
      },
      {
        "HlsManifests":{
          "AdMarkers":"NONE",
          "RepeatExtXKey":"False",
          "ManifestName":"index",
          "ProgramDateTimeIntervalSeconds":"0",
          "IncludeIframeOnlyStream":"False"
        }
      }
    ],
    {
      "Arn":"arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-
configurations/DASH",
      "Id":"DASH",
      "PackagingGroupId":"Packaging_group_1",
      "DashPackage":[
        {
          "SegmentDurationSeconds":"2"
        },
        {
          "DashManifests":{
            "ManifestName":"index",
            "MinBufferTimeSeconds":"30",
            "Profile":"NONE"
          }
        }
      ]
    },
    {
      "Arn":"arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-
configurations/HLS",
      "Id":"HLS",
      "PackagingGroupId":"Packaging_group_1",
      "HlsPackage":[
        {
          "SegmentDurationSeconds":"6",
          "UseAudioRenditionGroup":"False"
        },
        {

```

```

        "HlsManifests":{
            "AdMarkers":"NONE",
            "RepeatExtXKey":"False",
            "ManifestName":"index",
            "ProgramDateTimeIntervalSeconds":"0",
            "IncludeIframeOnlyStream":"False"
        }
    ]
},
{
    "Arn":"arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-
configurations/New_config_0_copy",
    "Id":"New_config_0_copy",
    "PackagingGroupId":"Packaging_group_1",
    "HlsPackage":[
        {
            "SegmentDurationSeconds":"6",
            "UseAudioRenditionGroup":"False"
        },
        {
            "Encryption":{
                "EncryptionMethod":"AWS_128",
                "SpekeKeyProvider":{
                    "RoleArn":"arn:aws:iam:111122223333::role/SPEKERole",
                    "Url":"https://lfgubdvs97.execute-api.us-
west-2.amazonaws.com/EkeStage/copyProtection/",
                    "SystemIds":[
                        "81376844-f976-481e-a84e-cc25d39b0b33"
                    ]
                }
            }
        },
        {
            "HlsManifests":{
                "AdMarkers":"NONE",
                "RepeatExtXKey":"False",
                "ManifestName":"index",
                "ProgramDateTimeIntervalSeconds":"0",
                "IncludeIframeOnlyStream":"False"
            }
        }
    ]
}
}

```

```
]
}
```

For more information, see [Viewing Packaging Configuration Details](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [ListPackagingConfigurations](#) in *AWS CLI Command Reference*.

list-packaging-groups

The following code example shows how to use `list-packaging-groups`.

AWS CLI

To list all packaging groups

The following `list-packaging-groups` example lists all of the packaging groups that are configured in the current AWS account.

```
aws mediapackage-vod list-packaging-groups
```

Output:

```
{
  "PackagingGroups": [
    {
      "Arn": "arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-
groups/Dash_widevine",
      "Id": "Dash_widevine"
    },
    {
      "Arn": "arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-
groups/Encrypted_HLS",
      "Id": "Encrypted_HLS"
    },
    {
      "Arn": "arn:aws:mediapackage-vod:us-west-2:111122223333:packaging-
groups/Packaging_group_1",
      "Id": "Packaging_group_1"
    }
  ]
}
```

For more information, see [Viewing Packaging Group Details](#) in the *AWS Elemental MediaPackage User Guide*.

- For API details, see [ListPackagingGroups](#) in *AWS CLI Command Reference*.

MediaStore Data Plane examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with MediaStore Data Plane.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

delete-object

The following code example shows how to use `delete-object`.

AWS CLI

To delete an object

The following `delete-object` example deletes the specified object.

```
aws mediastore-data delete-object \  
  --endpoint=https://aaabbbcccddee.data.mediastore.us-west-2.amazonaws.com \  
  --path=/folder_name/README.md
```

This command produces no output.

For more information, see [Deleting an Object](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [DeleteObject](#) in *AWS CLI Command Reference*.

describe-object

The following code example shows how to use describe-object.

AWS CLI

To view the headers for an object

The following describe-object example displays the headers for an object at the specified path.

```
aws mediastore-data describe-object \  
  --endpoint https://aaabbbccdddee.data.mediastore.us-west-2.amazonaws.com \  
  --path events/baseball/setup.jpg
```

Output:

```
{  
  "LastModified": "Fri, 19 Jul 2019 21:50:31 GMT",  
  "ContentType": "image/jpeg",  
  "ContentLength": "3860266",  
  "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9e4dd89ff7f5555555555555555da6d3"  
}
```

For more information, see [Viewing the Details of an Object](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [DescribeObject](#) in *AWS CLI Command Reference*.

get-object

The following code example shows how to use get-object.

AWS CLI

Example 1: To download an entire object

The following get-object example downloads the specified object.

```
aws mediastore-data get-object \  
  --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com \  
  --path events/baseball/setup.jpg setup.jpg
```

Output:

```
{  
  "ContentType": "image/jpeg",  
  "StatusCode": 200,  
  "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9e999999999994dd89ff7f555555555555da6d3",  
  "ContentLength": "3860266",  
  "LastModified": "Fri, 19 Jul 2019 21:50:31 GMT"  
}
```

Example 2: To download part of an object

The following `get-object` example downloads the specified part of an object.

```
aws mediastore-data get-object \  
  --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com \  
  --path events/baseball/setup.jpg setup.jpg \  
  --range "bytes=0-100"
```

Output:

```
{  
  "StatusCode": 206,  
  "LastModified": "Fri, 19 Jul 2019 21:50:31 GMT",  
  "ContentType": "image/jpeg",  
  "ContentRange": "bytes 0-100/3860266",  
  "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9e999999999994dd89ff7f555555555555da6d3",  
  "ContentLength": "101"  
}
```

For more information, see [Downloading an Object](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [GetObject](#) in *AWS CLI Command Reference*.

list-items

The following code example shows how to use `list-items`.

AWS CLI

Example 1: To view a list of items (objects and folders) stored in a container

The following `list-items` example displays a list of items (objects and folders) stored in the specified container.

```
aws mediastore-data list-items \  
--endpoint https://aaabbbccdddee.data.mediastore.us-west-2.amazonaws.com
```

Output:

```
{  
  "Items": [  
    {  
      "Type": "OBJECT",  
      "ContentLength": 3784,  
      "Name": "setup.jpg",  
      "ETag":  
      "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f555555555555da6d3",  
      "ContentType": "image/jpeg",  
      "LastModified": 1563571859.379  
    },  
    {  
      "Type": "FOLDER",  
      "Name": "events"  
    }  
  ]  
}
```

Example 2: To view a list of items (objects and folders) stored in a folder

The following `list-items` example displays a list of items (objects and folders) stored in the specified folder.

```
aws mediastore-data list-items \  
--endpoint https://aaabbbccdddee.data.mediastore.us-west-2.amazonaws.com \  
--path events/baseball
```

Output:

```
{
```

```

    "Items": [
      {
        "ETag":
"2aa333bbcc8d8d22d777e999c88d4aa9e4dd89ff7f5555555555555555da6d3",
        "ContentType": "image/jpeg",
        "Type": "OBJECT",
        "ContentLength": 3860266,
        "LastModified": 1563573031.872,
        "Name": "setup.jpg"
      }
    ]
  }
}

```

For more information, see [Viewing a List of Objects](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [ListItems](#) in *AWS CLI Command Reference*.

put-object

The following code example shows how to use `put-object`.

AWS CLI

Example 1: To upload an object to a container

The following `put-object` example upload an object to the specified container.

```

aws mediastore-data put-object \
  --endpoint https://aaabbbccdddee.data.mediastore.us-west-2.amazonaws.com \
  --body README.md \
  --path README.md \
  --cache-control "max-age=6, public" \
  --content-type binary/octet-stream

```

Output:

```

{
  "ContentSHA256":
"f29bc64a9d3732b4b9035125fdb3285f5b6455778edca72414671e0ca3b2e0de",
  "StorageClass": "TEMPORAL",
  "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9e4dd89ff7f5555555555555555da6d3"
}

```

```
}
```

Example 2: To upload an object to a folder within a container

The following `put-object` example upload an object to the specified folder within a container.

```
aws mediastore-data put-object \  
  --endpoint https://aaabbbccdddee.data.mediastore.us-west-2.amazonaws.com \  
  --body ReadMe.md \  
  --path /september-events/ReadMe.md \  
  --cache-control "max-age=6, public" \  
  --content-type binary/octet-stream
```

Output:

```
{  
  "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f555555555555da6d3",  
  "ContentSHA256":  
  "f29bc64a9d3732b4b9035125fdb3285f5b6455778edca72414671e0ca3b2e0de",  
  "StorageClass": "TEMPORAL"  
}
```

For more information, see [Uploading an Object](#) in the *AWS Elemental MediaStore User Guide*.

- For API details, see [PutObject](#) in *AWS CLI Command Reference*.

MediaTailor examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with MediaTailor.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

delete-playback-configuration

The following code example shows how to use `delete-playback-configuration`.

AWS CLI

To delete a configuration

The following `delete-playback-configuration` deletes a configuration named `campaign_short`.

```
aws mediatailor delete-playback-configuration \  
  --name campaign_short
```

This command produces no output.

For more information, see [Deleting a Configuration](#) in the *AWS Elemental MediaTailor User Guide*.

- For API details, see [DeletePlaybackConfiguration](#) in *AWS CLI Command Reference*.

get-playback-configuration

The following code example shows how to use `get-playback-configuration`.

AWS CLI

To describe a configuration

The following `get-playback-configuration` displays all of the details of the configuration named `west_campaign`.

```
aws mediatailor get-playback-configuration \  
  --name west_campaign
```

Output:

```
{
  "AdDecisionServerUrl": "http://your.ads.url",
  "CdnConfiguration": {},
  "DashConfiguration": {
    "ManifestEndpointPrefix":
    "https://170c14299689462897d0cc45fc2000bb.mediatailor.us-west-2.amazonaws.com/v1/
dash/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/west_campaign/",
    "MpdLocation": "EMT_DEFAULT",
    "OriginManifestType": "MULTI_PERIOD"
  },
  "HlsConfiguration": {
    "ManifestEndpointPrefix":
    "https://170c14299689462897d0cc45fc2000bb.mediatailor.us-west-2.amazonaws.com/v1/
master/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/west_campaign/"
  },
  "Name": "west_campaign",
  "PlaybackConfigurationArn": "arn:aws:mediatailor:us-
west-2:123456789012:playbackConfiguration/west_campaign",
  "PlaybackEndpointPrefix":
  "https://170c14299689462897d0cc45fc2000bb.mediatailor.us-west-2.amazonaws.com",
  "SessionInitializationEndpointPrefix":
  "https://170c14299689462897d0cc45fc2000bb.mediatailor.us-west-2.amazonaws.com/v1/
session/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/west_campaign/",
  "Tags": {},
  "VideoContentSourceUrl": "https://8343f7014c0ea438.mediapackage.us-
west-2.amazonaws.com/out/v1/683f0f2ff7cd43a48902e6dcd5e16dcf/index.m3u8"
}
```

For more information, see [Viewing a Configuration](#) in the *AWS Elemental MediaTailor User Guide*.

- For API details, see [GetPlaybackConfiguration](#) in *AWS CLI Command Reference*.

list-playback-configurations

The following code example shows how to use `list-playback-configurations`.

AWS CLI

To list all configurations

The following `list-playback-configurations` displays all of the details of the configuration on the current AWS account.

```
aws mediatailor list-playback-configurations
```

Output:

```
{
  "Items": [
    {
      "AdDecisionServerUrl": "http://your.ads.url",
      "CdnConfiguration": {},
      "DashConfiguration": {
        "ManifestEndpointPrefix":
"https://170c14299689462897d0cc45fc2000bb.mediatailor.us-west-2.amazonaws.com/v1/dash/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/west_campaign/",
        "MpdLocation": "EMT_DEFAULT",
        "OriginManifestType": "MULTI_PERIOD"
      },
      "HlsConfiguration": {
        "ManifestEndpointPrefix":
"https://170c14299689462897d0cc45fc2000bb.mediatailor.us-west-2.amazonaws.com/v1/master/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/west_campaign/"
      },
      "Name": "west_campaign",
      "PlaybackConfigurationArn": "arn:aws:mediatailor:us-west-2:123456789012:playbackConfiguration/west_campaign",
      "PlaybackEndpointPrefix":
"https://170c14299689462897d0cc45fc2000bb.mediatailor.us-west-2.amazonaws.com",
      "SessionInitializationEndpointPrefix":
"https://170c14299689462897d0cc45fc2000bb.mediatailor.us-west-2.amazonaws.com/v1/session/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/west_campaign/",
      "Tags": {},
      "VideoContentSourceUrl": "https://8343f7014c0ea438.mediapackage.us-west-2.amazonaws.com/out/v1/683f0f2ff7cd43a48902e6dcd5e16dcf/index.m3u8"
    },
    {
      "AdDecisionServerUrl": "http://your.ads.url",
      "CdnConfiguration": {},
      "DashConfiguration": {
        "ManifestEndpointPrefix":
"https://73511f91d6a24ca2b93f3cf1d7cedd67.mediatailor.us-west-2.amazonaws.com/v1/dash/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/sports_campaign/",
        "MpdLocation": "DISABLED",
        "OriginManifestType": "MULTI_PERIOD"
      },
    },
  ],
}
```



```

    "HlsConfiguration": {
      "ManifestEndpointPrefix":
        "https://73511f91d6a24ca2b93f3cf1d7cedd67.mediatailor.us-west-2.amazonaws.com/v1/
        master/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/sports_campaign/"
      },
      "Name": "sports_campaign",
      "PlaybackConfigurationArn": "arn:aws:mediatailor:us-
west-2:123456789012:playbackConfiguration/sports_campaign",
      "PlaybackEndpointPrefix":
        "https://73511f91d6a24ca2b93f3cf1d7cedd67.mediatailor.us-west-2.amazonaws.com",
      "SessionInitializationEndpointPrefix":
        "https://73511f91d6a24ca2b93f3cf1d7cedd67.mediatailor.us-west-2.amazonaws.com/v1/
        session/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/sports_campaign/",
      "SlateAdUrl": "http://s3.bucket/slate_ad.mp4",
      "Tags": {},
      "VideoContentSourceUrl": "https://c4af3793bf76b33c.mediapackage.us-
west-2.amazonaws.com/out/v1/1dc6718be36f4f34bb9cd86bc50925e6/sports_endpoint/
        index.m3u8"
    }
  ]
}

```

For more information, see [Viewing a Configuration](https://docs.aws.amazon.com/mediatailor/latest/ug/configurations-view.html) in the *AWS Elemental MediaTailor User Guide*.

- For API details, see [ListPlaybackConfigurations](#) in *AWS CLI Command Reference*.

put-playback-configuration

The following code example shows how to use `put-playback-configuration`.

AWS CLI

To create a configuration

The following `put-playback-configuration` creates a configuration named `campaign_short`.

```

aws mediatailor put-playback-configuration \
  --name campaign_short \
  --ad-decision-server-url http://your.ads.url \
  --video-content-source-url http://video.bucket/index.m3u8

```

Output:

```
{
  "AdDecisionServerUrl": "http://your.ads.url",
  "CdnConfiguration": {},
  "DashConfiguration": {
    "ManifestEndpointPrefix":
      "https://13484114d38f4383bc0d6a7cb879bd00.mediatailor.us-west-2.amazonaws.com/v1/
dash/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/campaign_short/",
    "MpdLocation": "EMT_DEFAULT",
    "OriginManifestType": "MULTI_PERIOD"
  },
  "HlsConfiguration": {
    "ManifestEndpointPrefix":
      "https://13484114d38f4383bc0d6a7cb879bd00.mediatailor.us-west-2.amazonaws.com/v1/
master/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/campaign_short/"
  },
  "Name": "campaign_short",
  "PlaybackConfigurationArn": "arn:aws:mediatailor:us-
west-2:123456789012:playbackConfiguration/campaign_short",
  "PlaybackEndpointPrefix":
    "https://13484114d38f4383bc0d6a7cb879bd00.mediatailor.us-west-2.amazonaws.com",
  "SessionInitializationEndpointPrefix":
    "https://13484114d38f4383bc0d6a7cb879bd00.mediatailor.us-west-2.amazonaws.com/v1/
session/1cbfeaaecb69778e0c167d0505a2bc57da2b1754/campaign_short/",
  "Tags": {},
  "VideoContentSourceUrl": "http://video.bucket/index.m3u8"
}
```

For more information, see [Creating a Configuration](#) in the *AWS Elemental MediaTailor User Guide*.

- For API details, see [PutPlaybackConfiguration](#) in *AWS CLI Command Reference*.

MemoryDB examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with MemoryDB.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

copy-snapshot

The following code example shows how to use `copy-snapshot`.

AWS CLI

To copy a snapshot

The following `copy-snapshot` example creates a copy of a snapshot.

```
aws memorydb copy-snapshot \  
  --source-snapshot-name my-cluster-snapshot \  
  --target-snapshot-name my-cluster-snapshot-copy
```

Output

```
{  
  "Snapshot": {  
    "Name": "my-cluster-snapshot-copy",  
    "Status": "creating",  
    "Source": "manual",  
    "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:snapshot/my-cluster-snapshot-copy",  
    "ClusterConfiguration": {  
      "Name": "my-cluster",  
      "Description": " ",  
      "NodeType": "db.r6g.large",  
      "EngineVersion": "6.2",  
      "MaintenanceWindow": "wed:03:00-wed:04:00",  
      "Port": 6379,  
    }  
  }  
}
```

```
        "ParameterGroupName": "default.memorydb-redis6",
        "SubnetGroupName": "my-sg",
        "VpcId": "vpc-xx2574fc",
        "SnapshotRetentionLimit": 0,
        "SnapshotWindow": "04:30-05:30",
        "NumShards": 2
    }
}
```

For more information, see [Copying a snapshot](#) in the *MemoryDB User Guide*.

- For API details, see [CopySnapshot](#) in *AWS CLI Command Reference*.

create-acl

The following code example shows how to use `create-acl`.

AWS CLI

To create an ACL

The following `create-acl` example creates a new Access control list.

```
aws memorydb create-acl \
  --acl-name "new-acl-1" \
  --user-names "my-user"
```

Output:

```
{
  "ACL": {
    "Name": "new-acl-1",
    "Status": "creating",
    "UserNames": [
      "my-user"
    ],
    "MinimumEngineVersion": "6.2",
    "Clusters": [],
    "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:acl/new-acl-1"
  }
}
```

For more information, see [Authenticating users with Access Control Lists](#) in the *MemoryDB User Guide*.

- For API details, see [CreateAcl](#) in *AWS CLI Command Reference*.

create-cluster

The following code example shows how to use `create-cluster`.

AWS CLI

To create a cluster

The following `create-cluster` example creates a new cluster.

```
aws memorydb create-cluster \  
  --cluster-name my-new-cluster \  
  --node-type db.r6g.large \  
  --acl-name my-acl \  
  --subnet-group my-sg
```

Output:

```
{  
  "Cluster": {  
    "Name": "my-new-cluster",  
    "Status": "creating",  
    "NumberOfShards": 1,  
    "AvailabilityMode": "MultiAZ",  
    "ClusterEndpoint": {  
      "Port": 6379  
    },  
    "NodeType": "db.r6g.large",  
    "EngineVersion": "6.2",  
    "EnginePatchVersion": "6.2.6",  
    "ParameterGroupName": "default.memorydb-redis6",  
    "ParameterGroupStatus": "in-sync",  
    "SubnetGroupName": "my-sg",  
    "TLSEnabled": true,  
    "ARN": "arn:aws:memorydb:us-east-1:49165xxxxxx:cluster/my-new-cluster",  
    "SnapshotRetentionLimit": 0,  
    "MaintenanceWindow": "sat:10:00-sat:11:00",  
    "SnapshotWindow": "07:30-08:30",
```

```
    "ACLName": "my-acl",
    "AutoMinorVersionUpgrade": true
  }
}
```

For more information, see [Managing Clusters](#) in the *MemoryDB User Guide*.

- For API details, see [CreateCluster](#) in *AWS CLI Command Reference*.

create-parameter-group

The following code example shows how to use `create-parameter-group`.

AWS CLI

To create a parameter group

The following `create-parameter-group` example creates a parameter group.

```
aws memorydb create-parameter-group \
  --parameter-group-name myRedis6x \
  --family memorydb_redis6 \
  --description "my-parameter-group"
```

Output:

```
{
  "ParameterGroup": {
    "Name": "myredis6x",
    "Family": "memorydb_redis6",
    "Description": "my-parameter-group",
    "ARN": "arn:aws:memorydb:us-east-1:49165xxxxxx:parametergroup/myredis6x"
  }
}
```

For more information, see [Creating a parameter group](#) in the *MemoryDB User Guide*.

- For API details, see [CreateParameterGroup](#) in *AWS CLI Command Reference*.

create-snapshot

The following code example shows how to use `create-snapshot`.

AWS CLI

To create a snapshot

The following `create-snapshot` example creates a snapshot.

```
aws memorydb create-snapshot \  
  --cluster-name my-cluster \  
  --snapshot-name my-cluster-snapshot
```

Output:

```
{  
  "Snapshot": {  
    "Name": "my-cluster-snapshot1",  
    "Status": "creating",  
    "Source": "manual",  
    "ARN": "arn:aws:memorydb:us-east-1:49165xxxxxx:snapshot/my-cluster-  
snapshot",  
    "ClusterConfiguration": {  
      "Name": "my-cluster",  
      "Description": "",  
      "NodeType": "db.r6g.large",  
      "EngineVersion": "6.2",  
      "MaintenanceWindow": "wed:03:00-wed:04:00",  
      "Port": 6379,  
      "ParameterGroupName": "default.memorydb-redis6",  
      "SubnetGroupName": "my-sg",  
      "VpcId": "vpc-862xxxxc",  
      "SnapshotRetentionLimit": 0,  
      "SnapshotWindow": "04:30-05:30",  
      "NumShards": 2  
    }  
  }  
}
```

For more information, see [Making manual snapshots](#) in the *MemoryDB User Guide*.

- For API details, see [CreateSnapshot](#) in *AWS CLI Command Reference*.

create-subnet-group

The following code example shows how to use `create-subnet-group`.

AWS CLI

To create a subnet group

The following `create-subnet-group` example creates a subnet group.

```
aws memorydb create-subnet-group \  
  --subnet-group-name mysubnetgroup \  
  --description "my subnet group" \  
  --subnet-ids subnet-5623xxxx
```

Output:

```
{  
  "SubnetGroup": {  
    "Name": "mysubnetgroup",  
    "Description": "my subnet group",  
    "VpcId": "vpc-86257xxx",  
    "Subnets": [  
      {  
        "Identifier": "subnet-5623xxxx",  
        "AvailabilityZone": {  
          "Name": "us-east-1a"  
        }  
      }  
    ],  
    "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:subnetgroup/mysubnetgroup"  
  }  
}
```

For more information, see [Creating a subnet group](#) in the *MemoryDB User Guide*.

- For API details, see [CreateSubnetGroup](#) in *AWS CLI Command Reference*.

create-user

The following code example shows how to use `create-user`.

AWS CLI

To create a user

The following `create-user` example creates a new user.


```
aws memorydb create-user \  
  --user-name user-name-1 \  
  --access-string "~objects:* ~items:* ~public:*" \  
  --authentication-mode \  
    Passwords="enterapasswordhere",Type=password
```

Output:

```
{  
  "User": {  
    "Name": "user-name-1",  
    "Status": "active",  
    "AccessString": "off ~objects:* ~items:* ~public:* resetchannels -@all",  
    "ACLNames": [],  
    "MinimumEngineVersion": "6.2",  
    "Authentication": {  
      "Type": "password",  
      "PasswordCount": 1  
    },  
    "ARN": "arn:aws:memorydb:us-west-2:491658xxxxxx:user/user-name-1"  
  }  
}
```

For more information, see [Authenticating users with Access Control Lists](#) in the *MemoryDB User Guide*.

- For API details, see [CreateUser](#) in *AWS CLI Command Reference*.

delete-acl

The following code example shows how to use `delete-acl`.

AWS CLI

To delete an ACL

The following `delete-acl` example deletes an Access control list.

```
aws memorydb delete-acl \  
  --acl-name "new-acl-1"
```

Output:

```
{
  "ACL": {
    "Name": "new-acl-1",
    "Status": "deleting",
    "UserNames": [
      "pat"
    ],
    "MinimumEngineVersion": "6.2",
    "Clusters": [],
    "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:acl/new-acl-1"
  }
}
```

For more information, see [Authenticating users with Access Control Lists](#) in the *MemoryDB User Guide*.

- For API details, see [DeleteAcl](#) in *AWS CLI Command Reference*.

delete-cluster

The following code example shows how to use `delete-cluster`.

AWS CLI

To delete a cluster

The following `delete-cluster` example deletes a cluster.

```
aws memorydb delete-cluster \
  --cluster-name my-new-cluster
```

Output:

```
{
  "Cluster": {
    "Name": "my-new-cluster",
    "Status": "deleting",
    "NumberOfShards": 1,
    "ClusterEndpoint": {
      "Address": "clustercfg.my-new-cluster.xxxxx.memorydb.us-
east-1.amazonaws.com",
```

```

        "Port": 6379
    },
    "NodeType": "db.r6g.large",
    "EngineVersion": "6.2",
    "EnginePatchVersion": "6.2.6",
    "ParameterGroupName": "default.memorydb-redis6",
    "ParameterGroupStatus": "in-sync",
    "SubnetGroupName": "my-sg",
    "TLSEnabled": true,
    "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:cluster/my-new-cluster",
    "SnapshotRetentionLimit": 0,
    "MaintenanceWindow": "sat:10:00-sat:11:00",
    "SnapshotWindow": "07:30-08:30",
    "AutoMinorVersionUpgrade": true
}
}

```

For more information, see [Deleting a cluster](#) in the *MemoryDB User Guide*.

- For API details, see [DeleteCluster](#) in *AWS CLI Command Reference*.

delete-parameter-group

The following code example shows how to use delete-parameter-group.

AWS CLI

To delete a parameter group

The following delete-parameter-group example deletes a parameter group.

```
aws memorydb delete-parameter-group \
  --parameter-group-name myRedis6x
```

Output:

```
{
  "ParameterGroup": {
    "Name": "myredis6x",
    "Family": "memorydb_redis6",
    "Description": "my-parameter-group",
    "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:parametergroup/myredis6x"
  }
}
```

```
}  
}
```

For more information, see [Deleting a parameter group](#) in the *MemoryDB User Guide*.

- For API details, see [DeleteParameterGroup](#) in *AWS CLI Command Reference*.

delete-snapshot

The following code example shows how to use `delete-snapshot`.

AWS CLI

To delete a snapshot

The following `delete-snapshot` example deletes a snapshot.

```
aws memorydb delete-snapshot \  
  --snapshot-name my-cluster-snapshot
```

Output:

```
{  
  "Snapshot": {  
    "Name": "my-cluster-snapshot",  
    "Status": "deleting",  
    "Source": "manual",  
    "ARN": "arn:aws:memorydb:us-east-1:49165xxxxxx:snapshot/my-cluster-snapshot",  
    "ClusterConfiguration": {  
      "Name": "my-cluster",  
      "Description": "",  
      "NodeType": "db.r6g.large",  
      "EngineVersion": "6.2",  
      "MaintenanceWindow": "wed:03:00-wed:04:00",  
      "Port": 6379,  
      "ParameterGroupName": "default.memorydb-redis6",  
      "SubnetGroupName": "my-sg",  
      "VpcId": "vpc-862xxxxc",  
      "SnapshotRetentionLimit": 0,  
      "SnapshotWindow": "04:30-05:30",  
      "NumShards": 2
```

```
    }  
  }  
}
```

For more information, see [Deleting a snapshot](#) in the *MemoryDB User Guide*.

- For API details, see [DeleteSnapshot](#) in *AWS CLI Command Reference*.

delete-subnet-group

The following code example shows how to use `delete-subnet-group`.

AWS CLI

To delete a subnet group

The following `delete-subnet-group` example deletes a subnet.

```
aws memorydb delete-subnet-group \  
  --subnet-group-name mysubnetgroup
```

Output:

```
{  
  "SubnetGroup": {  
    "Name": "mysubnetgroup",  
    "Description": "my subnet group",  
    "VpcId": "vpc-86xxxx4fc",  
    "Subnets": [  
      {  
        "Identifier": "subnet-56xxx61b",  
        "AvailabilityZone": {  
          "Name": "us-east-1a"  
        }  
      }  
    ],  
    "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:subnetgroup/mysubnetgroup"  
  }  
}
```

For more information, see [Deleting a subnet group](#) in the *MemoryDB User Guide*.

- For API details, see [DeleteSubnetGroup](#) in *AWS CLI Command Reference*.

delete-user

The following code example shows how to use `delete-user`.

AWS CLI

To delete a user

The following `delete-user` example deletes a user.

```
aws memorydb delete-user \  
  --user-name my-user
```

Output:

```
{  
  "User": {  
    "Name": "my-user",  
    "Status": "deleting",  
    "AccessString": "on ~app:* resetchannels -@all +@read",  
    "ACLNames": [  
      "my-acl"  
    ],  
    "MinimumEngineVersion": "6.2",  
    "Authentication": {  
      "Type": "password",  
      "PasswordCount": 1  
    },  
    "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:user/my-user"  
  }  
}
```

For more information, see [Authenticating users with Access Control Lists](#) in the *MemoryDB User Guide*.

- For API details, see [DeleteUser](#) in *AWS CLI Command Reference*.

describe-acls

The following code example shows how to use `describe-acls`.

AWS CLI

To return a list of ACLs

The following `describe-acls` returns a list of ACLs.`

```
aws memorydb describe-acls
```

Output:

```
{
  "ACLs": [
    {
      "Name": "open-access",
      "Status": "active",
      "UserNames": [
        "default"
      ],
      "MinimumEngineVersion": "6.2",
      "Clusters": [],
      "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:acl/open-access"
    },
    {
      "Name": "my-acl",
      "Status": "active",
      "UserNames": [],
      "MinimumEngineVersion": "6.2",
      "Clusters": [
        "my-cluster"
      ],
      "ARN": "arn:aws:memorydb:us-east-1:49165xxxxxx:acl/my-acl"
    }
  ]
}
```

For more information, see [Authenticating users with Access Control Lists](#) in the *MemoryDB User Guide*.

- For API details, see [DescribeAcls](#) in *AWS CLI Command Reference*.

describe-clusters

The following code example shows how to use `describe-clusters`.

AWS CLI

To return a list of clusters

The following `describe-clusters` returns a list of clusters.`

```
aws memorydb describe-clusters
```

Output:

```
{
  "Clusters": [
    {
      "Name": "my-cluster",
      "Status": "available",
      "NumberOfShards": 2,
      "ClusterEndpoint": {
        "Address": "clustercfg.my-cluster.llru6f.memorydb.us-
east-1.amazonaws.com",
        "Port": 6379
      },
      "NodeType": "db.r6g.large",
      "EngineVersion": "6.2",
      "EnginePatchVersion": "6.2.6",
      "ParameterGroupName": "default.memorydb-redis6",
      "ParameterGroupStatus": "in-sync",
      "SecurityGroups": [
        {
          "SecurityGroupId": "sg-0a1434xxxxxc9fae",
          "Status": "active"
        }
      ],
      "SubnetGroupName": "pat-sg",
      "TLSEnabled": true,
      "ARN": "arn:aws:memorydb:us-east-1:49165xxxxxx:cluster/my-cluster",
      "SnapshotRetentionLimit": 0,
      "MaintenanceWindow": "wed:03:00-wed:04:00",
      "SnapshotWindow": "04:30-05:30",
      "ACLName": "my-acl",
      "AutoMinorVersionUpgrade": true
    }
  ]
}
```


For more information, see [Managing clusters](#) in the *MemoryDB User Guide*.

- For API details, see [DescribeClusters](#) in *AWS CLI Command Reference*.

describe-engine-versions

The following code example shows how to use describe-engine-versions.

AWS CLI

To return a list of engine versions

The following describe-engine-versions ` returns a list of engine versions.

```
aws memorydb describe-engine-versions
```

Output:

```
{
  "EngineVersions": [
    {
      "EngineVersion": "6.2",
      "EnginePatchVersion": "6.2.6",
      "ParameterGroupFamily": "memorydb_redis6"
    }
  ]
}
```

For more information, see [Engine versions and upgrading](#) in the *MemoryDB User Guide*.

- For API details, see [DescribeEngineVersions](#) in *AWS CLI Command Reference*.

describe-events

The following code example shows how to use describe-events.

AWS CLI

To return a list of events

The following describe-events ` returns a list of events.

```
aws memorydb describe-events
```

Output:

```
{
  "Events": [
    {
      "SourceName": "my-cluster",
      "SourceType": "cluster",
      "Message": "Increase replica count started for replication group my-cluster on 2022-07-22T14:09:01.440Z",
      "Date": "2022-07-22T07:09:01.443000-07:00"
    },
    {
      "SourceName": "my-user",
      "SourceType": "user",
      "Message": "Create user my-user operation completed.",
      "Date": "2022-07-22T07:00:02.975000-07:00"
    }
  ]
}
```

For more information, see [Monitoring events](#) in the *MemoryDB User Guide*.

- For API details, see [DescribeEvents](#) in *AWS CLI Command Reference*.

describe-parameter-groups

The following code example shows how to use `describe-parameter-groups`.

AWS CLI**To return a list of parameter groups**

The following `describe-parameter-groups` returns a list of parameter groups.`

```
aws memorydb describe-parameter-groups
```

Output:

```
{
  "ParameterGroups": [
    {
      "Name": "default.memorydb-redis6",
      "Family": "memorydb_redis6",
```

```

        "Description": "Default parameter group for memorydb_redis6",
        "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:parametergroup/
default.memorydb-redis6"
    }
]
}

```

For more information, see [Configuring engine parameters using parameter groups](#) in the *MemoryDB User Guide*.

- For API details, see [DescribeParameterGroups](#) in *AWS CLI Command Reference*.

describe-parameters

The following code example shows how to use describe-parameters.

AWS CLI

To return a list of parameters

The following describe-parameters` returns a list of parameters.

```
aws memorydb describe-parameters
```

Output:

```

{
  "Parameters": [
    {
      "Name": "acllog-max-len",
      "Value": "128",
      "Description": "The maximum length of the ACL Log",
      "DataType": "integer",
      "AllowedValues": "1-10000",
      "MinimumEngineVersion": "6.2.4"
    },
    {
      "Name": "activedefrag",
      "Value": "no",
      "Description": "Enabled active memory defragmentation",
      "DataType": "string",
      "AllowedValues": "yes,no",
      "MinimumEngineVersion": "6.2.4"
    }
  ]
}

```

```
    },
    {
      "Name": "active-defrag-cycle-max",
      "Value": "75",
      "Description": "Maximal effort for defrag in CPU percentage",
      "DataType": "integer",
      "AllowedValues": "1-75",
      "MinimumEngineVersion": "6.2.4"
    },
    {
      "Name": "active-defrag-cycle-min",
      "Value": "5",
      "Description": "Minimal effort for defrag in CPU percentage",
      "DataType": "integer",
      "AllowedValues": "1-75",
      "MinimumEngineVersion": "6.2.4"
    },
    {
      "Name": "active-defrag-ignore-bytes",
      "Value": "104857600",
      "Description": "Minimum amount of fragmentation waste to start active
defrag",
      "DataType": "integer",
      "AllowedValues": "1048576-",
      "MinimumEngineVersion": "6.2.4"
    },
    {
      "Name": "active-defrag-max-scan-fields",
      "Value": "1000",
      "Description": "Maximum number of set/hash/zset/list fields that will be
processed from the main dictionary scan",
      "DataType": "integer",
      "AllowedValues": "1-1000000",
      "MinimumEngineVersion": "6.2.4"
    },
    {
      "Name": "active-defrag-threshold-lower",
      "Value": "10",
      "Description": "Minimum percentage of fragmentation to start active
defrag",
      "DataType": "integer",
      "AllowedValues": "1-100",
      "MinimumEngineVersion": "6.2.4"
    },
  ],
```

```
{
  "Name": "active-defrag-threshold-upper",
  "Value": "100",
  "Description": "Maximum percentage of fragmentation at which we use
maximum effort",
  "DataType": "integer",
  "AllowedValues": "1-100",
  "MinimumEngineVersion": "6.2.4"
},
{
  "Name": "active-expire-effort",
  "Value": "1",
  "Description": "The amount of effort that redis uses to expire items in
the active expiration job",
  "DataType": "integer",
  "AllowedValues": "1-10",
  "MinimumEngineVersion": "6.2.4"
},
{
  "Name": "activeresharding",
  "Value": "yes",
  "Description": "Apply resharding or not",
  "DataType": "string",
  "AllowedValues": "yes,no",
  "MinimumEngineVersion": "6.2.4"
},
{
  "Name": "client-output-buffer-limit-normal-hard-limit",
  "Value": "0",
  "Description": "Normal client output buffer hard limit in bytes",
  "DataType": "integer",
  "AllowedValues": "0-",
  "MinimumEngineVersion": "6.2.4"
},
{
  "Name": "client-output-buffer-limit-normal-soft-limit",
  "Value": "0",
  "Description": "Normal client output buffer soft limit in bytes",
  "DataType": "integer",
  "AllowedValues": "0-",
  "MinimumEngineVersion": "6.2.4"
},
{
  "Name": "client-output-buffer-limit-normal-soft-seconds",
```

```
    "Value": "0",
    "Description": "Normal client output buffer soft limit in seconds",
    "DataType": "integer",
    "AllowedValues": "0-",
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "client-output-buffer-limit-pubsub-hard-limit",
    "Value": "33554432",
    "Description": "Pubsub client output buffer hard limit in bytes",
    "DataType": "integer",
    "AllowedValues": "0-",
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "client-output-buffer-limit-pubsub-soft-limit",
    "Value": "8388608",
    "Description": "Pubsub client output buffer soft limit in bytes",
    "DataType": "integer",
    "AllowedValues": "0-",
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "client-output-buffer-limit-pubsub-soft-seconds",
    "Value": "60",
    "Description": "Pubsub client output buffer soft limit in seconds",
    "DataType": "integer",
    "AllowedValues": "0-",
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "hash-max-ziplist-entries",
    "Value": "512",
    "Description": "The maximum number of hash entries in order for the
dataset to be compressed",
    "DataType": "integer",
    "AllowedValues": "0-",
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "hash-max-ziplist-value",
    "Value": "64",
    "Description": "The threshold of biggest hash entries in order for the
dataset to be compressed",
```

```
    "DataType": "integer",
    "AllowedValues": "0-",
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "hll-sparse-max-bytes",
    "Value": "3000",
    "Description": "HyperLogLog sparse representation bytes limit",
    "DataType": "integer",
    "AllowedValues": "1-16000",
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "lazyfree-lazy-eviction",
    "Value": "no",
    "Description": "Perform an asynchronous delete on evictions",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "lazyfree-lazy-expire",
    "Value": "no",
    "Description": "Perform an asynchronous delete on expired keys",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "lazyfree-lazy-server-del",
    "Value": "no",
    "Description": "Perform an asynchronous delete on key updates",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "lazyfree-lazy-user-del",
    "Value": "no",
    "Description": "Specifies whether the default behavior of DEL command
acts the same as UNLINK",
    "DataType": "string",
    "AllowedValues": "yes,no",
    "MinimumEngineVersion": "6.2.4"
```

```
    },
    {
      "Name": "lfu-decay-time",
      "Value": "1",
      "Description": "The amount of time in minutes to decrement the key
counter for LFU eviction policyd",
      "DataType": "integer",
      "AllowedValues": "0-",
      "MinimumEngineVersion": "6.2.4"
    },
    {
      "Name": "lfu-log-factor",
      "Value": "10",
      "Description": "The log factor for incrementing key counter for LFU
eviction policy",
      "DataType": "integer",
      "AllowedValues": "1-",
      "MinimumEngineVersion": "6.2.4"
    },
    {
      "Name": "list-compress-depth",
      "Value": "0",
      "Description": "Number of quicklist ziplist nodes from each side of
the list to exclude from compression. The head and tail of the list are always
uncompressed for fast push/pop operations",
      "DataType": "integer",
      "AllowedValues": "0-",
      "MinimumEngineVersion": "6.2.4"
    },
    {
      "Name": "maxmemory-policy",
      "Value": "noeviction",
      "Description": "Max memory policy",
      "DataType": "string",
      "AllowedValues": "volatile-lru,allkeys-lru,volatile-lfu,allkeys-
lfu,volatile-random,allkeys-random,volatile-ttl,noeviction",
      "MinimumEngineVersion": "6.2.4"
    },
    {
      "Name": "maxmemory-samples",
      "Value": "3",
      "Description": "Max memory samples",
      "DataType": "integer",
      "AllowedValues": "1-",
```



```
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "notify-keyspace-events",
    "Description": "The keyspace events for Redis to notify Pub/Sub clients
about. By default all notifications are disabled",
    "DataType": "string",
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "set-max-intset-entries",
    "Value": "512",
    "Description": "The limit in the size of the set in order for the
dataset to be compressed",
    "DataType": "integer",
    "AllowedValues": "0-",
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "slowlog-log-slower-than",
    "Value": "10000",
    "Description": "The execution time, in microseconds, to exceed in order
for the command to get logged. Note that a negative number disables the slow log,
while a value of zero forces the logging of every command",
    "DataType": "integer",
    "AllowedValues": "-",
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "slowlog-max-len",
    "Value": "128",
    "Description": "The length of the slow log. There is no limit to this
length. Just be aware that it will consume memory. You can reclaim memory used by
the slow log with SLOWLOG RESET.",
    "DataType": "integer",
    "AllowedValues": "0-",
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "stream-node-max-bytes",
    "Value": "4096",
    "Description": "The maximum size of a single node in a stream in bytes",
    "DataType": "integer",
    "AllowedValues": "0-",
```

```
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "stream-node-max-entries",
    "Value": "100",
    "Description": "The maximum number of items a single node in a stream
can contain",
    "DataType": "integer",
    "AllowedValues": "0-",
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "tcp-keepalive",
    "Value": "300",
    "Description": "If non-zero, send ACKs every given number of seconds",
    "DataType": "integer",
    "AllowedValues": "0-",
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "timeout",
    "Value": "0",
    "Description": "Close connection if client is idle for a given number of
seconds, or never if 0",
    "DataType": "integer",
    "AllowedValues": "0,20-",
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "tracking-table-max-keys",
    "Value": "1000000",
    "Description": "The maximum number of keys allowed for the tracking
table for client side caching",
    "DataType": "integer",
    "AllowedValues": "1-1000000000",
    "MinimumEngineVersion": "6.2.4"
  },
  {
    "Name": "zset-max-ziplist-entries",
    "Value": "128",
    "Description": "The maximum number of sorted set entries in order for
the dataset to be compressed",
    "DataType": "integer",
    "AllowedValues": "0-",
```

```

        "MinimumEngineVersion": "6.2.4"
    },
    {
        "Name": "zset-max-ziplist-value",
        "Value": "64",
        "Description": "The threshold of biggest sorted set entries in order for
the dataset to be compressed",
        "DataType": "integer",
        "AllowedValues": "0-",
        "MinimumEngineVersion": "6.2.4"
    }
]
}

```

For more information, see [Configuring engine parameters using parameter groups](#) in the *MemoryDB User Guide*.

- For API details, see [DescribeParameters](#) in *AWS CLI Command Reference*.

describe-snapshots

The following code example shows how to use `describe-snapshots`.

AWS CLI

To return a list of snapshots

The following `describe-snapshots`` returns a list of snapshots.

```
aws memorydb describe-snapshots
```

Output:

```

{
  "Snapshots": [
    {
      "Name": "my-cluster-snapshot",
      "Status": "available",
      "Source": "manual",
      "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx2:snapshot/my-cluster-
snapshot",
      "ClusterConfiguration": {

```

```
    "Name": "my-cluster",
    "Description": " ",
    "NodeType": "db.r6g.large",
    "EngineVersion": "6.2",
    "MaintenanceWindow": "wed:03:00-wed:04:00",
    "Port": 6379,
    "ParameterGroupName": "default.memorydb-redis6",
    "SubnetGroupName": "my-sg",
    "VpcId": "vpc-862574fc",
    "SnapshotRetentionLimit": 0,
    "SnapshotWindow": "04:30-05:30",
    "NumShards": 2
  }
}
```

For more information, see [Snapshot and restore](#) in the *MemoryDB User Guide*.

- For API details, see [DescribeSnapshots](#) in *AWS CLI Command Reference*.

describe-subnet-groups

The following code example shows how to use `describe-subnet-groups`.

AWS CLI

To return a list of subnet groups

The following `describe-subnet-groups` returns a list of subnet groups.`

```
aws memorydb describe-subnet-groups
```

Output

```
{
  "SubnetGroups": [
    {
      "Name": "my-sg",
      "Description": "pat-sg",
      "VpcId": "vpc-86xxx4fc",
      "Subnets": [
        {
          "Identifier": "subnet-faxx84a6",
```

```

        "AvailabilityZone": {
            "Name": "us-east-1b"
        }
    },
    {
        "Identifier": "subnet-56xxf61b",
        "AvailabilityZone": {
            "Name": "us-east-1a"
        }
    }
],
"ARN": "arn:aws:memorydb:us-east-1:49165xxxxxx:subnetgroup/my-sg"
}
]
}

```

For more information, see [Subnets and subnet groups](#) in the *MemoryDB User Guide*.

- For API details, see [DescribeSubnetGroups](#) in *AWS CLI Command Reference*.

describe-users

The following code example shows how to use `describe-users`.

AWS CLI

To return a list of users

The following `describe-users`` returns a list of users.

```
aws memorydb describe-users
```

Output

```

{
  "Users": [
    {
      "Name": "default",
      "Status": "active",
      "AccessString": "on ~* &* +@all",
      "ACLNames": [
        "open-access"
      ],
    },
  ],
}

```

```

    "MinimumEngineVersion": "6.0",
    "Authentication": {
      "Type": "no-password"
    },
    "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:user/default"
  },
  {
    "Name": "my-user",
    "Status": "active",
    "AccessString": "off ~objects:* ~items:* ~public:* resetchannels -@all",
    "ACLNames": [],
    "MinimumEngineVersion": "6.2",
    "Authentication": {
      "Type": "password",
      "PasswordCount": 2
    },
    "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:user/my-user"
  }
]
}

```

For more information, see [Authenticating users with Access Control Lists](#) in the *MemoryDB User Guide*.

- For API details, see [DescribeUsers](#) in *AWS CLI Command Reference*.

failover-shard

The following code example shows how to use failover-shard.

AWS CLI

To fail over a shard

The following failover-shard` fails over a shard.

```
aws memorydb failover-shard \
  --cluster-name my-cluster --shard-name 0001
```

Output:

```
{
```

```
"Cluster": {
  "Name": "my-cluster",
  "Status": "available",
  "NumberOfShards": 2,
  "ClusterEndpoint": {
    "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
    "Port": 6379
  },
  "NodeType": "db.r6g.large",
  "EngineVersion": "6.2",
  "EnginePatchVersion": "6.2.6",
  "ParameterGroupName": "default.memorydb-redis6",
  "ParameterGroupStatus": "in-sync",
  "SecurityGroups": [
    {
      "SecurityGroupId": "sg-0a143xxxx45c9fae",
      "Status": "active"
    }
  ],
  "SubnetGroupName": "my-sg",
  "TLSEnabled": true,
  "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:cluster/my-cluster",
  "SnapshotRetentionLimit": 0,
  "MaintenanceWindow": "wed:03:00-wed:04:00",
  "SnapshotWindow": "04:30-05:30",
  "AutoMinorVersionUpgrade": true
}
}
```

For more information, see [Minimizing downtime with MultiAZ](#) in the *MemoryDB User Guide*.

- For API details, see [FailoverShard](#) in *AWS CLI Command Reference*.

list-allowed-node-type-updates

The following code example shows how to use `list-allowed-node-type-updates`.

AWS CLI

To return a list of allowed node type updates

The following `list-allowed-node-type-updates` returns a list of available node type updates.

```
aws memorydb list-allowed-node-type-updates
```

Output:

```
{
  "Cluster": {
    "Name": "my-cluster",
    "Status": "available",
    "NumberOfShards": 2,
    "ClusterEndpoint": {
      "Address": "clustercfg.my-cluster.xxxxxx.memorydb.us-
east-1.amazonaws.com",
      "Port": 6379
    },
    "NodeType": "db.r6g.large",
    "EngineVersion": "6.2",
    "EnginePatchVersion": "6.2.6",
    "ParameterGroupName": "default.memorydb-redis6",
    "ParameterGroupStatus": "in-sync",
    "SecurityGroups": [
      {
        "SecurityGroupId": "sg-0a143xxxx45c9fae",
        "Status": "active"
      }
    ],
    "SubnetGroupName": "my-sg",
    "TLSEnabled": true,
    "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:cluster/my-cluster",
    "SnapshotRetentionLimit": 0,
    "MaintenanceWindow": "wed:03:00-wed:04:00",
    "SnapshotWindow": "04:30-05:30",
    "AutoMinorVersionUpgrade": true
  }
}
```

For more information, see [Scaling](#) in the *MemoryDB User Guide*.

- For API details, see [ListAllowedNodeTypeUpdates](#) in *AWS CLI Command Reference*.

list-tags

The following code example shows how to use `list-tags`.

AWS CLI

To return a list of tags

The following `list-tags` returns a list of tags.

```
aws memorydb list-tags \  
  --resource-arn arn:aws:memorydb:us-east-1:491658xxxxxx:cluster/my-cluster
```

Output:

```
{  
  "TagList": [  
    {  
      "Key": "mytag",  
      "Value": "myvalue"  
    }  
  ]  
}
```

For more information, see [Tagging resources](#) in the *MemoryDB User Guide*.

- For API details, see [ListTags](#) in *AWS CLI Command Reference*.

reset-parameter-group

The following code example shows how to use `reset-parameter-group`.

AWS CLI

To reset a parameter group

The following `reset-parameter-group` resets a parameter group.

```
aws memorydb reset-parameter-group \  
  --parameter-group-name my-parameter-group \  
  --all-parameters
```

Output:

```
{
```

```
"ParameterGroup": {
  "Name": "my-parameter-group",
  "Family": "memorydb_redis6",
  "Description": "my parameter group",
  "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:parametergroup/my-parameter-
group"
}
```

For more information, see [Configuring engine parameters using parameter groups](#) in the *MemoryDB User Guide*.

- For API details, see [ResetParameterGroup](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To tag a resource

The following tag-resource` adds a tag to a resource.

```
aws memorydb tag-resource \
  --resource-arn arn:aws:memorydb:us-east-1:491658xxxxxx:cluster/my-cluster \
  --tags Key="mykey",Value="myvalue"
```

Output:

```
{
  "TagList": [
    {
      "Key": "mytag",
      "Value": "myvalue"
    },
    {
      "Key": "mykey",
      "Value": "myvalue"
    }
  ]
}
```

For more information, see [Tagging resources](#) in the *MemoryDB User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To update an ACL

The following `update-acl`` updates an ACL by adding a user.

```
aws memorydb untag-resource \  
  --resource-arn arn:aws:memorydb:us-east-1:491658xxxxx:cluster/my-cluster \  
  --tag-keys mykey
```

Output:

```
{  
  "TagList": [  
    {  
      "Key": "mytag",  
      "Value": "myvalue"  
    }  
  ]  
}
```

For more information, see [Tagging resources](#) in the *MemoryDB User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-cluster

The following code example shows how to use `update-cluster`.

AWS CLI

To update a cluster

The following `update-cluster` ``` updates the parameter group of a cluster to `my-parameter-group`.

```
aws memorydb update-cluster \  
  --cluster-name my-cluster \  
  --parameter-group-name my-parameter-group
```

Output:

```
{  
  "Cluster": {  
    "Name": "my-cluster",  
    "Status": "available",  
    "NumberOfShards": 2,  
    "AvailabilityMode": "MultiAZ",  
    "ClusterEndpoint": {  
      "Address": "clustercfg.my-cluster.llru6f.memorydb.us-  
east-1.amazonaws.com",  
      "Port": 6379  
    },  
    "NodeType": "db.r6g.large",  
    "EngineVersion": "6.2",  
    "EnginePatchVersion": "6.2.6",  
    "ParameterGroupName": "my-parameter-group",  
    "ParameterGroupStatus": "in-sync",  
    "SecurityGroups": [  
      {  
        "SecurityGroupId": "sg-0a143xxxxxc9fae",  
        "Status": "active"  
      }  
    ],  
    "SubnetGroupName": "pat-sg",  
    "TLSEnabled": true,  
    "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:cluster/my-cluster",  
    "SnapshotRetentionLimit": 0,  
    "MaintenanceWindow": "wed:03:00-wed:04:00",  
    "SnapshotWindow": "04:30-05:30",  
    "ACLName": "my-acl",  
    "AutoMinorVersionUpgrade": true  
  }  
}
```

For more information, see [Modifying a cluster](#) in the *MemoryDB User Guide*.

- For API details, see [UpdateCluster](#) in *AWS CLI Command Reference*.

update-parameter-group

The following code example shows how to use `update-parameter-group`.

AWS CLI

To update a parameter group

The following `update-parameter-group` updates a parameter group.

```
aws memorydb update-parameter-group \  
  --parameter-group-name my-parameter-group \  
  --parameter-name-values "ParameterName=activedefrag, ParameterValue=no"
```

Output:

```
{  
  "ParameterGroup": {  
    "Name": "my-parameter-group",  
    "Family": "memorydb_redis6",  
    "Description": "my parameter group",  
    "ARN": "arn:aws:memorydb:us-east-1:49165xxxxxx:parametergroup/my-parameter-  
group"  
  }  
}
```

For more information, see [Modifying a parameter group](#) in the *MemoryDB User Guide*.

- For API details, see [UpdateParameterGroup](#) in *AWS CLI Command Reference*.

update-subnet-group

The following code example shows how to use `update-subnet-group`.

AWS CLI

To update a subnet group

The following `update-subnet-group` updates a subnet group's subnet ID.

```
aws memorydb update-subnet-group \  
  --subnet-group-name my-sg \  
  --subnet-ids subnet-01f29d458f3xxxxx
```

Output:

```
{
  "SubnetGroup": {
    "Name": "my-sg-1",
    "Description": "my-sg",
    "VpcId": "vpc-09d2cfc01xxxxxxx",
    "Subnets": [
      {
        "Identifier": "subnet-01f29d458fxxxxxx",
        "AvailabilityZone": {
          "Name": "us-east-1a"
        }
      }
    ],
    "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:subnetgroup/my-sg"
  }
}
```

For more information, see [Subnets and subnet groups](#) in the *MemoryDB User Guide*.

- For API details, see [UpdateSubnetGroup](#) in *AWS CLI Command Reference*.

update-user

The following code example shows how to use `update-user`.

AWS CLI**To update a user**

The following `update-user` modifies a user's access string.

```
aws memorydb update-user \
  --user-name my-user \
  --access-string "off ~objects:* ~items:* ~public:* resetchannels -@all"
```

Output:

```
{
  "User": {
    "Name": "my-user",
    "Status": "modifying",
```

```
    "AccessString": "off ~objects:* ~items:* ~public:* resetchannels -@all",
    "ACLNames": [
      "myt-acl"
    ],
    "MinimumEngineVersion": "6.2",
    "Authentication": {
      "Type": "password",
      "PasswordCount": 2
    },
    "ARN": "arn:aws:memorydb:us-east-1:491658xxxxxx:user/my-user"
  }
}
```

For more information, see [Authenticating users with Access Control Lists](#) in the *MemoryDB User Guide*.

- For API details, see [UpdateUser](#) in *AWS CLI Command Reference*.

Amazon MSK examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon MSK.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-cluster

The following code example shows how to use `create-cluster`.

AWS CLI

To create an Amazon MSK cluster

The following `create-cluster` example creates an MSK cluster named `MessagingCluster` with three broker nodes. A JSON file named `brokernodegroupinfo.json` specifies the three subnets over which you want Amazon MSK to distribute the broker nodes. This example doesn't specify the monitoring level, so the cluster gets the `DEFAULT` level.

```
aws kafka create-cluster \  
  --cluster-name "MessagingCluster" \  
  --broker-node-group-info file://brokernodegroupinfo.json \  
  --kafka-version "2.2.1" \  
  --number-of-broker-nodes 3
```

Contents of `brokernodegroupinfo.json`:

```
{  
  "InstanceType": "kafka.m5.xlarge",  
  "BrokerAZDistribution": "DEFAULT",  
  "ClientSubnets": [  
    "subnet-0123456789111abcd",  
    "subnet-0123456789222abcd",  
    "subnet-0123456789333abcd"  
  ]  
}
```

Output:

```
{  
  "ClusterArn": "arn:aws:kafka:us-west-2:123456789012:cluster/MessagingCluster/  
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE-2",  
  "ClusterName": "MessagingCluster",  
  "State": "CREATING"  
}
```

For more information, see [Create an Amazon MSK Cluster](#) in the *Amazon Managed Streaming for Apache Kafka*.

- For API details, see [CreateCluster](#) in *AWS CLI Command Reference*.

create-configuration

The following code example shows how to use create-configuration.

AWS CLI

To create a custom Amazon MSK configuration

The following create-configuration example creates a custom MSK configuration with the server properties that are specified in the input file.

```
aws kafka create-configuration \  
  --name "CustomConfiguration" \  
  --description "Topic autocreation enabled; Apache ZooKeeper timeout 2000 ms; Log  
rolling 604800000 ms." \  
  --kafka-versions "2.2.1" \  
  --server-properties file://configuration.txt
```

Contents of configuration.txt:

```
auto.create.topics.enable = true  
zookeeper.connection.timeout.ms = 2000  
log.roll.ms = 604800000
```

This command produces no output. Output:

```
{  
  "Arn": "arn:aws:kafka:us-west-2:123456789012:configuration/CustomConfiguration/  
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE-2",  
  "CreationTime": "2019-10-09T15:26:05.548Z",  
  "LatestRevision":  
    {  
      "CreationTime": "2019-10-09T15:26:05.548Z",  
      "Description": "Topic autocreation enabled; Apache ZooKeeper timeout  
2000 ms; Log rolling 604800000 ms.",  
      "Revision": 1  
    },  
  "Name": "CustomConfiguration"  
}
```

For more information, see [Amazon MSK Configuration Operations](#) in the *Amazon Managed Streaming for Apache Kafka Developer Guide*.

- For API details, see [CreateConfiguration](#) in *AWS CLI Command Reference*.

describe-cluster

The following code example shows how to use `describe-cluster`.

AWS CLI

To describe a cluster

The following `describe-cluster` example describes an Amazon MSK cluster.

```
aws kafka describe-cluster \  
  --cluster-arn arn:aws:kafka:us-east-1:123456789012:cluster/demo-  
cluster-1/6357e0b2-0e6a-4b86-a0b4-70df934c2e31-5
```

Output:

```
{  
  "ClusterInfo": {  
    "BrokerNodeGroupInfo": {  
      "BrokerAZDistribution": "DEFAULT",  
      "ClientSubnets": [  
        "subnet-cbfff283",  
        "subnet-6746046b"  
      ],  
      "InstanceType": "kafka.m5.large",  
      "SecurityGroups": [  
        "sg-f839b688"  
      ],  
      "StorageInfo": {  
        "EbsStorageInfo": {  
          "VolumeSize": 100  
        }  
      }  
    },  
    "ClusterArn": "arn:aws:kafka:us-east-1:123456789012:cluster/demo-  
cluster-1/6357e0b2-0e6a-4b86-a0b4-70df934c2e31-5",  
    "ClusterName": "demo-cluster-1",  
    "CreationTime": "2020-07-09T02:31:36.223000+00:00",  
    "CurrentBrokerSoftwareInfo": {  
      "KafkaVersion": "2.2.1"  
    }  
  }  
}
```

```

    },
    "CurrentVersion": "K3AEGXETSR30VB",
    "EncryptionInfo": {
      "EncryptionAtRest": {
        "DataVolumeKMSKeyId": "arn:aws:kms:us-east-1:123456789012:key/
a7ca56d5-0768-4b64-a670-339a9fbef81c"
      },
      "EncryptionInTransit": {
        "ClientBroker": "TLS_PLAINTEXT",
        "InCluster": true
      }
    },
    "EnhancedMonitoring": "DEFAULT",
    "OpenMonitoring": {
      "Prometheus": {
        "JmxExporter": {
          "EnabledInBroker": false
        },
        "NodeExporter": {
          "EnabledInBroker": false
        }
      }
    },
    "NumberOfBrokerNodes": 2,
    "State": "ACTIVE",
    "Tags": {},
    "ZookeeperConnectionString": "z-2.demo-cluster-1.xuy0sb.c5.kafka.us-
east-1.amazonaws.com:2181,z-1.demo-cluster-1.xuy0sb.c5.kafka.us-
east-1.amazonaws.com:2181,z-3.demo-cluster-1.xuy0sb.c5.kafka.us-
east-1.amazonaws.com:2181"
  }
}

```

For more information, see [Listing Amazon MSK Clusters](#) in the *Amazon Managed Streaming for Apache Kafka Developer Guide*.

- For API details, see [DescribeCluster](#) in *AWS CLI Command Reference*.

get-bootstrap-brokers

The following code example shows how to use `get-bootstrap-brokers`.

AWS CLI

To get bootstrap brokers

The following `get-bootstrap-brokers` example retrieves the bootstrap broker information for an Amazon MSK cluster.

```
aws kafka get-bootstrap-brokers \  
  --cluster-arn arn:aws:kafka:us-east-1:123456789012:cluster/demo-  
cluster-1/6357e0b2-0e6a-4b86-a0b4-70df934c2e31-5
```

Output:

```
{  
  "BootstrapBrokerString": "b-1.demo-cluster-1.xuy0sb.c5.kafka.us-  
east-1.amazonaws.com:9092,b-2.demo-cluster-1.xuy0sb.c5.kafka.us-  
east-1.amazonaws.com:9092",  
  "BootstrapBrokerStringTls": "b-1.demo-cluster-1.xuy0sb.c5.kafka.us-  
east-1.amazonaws.com:9094,b-2.demo-cluster-1.xuy0sb.c5.kafka.us-  
east-1.amazonaws.com:9094"  
}
```

For more information, see [Getting the Bootstrap Brokers](#) in the *Amazon Managed Streaming for Apache Kafka Developer Guide*.

- For API details, see [GetBootstrapBrokers](#) in *AWS CLI Command Reference*.

list-clusters

The following code example shows how to use `list-clusters`.

AWS CLI

To list the available clusters

The following `list-clusters` example lists the Amazon MSK clusters in your AWS account.

```
aws kafka list-clusters
```

Output:

```
{
```

```
"ClusterInfoList": [
  {
    "BrokerNodeGroupInfo": {
      "BrokerAZDistribution": "DEFAULT",
      "ClientSubnets": [
        "subnet-cbfff283",
        "subnet-6746046b"
      ],
      "InstanceType": "kafka.m5.large",
      "SecurityGroups": [
        "sg-f839b688"
      ],
      "StorageInfo": {
        "EbsStorageInfo": {
          "VolumeSize": 100
        }
      }
    },
    "ClusterArn": "arn:aws:kafka:us-east-1:123456789012:cluster/demo-
cluster-1/6357e0b2-0e6a-4b86-a0b4-70df934c2e31-5",
    "ClusterName": "demo-cluster-1",
    "CreationTime": "2020-07-09T02:31:36.223000+00:00",
    "CurrentBrokerSoftwareInfo": {
      "KafkaVersion": "2.2.1"
    },
    "CurrentVersion": "K3AEGXETSR30VB",
    "EncryptionInfo": {
      "EncryptionAtRest": {
        "DataVolumeKMSKeyId": "arn:aws:kms:us-east-1:123456789012:key/
a7ca56d5-0768-4b64-a670-339a9fbef81c"
      },
      "EncryptionInTransit": {
        "ClientBroker": "TLS_PLAINTEXT",
        "InCluster": true
      }
    },
    "EnhancedMonitoring": "DEFAULT",
    "OpenMonitoring": {
      "Prometheus": {
        "JmxExporter": {
          "EnabledInBroker": false
        },
        "NodeExporter": {
          "EnabledInBroker": false
        }
      }
    }
  }
]
```

```

        }
    },
    "NumberOfBrokerNodes": 2,
    "State": "ACTIVE",
    "Tags": {},
    "ZookeeperConnectionString": "z-2.demo-cluster-1.xuy0sb.c5.kafka.us-
east-1.amazonaws.com:2181,z-1.demo-cluster-1.xuy0sb.c5.kafka.us-
east-1.amazonaws.com:2181,z-3.demo-cluster-1.xuy0sb.c5.kafka.us-
east-1.amazonaws.com:2181"
    }
]
}

```

For more information, see [Listing Amazon MSK Clusters](#) in the *Amazon Managed Streaming for Apache Kafka Developer Guide*.

- For API details, see [ListClusters](#) in *AWS CLI Command Reference*.

update-broker-storage

The following code example shows how to use `update-broker-storage`.

AWS CLI

To update the EBS storage for brokers

The following `update-broker-storage` example updates the amount of EBS storage for all the brokers in the cluster. Amazon MSK sets the target storage amount for each broker to the amount specified in the example. You can get the current version of the cluster by describing the cluster or by listing all of the clusters.

```

aws kafka update-broker-storage \
  --cluster-arn "arn:aws:kafka:us-west-2:123456789012:cluster/MessagingCluster/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE-2" \
  --current-version "K21V3IB1VIZYYH" \
  --target-broker-ebs-volume-info "KafkaBrokerNodeId=ALL,VolumeSizeGB=1100"

```

The output returns an ARN for this `update-broker-storage` operation. To determine if this operation is complete, use the `describe-cluster-operation` command with this ARN as input.

```
{
  "ClusterArn": "arn:aws:kafka:us-west-2:123456789012:cluster/MessagingCluster/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE-2",
  "ClusterOperationArn": "arn:aws:kafka:us-west-2:123456789012:cluster-
operation/V123450123/a1b2c3d4-1234-abcd-cdef-22222EXAMPLE-2/a1b2c3d4-abcd-1234-
bcde-33333EXAMPLE"
}
```

For more information, see [Update the EBS Storage for Brokers](#) in the *Amazon Managed Streaming for Apache Kafka Developer Guide*.

- For API details, see [UpdateBrokerStorage](#) in *AWS CLI Command Reference*.

update-cluster-configuration

The following code example shows how to use `update-cluster-configuration`.

AWS CLI

To update the configuration of an Amazon MSK cluster

The following `update-cluster-configuration` example updates the configuration of the specified existing MSK cluster. It uses a custom MSK configuration.

```
aws kafka update-cluster-configuration \
  --cluster-arn "arn:aws:kafka:us-west-2:123456789012:cluster/MessagingCluster/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE-2" \
  --configuration-info file://configuration-info.json \
  --current-version "K21V3IB1VIZYYH"
```

Contents of `configuration-info.json`:

```
{
  "Arn": "arn:aws:kafka:us-west-2:123456789012:configuration/CustomConfiguration/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE-2",
  "Revision": 1
}
```

The output returns an ARN for this `update-cluster-configuration` operation. To determine if this operation is complete, use the `describe-cluster-operation` command with this ARN as input.

```
{
  "ClusterArn": "arn:aws:kafka:us-west-2:123456789012:cluster/MessagingCluster/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE-2",
  "ClusterOperationArn": "arn:aws:kafka:us-west-2:123456789012:cluster-
operation/V123450123/a1b2c3d4-1234-abcd-cdef-22222EXAMPLE-2/a1b2c3d4-abcd-1234-
bcde-33333EXAMPLE"
}
```

For more information, see [Update the Configuration of an Amazon MSK Cluster](#) in the *Amazon Managed Streaming for Apache Kafka Developer Guide*.

- For API details, see [UpdateClusterConfiguration](#) in *AWS CLI Command Reference*.

Network Manager examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Network Manager.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-customer-gateway

The following code example shows how to use `associate-customer-gateway`.

AWS CLI

To associate a customer gateway

The following `associate-customer-gateway` example associates customer gateway `cgw-11223344556677889` in the specified global network with device `device-07f6fd08867abc123`.

```
aws networkmanager associate-customer-gateway \  
  --customer-gateway-arn arn:aws:ec2:us-west-2:123456789012:customer-gateway/  
cgw-11223344556677889 \  
  --global-network-id global-network-01231231231231231 \  
  --device-id device-07f6fd08867abc123 \  
  --region us-west-2
```

Output:

```
{  
  "CustomerGatewayAssociation": {  
    "CustomerGatewayArn": "arn:aws:ec2:us-west-2:123456789012:customer-gateway/  
cgw-11223344556677889",  
    "GlobalNetworkId": "global-network-01231231231231231",  
    "DeviceId": "device-07f6fd08867abc123",  
    "State": "PENDING"  
  }  
}
```

For more information, see [Customer Gateway Associations](#) in the *Transit Gateway Network Manager Guide*.

- For API details, see [AssociateCustomerGateway](#) in *AWS CLI Command Reference*.

associate-link

The following code example shows how to use `associate-link`.

AWS CLI

To associate a link

The following `associate-link` example associates link `link-11112222aaaabbbb1` with device `device-07f6fd08867abc123`. The link and device are in the specified global network.

```
aws networkmanager associate-link \  
  --global-network-id global-network-01231231231231231 \  
  --link-id link-11112222aaaabbbb1 \  
  --device-id device-07f6fd08867abc123 \  
  --region us-west-2
```

```
--global-network-id global-network-01231231231231231 \  
--device-id device-07f6fd08867abc123 \  
--link-id link-11112222aaaabbbb1 \  
--region us-west-2
```

Output:

```
{  
  "LinkAssociation": {  
    "GlobalNetworkId": "global-network-01231231231231231",  
    "DeviceId": "device-07f6fd08867abc123",  
    "LinkId": "link-11112222aaaabbbb1",  
    "LinkAssociationState": "PENDING"  
  }  
}
```

For more information, see [Device and Link Associations](#) in the *Transit Gateway Network Manager Guide*.

- For API details, see [AssociateLink](#) in *AWS CLI Command Reference*.

create-core-network

The following code example shows how to use `create-core-network`.

AWS CLI

To create a core network

The following `create-core-network` example creates a core network using an optional description and tags within an AWS Cloud WAN global network.

```
aws networkmanager create-core-network \  
  --global-network-id global-network-0d59060f16a73bc41\  
  --description "Main headquarters location"\  
  --tags Key=Name,Value="New York City office"
```

Output:

```
{
```

```
"CoreNetwork": {
  "GlobalNetworkId": "global-network-0d59060f16a73bc41",
  "CoreNetworkId": "core-network-0fab62fe438d94db6",
  "CoreNetworkArn": "arn:aws:networkmanager::987654321012:core-network/core-
network-0fab62fe438d94db6",
  "Description": "Main headquarters location",
  "CreatedAt": "2022-01-10T19:53:59+00:00",
  "State": "AVAILABLE",
  "Tags": [
    {
      "Key": "Name",
      "Value": "New York City office"
    }
  ]
}
```

For more information, see [Core networks](#) in the *AWS Cloud WAN User Guide*.

- For API details, see [CreateCoreNetwork](#) in *AWS CLI Command Reference*.

create-device

The following code example shows how to use create-device.

AWS CLI

To create a device

The following create-device example creates a device in the specified global network. The device details include a description, the type, vendor, model, and serial number.

```
aws networkmanager create-device
  --global-network-id global-network-01231231231231231 \
  --description "New York office device" \
  --type "office device" \
  --vendor "anycompany" \
  --model "abcabc" \
  --serial-number "1234" \
  --region us-west-2
```

Output:

```
{
  "Device": {
    "DeviceId": "device-07f6fd08867abc123",
    "DeviceArn": "arn:aws:networkmanager::123456789012:device/global-
network-01231231231231231231/device-07f6fd08867abc123",
    "GlobalNetworkId": "global-network-01231231231231231",
    "Description": "New York office device",
    "Type": "office device",
    "Vendor": "anycompany",
    "Model": "abcabc",
    "SerialNumber": "1234",
    "CreatedAt": 1575554005.0,
    "State": "PENDING"
  }
}
```

For more information, see [Working with Devices](#) in the *Transit Gateway Network Manager Guide*.

- For API details, see [CreateDevice](#) in *AWS CLI Command Reference*.

create-global-network

The following code example shows how to use `create-global-network`.

AWS CLI

To create a global network

The following `create-global-network` examples creates a new global network. The initial state upon creation is PENDING.

```
aws networkmanager create-global-network
```

Output:

```
{
  "GlobalNetwork": {
    "GlobalNetworkId": "global-network-00a77fc0f722dae74",
    "GlobalNetworkArn": "arn:aws:networkmanager::987654321012:global-network/
global-network-00a77fc0f722dae74",
    "CreatedAt": "2022-03-14T20:31:56+00:00",
```

```
    "State": "PENDING"
  }
}
```

- For API details, see [CreateGlobalNetwork](#) in *AWS CLI Command Reference*.

create-link

The following code example shows how to use `create-link`.

AWS CLI

To create a link

The following `create-link` example creates a link in the specified global network. The link includes a description and details about the link type, bandwidth, and provider. The site ID indicates the site to which the link is associated.

```
aws networkmanager create-link \
  --global-network-id global-network-01231231231231231 \
  --description "VPN Link" \
  --type "broadband" \
  --bandwidth UploadSpeed=10,DownloadSpeed=20 \
  --provider "AnyCompany" \
  --site-id site-444555aaaabbb11223 \
  --region us-west-2
```

Output:

```
{
  "Link": {
    "LinkId": "link-11112222aaaabbbb1",
    "LinkArn": "arn:aws:networkmanager::123456789012:link/global-
network-01231231231231231/link-11112222aaaabbbb1",
    "GlobalNetworkId": "global-network-01231231231231231",
    "SiteId": "site-444555aaaabbb11223",
    "Description": "VPN Link",
    "Type": "broadband",
    "Bandwidth": {
      "UploadSpeed": 10,
      "DownloadSpeed": 20
    }
  }
}
```

```

    },
    "Provider": "AnyCompany",
    "CreatedAt": 1575555811.0,
    "State": "PENDING"
  }
}

```

For more information, see [Working with Links](#) in the *Transit Gateway Network Manager Guide*.

- For API details, see [CreateLink](#) in *AWS CLI Command Reference*.

create-site

The following code example shows how to use `create-site`.

AWS CLI

To create a site

The following `create-site` example creates a site in the specified global network. The site details include a description and the location information.

```

aws networkmanager create-site \
  --global-network-id global-network-01231231231231231 \
  --description "New York head office" \
  --location Latitude=40.7128,Longitude=-74.0060 \
  --region us-west-2

```

Output:

```

{
  "Site": {
    "SiteId": "site-444555aaabbb11223",
    "SiteArn": "arn:aws:networkmanager::123456789012:site/global-
network-01231231231231231/site-444555aaabbb11223",
    "GlobalNetworkId": "global-network-01231231231231231",
    "Description": "New York head office",
    "Location": {
      "Latitude": "40.7128",
      "Longitude": "-74.0060"
    },
  },
  "CreatedAt": 1575554300.0,
}

```

```
    "State": "PENDING"
  }
}
```

For more information, see [Working with Sites](#) in the *Transit Gateway Network Manager Guide*.

- For API details, see [CreateSite](#) in *AWS CLI Command Reference*.

create-vpc-attachment

The following code example shows how to use `create-vpc-attachment`.

AWS CLI

To create a VPC attachment

The following `create-vpc-attachment` example creates a VPC attachment with IPv6 support in a core network.

```
aws networkmanager create-vpc-attachment \
  --core-network-id core-network-0fab62fe438d94db6 \
  --vpc-arn arn:aws:ec2:us-east-1:987654321012:vpc/vpc-09f37f69e2786eeb8 \
  --subnet-arns arn:aws:ec2:us-east-1:987654321012:subnet/subnet-04ca4e010857e7bb7 \
  --Ipv6Support=true
```

Output:

```
{
  "VpcAttachment": {
    "Attachment": {
      "CoreNetworkId": "core-network-0fab62fe438d94db6",
      "AttachmentId": "attachment-05e1da6eba87a06e6",
      "OwnerAccountId": "987654321012",
      "AttachmentType": "VPC",
      "State": "CREATING",
      "EdgeLocation": "us-east-1",
      "ResourceArn": "arn:aws:ec2:us-east-1:987654321012:vpc/
vpc-09f37f69e2786eeb8",
      "Tags": [],
      "CreatedAt": "2022-03-10T20:59:14+00:00",
      "UpdatedAt": "2022-03-10T20:59:14+00:00"
    }
  }
}
```

```
    },
    "SubnetArns": [
      "arn:aws:ec2:us-east-1:987654321012:subnet/subnet-04ca4e010857e7bb7"
    ],
    "Options": {
      "Ipv6Support": true
    }
  }
}
```

For more information, see [Create an attachment](#) in the *Cloud WAN User Guide*.

- For API details, see [CreateVpcAttachment](#) in *AWS CLI Command Reference*.

delete-attachment

The following code example shows how to use delete-attachment.

AWS CLI

To delete an attachment

The following delete-attachment example deletes a Connect attachment.

```
aws networkmanager delete-attachment \
  --attachment-id attachment-01feddaeeae26ab68c
```

Output:

```
{
  "Attachment": {
    "CoreNetworkId": "core-network-0f4b0a9d5ee7761d1",
    "AttachmentId": "attachment-01feddaeeae26ab68c",
    "OwnerAccountId": "987654321012",
    "AttachmentType": "CONNECT",
    "State": "DELETING",
    "EdgeLocation": "us-east-1",
    "ResourceArn": "arn:aws:networkmanager::987654321012:attachment/attachment-02c3964448fedf5aa",
    "CreatedAt": "2022-03-15T19:18:41+00:00",
    "UpdatedAt": "2022-03-15T19:28:59+00:00"
  }
}
```



```
}
```

For more information, see [Delete attachments](#) in the *Cloud WAN User Guide*.

- For API details, see [DeleteAttachment](#) in *AWS CLI Command Reference*.

delete-bucket-analytics-configuration

The following code example shows how to use `delete-bucket-analytics-configuration`.

AWS CLI

To delete an analytics configuration for a bucket

The following `delete-bucket-analytics-configuration` example removes the analytics configuration for the specified bucket and ID.

```
aws s3api delete-bucket-analytics-configuration \  
  --bucket my-bucket \  
  --id 1
```

This command produces no output.

- For API details, see [DeleteBucketAnalyticsConfiguration](#) in *AWS CLI Command Reference*.

delete-bucket-metrics-configuration

The following code example shows how to use `delete-bucket-metrics-configuration`.

AWS CLI

To delete a metrics configuration for a bucket

The following `delete-bucket-metrics-configuration` example removes the metrics configuration for the specified bucket and ID.

```
aws s3api delete-bucket-metrics-configuration \  
  --bucket my-bucket \  
  --id 123
```

This command produces no output.

- For API details, see [DeleteBucketMetricsConfiguration](#) in *AWS CLI Command Reference*.

delete-core-network

The following code example shows how to use `delete-core-network`.

AWS CLI

To delete a core network

The following `delete-core-network` example deletes a core network from a Cloud WAN global network.

```
aws networkmanager delete-core-network \  
  --core-network-id core-network-0fab62fe438d94db6
```

Output:

```
{  
  "CoreNetwork": {  
    "GlobalNetworkId": "global-network-0d59060f16a73bc41",  
    "CoreNetworkId": "core-network-0fab62fe438d94db6",  
    "Description": "Main headquarters location",  
    "CreatedAt": "2021-12-09T18:31:11+00:00",  
    "State": "DELETING",  
    "Segments": [  
      {  
        "Name": "dev",  
        "EdgeLocations": [  
          "us-east-1"  
        ],  
        "SharedSegments": []  
      }  
    ],  
    "Edges": [  
      {  
        "EdgeLocation": "us-east-1",  
        "Asn": 64512,  
        "InsideCidrBlocks": []  
      }  
    ]  
  }  
}
```

```
}
```

For more information, see [Core networks](#) in the *Cloud WAN User Guide*.

- For API details, see [DeleteCoreNetwork](#) in *AWS CLI Command Reference*.

delete-device

The following code example shows how to use `delete-device`.

AWS CLI

To delete a device

The following `delete-device` example deletes the specified device from the specified global network.

```
aws networkmanager delete-device \  
  --global-network-id global-network-01231231231231231 \  
  --device-id device-07f6fd08867abc123 \  
  --region us-west-2
```

Output:

```
{  
  "Device": {  
    "DeviceId": "device-07f6fd08867abc123",  
    "DeviceArn": "arn:aws:networkmanager::123456789012:device/global-  
network-01231231231231231/device-07f6fd08867abc123",  
    "GlobalNetworkId": "global-network-01231231231231231",  
    "Description": "New York office device",  
    "Type": "office device",  
    "Vendor": "anycompany",  
    "Model": "abcabc",  
    "SerialNumber": "1234",  
    "SiteId": "site-444555aaabbb11223",  
    "CreatedAt": 1575554005.0,  
    "State": "DELETING"  
  }  
}
```

For more information, see [Working with Devices](#) in the *Transit Gateway Network Manager Guide*.

- For API details, see [DeleteDevice](#) in *AWS CLI Command Reference*.

delete-global-network

The following code example shows how to use `delete-global-network`.

AWS CLI

To delete a global network

The following `delete-global-network` example deletes a global network.

```
aws networkmanager delete-global-network \  
  --global-network-id global-network-052bedddccb193b6b
```

Output:

```
{  
  "GlobalNetwork": {  
    "GlobalNetworkId": "global-network-052bedddccb193b6b",  
    "GlobalNetworkArn": "arn:aws:networkmanager::987654321012:global-network/  
global-network-052bedddccb193b6b",  
    "CreatedAt": "2021-12-09T18:19:12+00:00",  
    "State": "DELETING"  
  }  
}
```

- For API details, see [DeleteGlobalNetwork](#) in *AWS CLI Command Reference*.

delete-link

The following code example shows how to use `delete-link`.

AWS CLI

To delete a link

The following `delete-link` example deletes the specified link from the specified global network.

```
aws networkmanager delete-link \  
  --global-network-id global-network-052bedddccb193b6b
```

```
--global-network-id global-network-01231231231231231 \
--link-id link-11112222aaaabbbb1 \
--region us-west-2
```

Output:

```
{
  "Link": {
    "LinkId": "link-11112222aaaabbbb1",
    "LinkArn": "arn:aws:networkmanager::123456789012:link/global-
network-01231231231231231/link-11112222aaaabbbb1",
    "GlobalNetworkId": "global-network-01231231231231231",
    "SiteId": "site-444555aaaabbb11223",
    "Description": "VPN Link",
    "Type": "broadband",
    "Bandwidth": {
      "UploadSpeed": 20,
      "DownloadSpeed": 20
    },
    "Provider": "AnyCompany",
    "CreatedAt": 1575555811.0,
    "State": "DELETING"
  }
}
```

For more information, see [Working with Links](#) in the *Transit Gateway Network Manager Guide*.

- For API details, see [DeleteLink](#) in *AWS CLI Command Reference*.

delete-public-access-block

The following code example shows how to use `delete-public-access-block`.

AWS CLI

To delete the block public access configuration for a bucket

The following `delete-public-access-block` example removes the block public access configuration on the specified bucket.

```
aws s3api delete-public-access-block \
  --bucket my-bucket
```

This command produces no output.

- For API details, see [DeletePublicAccessBlock](#) in *AWS CLI Command Reference*.

delete-site

The following code example shows how to use `delete-site`.

AWS CLI

To delete a site

The following `delete-site` example deletes the specified site (`site-444555aaabbb11223`) in the specified global network.

```
aws networkmanager delete-site \  
  --global-network-id global-network-01231231231231231 \  
  --site-id site-444555aaabbb11223 \  
  --region us-west-2
```

Output:

```
{  
  "Site": {  
    "SiteId": "site-444555aaabbb11223",  
    "SiteArn": "arn:aws:networkmanager::123456789012:site/global-  
network-01231231231231231/site-444555aaabbb11223",  
    "GlobalNetworkId": "global-network-01231231231231231",  
    "Description": "New York head office",  
    "Location": {  
      "Latitude": "40.7128",  
      "Longitude": "-74.0060"  
    },  
    "CreatedAt": 1575554300.0,  
    "State": "DELETING"  
  }  
}
```

For more information, see [Working with Sites](#) in the *Transit Gateway Network Manager Guide*.

- For API details, see [DeleteSite](#) in *AWS CLI Command Reference*.

deregister-transit-gateway

The following code example shows how to use `deregister-transit-gateway`.

AWS CLI

To deregister a transit gateway from a global network

The following `deregister-transit-gateway` example deregisters the specified transit gateway from the specified global network.

```
aws networkmanager deregister-transit-gateway \
  --global-network-id global-network-01231231231231231 \
  --transit-gateway-arn arn:aws:ec2:us-west-2:123456789012:transit-gateway/
  tgw-123abc05e04123abc \
  --region us-west-2
```

Output:

```
{
  "TransitGatewayRegistration": {
    "GlobalNetworkId": "global-network-01231231231231231",
    "TransitGatewayArn": "arn:aws:ec2:us-west-2:123456789012:transit-gateway/
  tgw-123abc05e04123abc",
    "State": {
      "Code": "DELETING"
    }
  }
}
```

For more information, see [Transit Gateway Registrations](#) in the *Transit Gateway Network Manager Guide*.

- For API details, see [DeregisterTransitGateway](#) in *AWS CLI Command Reference*.

describe-global-networks

The following code example shows how to use `describe-global-networks`.

AWS CLI

To describe your global networks

The following `describe-global-networks` example describes all of your global networks in your account.

```
aws networkmanager describe-global-networks \  
  --region us-west-2
```

Output:

```
{  
  "GlobalNetworks": [  
    {  
      "GlobalNetworkId": "global-network-01231231231231231",  
      "GlobalNetworkArn": "arn:aws:networkmanager::123456789012:global-  
network/global-network-01231231231231231",  
      "Description": "Company 1 global network",  
      "CreatedAt": 1575553525.0,  
      "State": "AVAILABLE"  
    }  
  ]  
}
```

- For API details, see [DescribeGlobalNetworks](#) in *AWS CLI Command Reference*.

disassociate-customer-gateway

The following code example shows how to use `disassociate-customer-gateway`.

AWS CLI

To disassociate a customer gateway

The following `disassociate-customer-gateway` example disassociates the specified customer gateway (`cgw-11223344556677889`) from the specified global network.

```
aws networkmanager disassociate-customer-gateway \  
  --global-network-id global-network-01231231231231231 \  
  --customer-gateway-arn arn:aws:ec2:us-west-2:123456789012:customer-gateway/  
cgw-11223344556677889 \  
  --region us-west-2
```

Output:


```
{
  "CustomerGatewayAssociation": {
    "CustomerGatewayArn": "arn:aws:ec2:us-west-2:123456789012:customer-gateway/cgw-11223344556677889",
    "GlobalNetworkId": "global-network-01231231231231231",
    "DeviceId": "device-07f6fd08867abc123",
    "State": "DELETING"
  }
}
```

For more information, see [Customer Gateway Associations](#) in the *Transit Gateway Network Manager Guide*.

- For API details, see [DisassociateCustomerGateway](#) in *AWS CLI Command Reference*.

disassociate-link

The following code example shows how to use `disassociate-link`.

AWS CLI

To disassociate a link

The following `disassociate-link` example disassociates the specified link from device `device-07f6fd08867abc123` in the specified global network.

```
aws networkmanager disassociate-link \
  --global-network-id global-network-01231231231231231 \
  --device-id device-07f6fd08867abc123 \
  --link-id link-11112222aaaabbbb1 \
  --region us-west-2
```

Output:

```
{
  "LinkAssociation": {
    "GlobalNetworkId": "global-network-01231231231231231",
    "DeviceId": "device-07f6fd08867abc123",
    "LinkId": "link-11112222aaaabbbb1",
    "LinkAssociationState": "DELETING"
  }
}
```

For more information, see [Device and Link Associations](#) in the *Transit Gateway Network Manager Guide*.

- For API details, see [DisassociateLink](#) in *AWS CLI Command Reference*.

get-bucket-analytics-configuration

The following code example shows how to use `get-bucket-analytics-configuration`.

AWS CLI

To retrieve the analytics configuration for a bucket with a specific ID

The following `get-bucket-analytics-configuration` example displays the analytics configuration for the specified bucket and ID.

```
aws s3api get-bucket-analytics-configuration \
  --bucket my-bucket \
  --id 1
```

Output:

```
{
  "AnalyticsConfiguration": {
    "StorageClassAnalysis": {},
    "Id": "1"
  }
}
```

- For API details, see [GetBucketAnalyticsConfiguration](#) in *AWS CLI Command Reference*.

get-bucket-metrics-configuration

The following code example shows how to use `get-bucket-metrics-configuration`.

AWS CLI

To retrieve the metrics configuration for a bucket with a specific ID

The following `get-bucket-metrics-configuration` example displays the metrics configuration for the specified bucket and ID.

```
aws s3api get-bucket-metrics-configuration \  
  --bucket my-bucket \  
  --id 123
```

Output:

```
{  
  "MetricsConfiguration": {  
    "Filter": {  
      "Prefix": "logs"  
    },  
    "Id": "123"  
  }  
}
```

- For API details, see [GetBucketMetricsConfiguration](#) in *AWS CLI Command Reference*.

get-customer-gateway-associations

The following code example shows how to use `get-customer-gateway-associations`.

AWS CLI**To get your customer gateway associations**

The following `get-customer-gateway-associations` example gets the customer gateway associations for the specified global network.

```
aws networkmanager get-customer-gateway-associations \  
  --global-network-id global-network-01231231231231231 \  
  --region us-west-2
```

Output:

```
{  
  "CustomerGatewayAssociations": [  
    {  
      "CustomerGatewayArn": "arn:aws:ec2:us-west-2:123456789012:customer-  
gateway/cgw-11223344556677889",  
      "GlobalNetworkId": "global-network-01231231231231231",  
      "DeviceId": "device-07f6fd08867abc123",
```

```
        "State": "AVAILABLE"
      }
    ]
  }
```

- For API details, see [GetCustomerGatewayAssociations](#) in *AWS CLI Command Reference*.

get-devices

The following code example shows how to use `get-devices`.

AWS CLI

To get your devices

The following `get-devices` example gets the devices in the specified global network.

```
aws networkmanager get-devices \
  --global-network-id global-network-01231231231231231 \
  --region us-west-2
```

Output:

```
{
  "Devices": [
    {
      "DeviceId": "device-07f6fd08867abc123",
      "DeviceArn": "arn:aws:networkmanager::123456789012:device/global-
network-01231231231231231/device-07f6fd08867abc123",
      "GlobalNetworkId": "global-network-01231231231231231",
      "Description": "NY office device",
      "Type": "office device",
      "Vendor": "anycompany",
      "Model": "abcabc",
      "SerialNumber": "1234",
      "CreatedAt": 1575554005.0,
      "State": "AVAILABLE"
    }
  ]
}
```

- For API details, see [GetDevices](#) in *AWS CLI Command Reference*.

get-link-associations

The following code example shows how to use `get-link-associations`.

AWS CLI

To get your link associations

The following `get-link-associations` example gets the link associations in the specified global network.

```
aws networkmanager get-link-associations \  
  --global-network-id global-network-01231231231231231 \  
  --region us-west-2
```

Output:

```
{  
  "LinkAssociations": [  
    {  
      "GlobalNetworkId": "global-network-01231231231231231",  
      "DeviceId": "device-07f6fd08867abc123",  
      "LinkId": "link-11112222aaaabbbb1",  
      "LinkAssociationState": "AVAILABLE"  
    }  
  ]  
}
```

- For API details, see [GetLinkAssociations](#) in *AWS CLI Command Reference*.

get-links

The following code example shows how to use `get-links`.

AWS CLI

To get your links

The following `get-links` example gets the links in the specified global network.

```
aws networkmanager get-links \  
  --global-network-id global-network-01231231231231231 \  
  --region us-west-2
```

```
--region us-west-2
```

Output:

```
{
  "Links": [
    {
      "LinkId": "link-11112222aaaabbbb1",
      "LinkArn": "arn:aws:networkmanager::123456789012:link/global-
network-01231231231231231/link-11112222aaaabbbb1",
      "GlobalNetworkId": "global-network-01231231231231231",
      "SiteId": "site-444555aaaabbb11223",
      "Description": "VPN Link",
      "Type": "broadband",
      "Bandwidth": {
        "UploadSpeed": 10,
        "DownloadSpeed": 20
      },
      "Provider": "AnyCompany",
      "CreatedAt": 1575555811.0,
      "State": "AVAILABLE"
    }
  ]
}
```

- For API details, see [GetLinks](#) in *AWS CLI Command Reference*.

get-object-retention

The following code example shows how to use `get-object-retention`.

AWS CLI

To retrieve the object retention configuration for an object

The following `get-object-retention` example retrieves the object retention configuration for the specified object.

```
aws s3api get-object-retention \
  --bucket my-bucket-with-object-lock \
  --key doc1.rtf
```

Output:

```
{
  "Retention": {
    "Mode": "GOVERNANCE",
    "RetainUntilDate": "2025-01-01T00:00:00.000Z"
  }
}
```

- For API details, see [GetObjectRetention](#) in *AWS CLI Command Reference*.

get-public-access-block

The following code example shows how to use `get-public-access-block`.

AWS CLI**To set or modify the block public access configuration for a bucket**

The following `get-public-access-block` example displays the block public access configuration for the specified bucket.

```
aws s3api get-public-access-block --bucket my-bucket
```

Output:

```
{
  "PublicAccessBlockConfiguration": {
    "IgnorePublicAcls": true,
    "BlockPublicPolicy": true,
    "BlockPublicAcls": true,
    "RestrictPublicBuckets": true
  }
}
```

- For API details, see [GetPublicAccessBlock](#) in *AWS CLI Command Reference*.

get-sites

The following code example shows how to use `get-sites`.

AWS CLI

To get your sites

The following `get-sites` example gets the sites in the specified global network.

```
aws networkmanager get-sites \  
  --global-network-id global-network-01231231231231231 \  
  --region us-west-2
```

Output:

```
{  
  "Sites": [  
    {  
      "SiteId": "site-444555aaabbb11223",  
      "SiteArn": "arn:aws:networkmanager::123456789012:site/global-  
network-01231231231231231/site-444555aaabbb11223",  
      "GlobalNetworkId": "global-network-01231231231231231",  
      "Description": "NY head office",  
      "Location": {  
        "Latitude": "40.7128",  
        "Longitude": "-74.0060"  
      },  
      "CreatedAt": 1575554528.0,  
      "State": "AVAILABLE"  
    }  
  ]  
}
```

- For API details, see [GetSites](#) in *AWS CLI Command Reference*.

get-transit-gateway-registrations

The following code example shows how to use `get-transit-gateway-registrations`.

AWS CLI

To get your transit gateway registrations

The following `get-transit-gateway-registrations` example gets the transit gateways that are registered to the specified global network.


```
aws networkmanager get-transit-gateway-registrations \  
  --global-network-id global-network-01231231231231231 \  
  --region us-west-2
```

Output:

```
{  
  "TransitGatewayRegistrations": [  
    {  
      "GlobalNetworkId": "global-network-01231231231231231",  
      "TransitGatewayArn": "arn:aws:ec2:us-west-2:123456789012:transit-  
gateway/tgw-123abc05e04123abc",  
      "State": {  
        "Code": "AVAILABLE"  
      }  
    }  
  ]  
}
```

- For API details, see [GetTransitGatewayRegistrations](#) in *AWS CLI Command Reference*.

get-vpc-attachment

The following code example shows how to use `get-vpc-attachment`.

AWS CLI

To get an a VPC attachment

The following `get-vpc-attachment` example returns information about a VPC attachment.

```
aws networkmanager get-vpc-attachment \  
  --attachment-id attachment-03b7ea450134787da
```

Output:

```
{  
  "VpcAttachment": {  
    "Attachment": {  
      "CoreNetworkId": "core-network-0522de1b226a5d7b3",  
      "AttachmentId": "attachment-03b7ea450134787da",  
    }  
  }  
}
```

```
    "OwnerAccountId": "987654321012",
    "AttachmentType": "VPC",
    "State": "CREATING",
    "EdgeLocation": "us-east-1",
    "ResourceArn": "arn:aws:ec2:us-east-1:987654321012:vpc/vpc-a7c4bbda",
    "Tags": [
      {
        "Key": "Name",
        "Value": "DevVPC"
      }
    ],
    "CreatedAt": "2022-03-11T17:48:58+00:00",
    "UpdatedAt": "2022-03-11T17:48:58+00:00"
  },
  "SubnetArns": [
    "arn:aws:ec2:us-east-1:987654321012:subnet/subnet-202cde6c",
    "arn:aws:ec2:us-east-1:987654321012:subnet/subnet-e5022dba",
    "arn:aws:ec2:us-east-1:987654321012:subnet/subnet-2387ae02",
    "arn:aws:ec2:us-east-1:987654321012:subnet/subnet-cda9dffc"
  ],
  "Options": {
    "Ipv6Support": false
  }
}
```

For more information, see [Attachments](#) in the *Cloud WAN User Guide*.

- For API details, see [GetVpcAttachment](#) in *AWS CLI Command Reference*.

list-bucket-analytics-configurations

The following code example shows how to use `list-bucket-analytics-configurations`.

AWS CLI

To retrieve a list of analytics configurations for a bucket

The following `list-bucket-analytics-configurations` retrieves a list of analytics configurations for the specified bucket.

```
aws s3api list-bucket-analytics-configurations \
  --bucket my-bucket
```

Output:

```
{
  "AnalyticsConfigurationList": [
    {
      "StorageClassAnalysis": {},
      "Id": "1"
    }
  ],
  "IsTruncated": false
}
```

- For API details, see [ListBucketAnalyticsConfigurations](#) in *AWS CLI Command Reference*.

list-bucket-metrics-configurations

The following code example shows how to use `list-bucket-metrics-configurations`.

AWS CLI**To retrieve a list of metrics configurations for a bucket**

The following `list-bucket-metrics-configurations` example retrieves a list of metrics configurations for the specified bucket.

```
aws s3api list-bucket-metrics-configurations \
  --bucket my-bucket
```

Output:

```
{
  "IsTruncated": false,
  "MetricsConfigurationList": [
    {
      "Filter": {
        "Prefix": "logs"
      },
      "Id": "123"
    },
    {
      "Filter": {
```

```
        "Prefix": "tmp"
      },
      "Id": "234"
    ]
  ]
}
```

- For API details, see [ListBucketMetricsConfigurations](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags for a resource

The following `list-tags-for-resource` example lists the tags for the specified device resource (`device-07f6fd08867abc123`).

```
aws networkmanager list-tags-for-resource \
  --resource-arn arn:aws:networkmanager::123456789012:device/global-
network-01231231231231231/device-07f6fd08867abc123 \
  --region us-west-2
```

Output:

```
{
  "TagList": [
    {
      "Key": "Network",
      "Value": "Northeast"
    }
  ]
}
```

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

put-bucket-metrics-configuration

The following code example shows how to use `put-bucket-metrics-configuration`.

AWS CLI

To set a metrics configuration for a bucket

The following `put-bucket-metrics-configuration` example sets a metric configuration with ID 123 for the specified bucket.

```
aws s3api put-bucket-metrics-configuration \  
  --bucket my-bucket \  
  --id 123 \  
  --metrics-configuration '{"Id": "123", "Filter": {"Prefix": "logs"}}'
```

This command produces no output.

- For API details, see [PutBucketMetricsConfiguration](#) in *AWS CLI Command Reference*.

put-object-retention

The following code example shows how to use `put-object-retention`.

AWS CLI

To set an object retention configuration for an object

The following `put-object-retention` example sets an object retention configuration for the specified object until 2025-01-01.

```
aws s3api put-object-retention \  
  --bucket my-bucket-with-object-lock \  
  --key doc1.rtf \  
  --retention '{ "Mode": "GOVERNANCE", "RetainUntilDate": "2025-01-01T00:00:00" }'
```

This command produces no output.

- For API details, see [PutObjectRetention](#) in *AWS CLI Command Reference*.

put-public-access-block

The following code example shows how to use `put-public-access-block`.

AWS CLI

To set the block public access configuration for a bucket

The following `put-public-access-block` example sets a restrictive block public access configuration for the specified bucket.

```
aws s3api put-public-access-block \  
  --bucket my-bucket \  
  --public-access-block-configuration  
  "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets=tr
```

This command produces no output.

- For API details, see [PutPublicAccessBlock](#) in *AWS CLI Command Reference*.

register-transit-gateway

The following code example shows how to use `register-transit-gateway`.

AWS CLI

To register a transit gateway in a global network

The following `register-transit-gateway` example registers transit gateway `tgw-123abc05e04123abc` in the specified global network.

```
aws networkmanager register-transit-gateway \  
  --global-network-id global-network-01231231231231231 \  
  --transit-gateway-arn arn:aws:ec2:us-west-2:123456789012:transit-gateway/  
tgw-123abc05e04123abc \  
  --region us-west-2
```

Output:

```
{  
  "TransitGatewayRegistration": {  
    "GlobalNetworkId": "global-network-01231231231231231",  
    "TransitGatewayArn": "arn:aws:ec2:us-west-2:123456789012:transit-gateway/  
tgw-123abc05e04123abc",  
    "State": {  
      "Code": "PENDING"  
    }  
  }  
}
```

For more information, see [Transit Gateway Registrations](#) in the *Transit Gateway Network Manager Guide*.

- For API details, see [RegisterTransitGateway](#) in *AWS CLI Command Reference*.

reject-attachment

The following code example shows how to use `reject-attachment`.

AWS CLI

To reject an attachment

The following `reject-attachment` example rejects a VPC attachment request.

```
aws networkmanager reject-attachment \  
  --attachment-id attachment-03b7ea450134787da
```

Output:

```
{  
  "Attachment": {  
    "CoreNetworkId": "core-network-0522de1b226a5d7b3",  
    "AttachmentId": "attachment-03b7ea450134787da",  
    "OwnerAccountId": "987654321012",  
    "AttachmentType": "VPC",  
    "State": "AVAILABLE",  
    "EdgeLocation": "us-east-1",  
    "ResourceArn": "arn:aws:ec2:us-east-1:987654321012:vpc/vpc-a7c4bbda",  
    "CreatedAt": "2022-03-11T17:48:58+00:00",  
    "UpdatedAt": "2022-03-11T17:51:25+00:00"  
  }  
}
```

For more information, see [Attachment acceptance](#) in the *Cloud WAN User Guide*.

- For API details, see [RejectAttachment](#) in *AWS CLI Command Reference*.

start-route-analysis

The following code example shows how to use `start-route-analysis`.

AWS CLI

To start route analysis

The following `start-route-analysis` example starts the analysis between a source and destination, including the optional `include-return-path`.

```
aws networkmanager start-route-analysis \  
  --global-network-id global-network-00aa0aaa0b0aaa000 \  
  --source TransitGatewayAttachmentArn=arn:aws:ec2:us-east-1:503089527312:transit-  
gateway-attachment/tgw-attach-0d4a2d491bf68c093,IpAddress=10.0.0.0 \  
  --destination TransitGatewayAttachmentArn=arn:aws:ec2:us-  
west-1:503089527312:transit-gateway-attachment/tgw-  
attach-002577f30bb181742,IpAddress=11.0.0.0 \  
  --include-return-path
```

Output:

```
{  
  "RouteAnalysis": {  
    "GlobalNetworkId": "global-network-00aa0aaa0b0aaa000"  
    "OwnerId": "111122223333",  
    "RouteAnalysisId": "a1873de1-273c-470c-1a2bc2345678",  
    "StartTimestamp": 1695760154.0,  
    "Status": "RUNNING",  
    "Source": {  
      "TransitGatewayAttachmentArn": "arn:aws:ec2:us-  
east-1:111122223333:transit-gateway-attachment/tgw-attach-1234567890abcdef0",  
      "TransitGatewayArn": "arn:aws:ec2:us-east-1:111122223333:transit-  
gateway/tgw-abcdef01234567890",  
      "IpAddress": "10.0.0.0"  
    },  
    "Destination": {  
      "TransitGatewayAttachmentArn": "arn:aws:ec2:us-  
west-1:555555555555:transit-gateway-attachment/tgw-attach-021345abcdef6789",  
      "TransitGatewayArn": "arn:aws:ec2:us-west-1:111122223333:transit-  
gateway/tgw-09876543210fedcba0",  
      "IpAddress": "11.0.0.0"  
    },  
    "IncludeReturnPath": true,  
    "UseMiddleboxes": false  
  }  
}
```


For more information, see [Route Analyzer](#) in the *AWS Global Networks for Transit Gateways User Guide*.

- For API details, see [StartRouteAnalysis](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To apply tags to a resource

The following `tag-resource` example applies the tag `Network=Northeast` to the device `device-07f6fd08867abc123`.

```
aws networkmanager tag-resource \  
  --resource-arn arn:aws:networkmanager::123456789012:device/global-  
network-01231231231231231231231231231231/device-07f6fd08867abc123 \  
  --tags Key=Network,Value=Northeast \  
  --region us-west-2
```

This command produces no output.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from a resource

The following `untag-resource` example removes the tag with the key `Network` from the device `device-07f6fd08867abc123`.

```
aws networkmanager untag-resource \  
  --resource-arn arn:aws:networkmanager::123456789012:device/global-  
network-01231231231231231231231231231231/device-07f6fd08867abc123 ]  
  --tag-keys Network \  
  --region us-west-2
```

This command produces no output.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-device

The following code example shows how to use update-device.

AWS CLI

To update a device

The following update-device example updates device device-07f6fd08867abc123 by specifying a site ID for the device.

```
aws networkmanager update-device \  
  --global-network-id global-network-01231231231231231 \  
  --device-id device-07f6fd08867abc123 \  
  --site-id site-444555aaabbb11223 \  
  --region us-west-2
```

Output:

```
{  
  "Device": {  
    "DeviceId": "device-07f6fd08867abc123",  
    "DeviceArn": "arn:aws:networkmanager::123456789012:device/global-  
network-01231231231231231/device-07f6fd08867abc123",  
    "GlobalNetworkId": "global-network-01231231231231231",  
    "Description": "NY office device",  
    "Type": "Office device",  
    "Vendor": "anycompany",  
    "Model": "abcabc",  
    "SerialNumber": "1234",  
    "SiteId": "site-444555aaabbb11223",  
    "CreatedAt": 1575554005.0,  
    "State": "UPDATING"  
  }  
}
```

For more information, see [Working with Devices](#) in the *Transit Gateway Network Manager Guide*.

- For API details, see [UpdateDevice](#) in *AWS CLI Command Reference*.

update-global-network

The following code example shows how to use `update-global-network`.

AWS CLI

To update a global network

The following `update-global-network` example updates the description for global network `global-network-01231231231231231`.

```
aws networkmanager update-global-network \  
  --global-network-id global-network-01231231231231231 \  
  --description "Head offices" \  
  --region us-west-2
```

Output:

```
{  
  "GlobalNetwork": {  
    "GlobalNetworkId": "global-network-01231231231231231",  
    "GlobalNetworkArn": "arn:aws:networkmanager::123456789012:global-network/  
global-network-01231231231231231",  
    "Description": "Head offices",  
    "CreatedAt": 1575553525.0,  
    "State": "UPDATING"  
  }  
}
```

For more information, see [Global Networks](#) in the *Transit Gateway Network Manager Guide*.

- For API details, see [UpdateGlobalNetwork](#) in *AWS CLI Command Reference*.

update-link

The following code example shows how to use `update-link`.

AWS CLI

To update a link

The following `update-link` example updates the bandwidth information for link `link-11112222aaaabbbb1`.

```
aws networkmanager update-link \  
  --global-network-id global-network-01231231231231231 \  
  --link-id link-11112222aaaabbbb1 \  
  --bandwidth UploadSpeed=20,DownloadSpeed=20 \  
  --region us-west-2
```

Output:

```
{  
  "Link": {  
    "LinkId": "link-11112222aaaabbbb1",  
    "LinkArn": "arn:aws:networkmanager::123456789012:link/global-  
network-01231231231231231/link-11112222aaaabbbb1",  
    "GlobalNetworkId": "global-network-01231231231231231",  
    "SiteId": "site-444555aaabbb11223",  
    "Description": "VPN Link",  
    "Type": "broadband",  
    "Bandwidth": {  
      "UploadSpeed": 20,  
      "DownloadSpeed": 20  
    },  
    "Provider": "AnyCompany",  
    "CreatedAt": 1575555811.0,  
    "State": "UPDATING"  
  }  
}
```

For more information, see [Working with Links](#) in the *Transit Gateway Network Manager Guide*.

- For API details, see [UpdateLink](#) in *AWS CLI Command Reference*.

update-site

The following code example shows how to use `update-site`.

AWS CLI

To update a site

The following `update-site` example updates the description for site `site-444555aaabbb11223` in the specified global network.

```
aws networkmanager update-site \  
  --global-network-id global-network-01231231231231231 \  
  --site-id site-444555aaabbb11223 \  
  --description "New York Office site" \  
  --region us-west-2
```

Output:

```
{  
  "Site": {  
    "SiteId": "site-444555aaabbb11223",  
    "SiteArn": "arn:aws:networkmanager::123456789012:site/global-  
network-01231231231231231/site-444555aaabbb11223",  
    "GlobalNetworkId": "global-network-01231231231231231",  
    "Description": "New York Office site",  
    "Location": {  
      "Latitude": "40.7128",  
      "Longitude": "-74.0060"  
    },  
    "CreatedAt": 1575554528.0,  
    "State": "UPDATING"  
  }  
}
```

For more information, see [Working with Sites](#) in the *Transit Gateway Network Manager Guide*.

- For API details, see [UpdateSite](#) in *AWS CLI Command Reference*.

Nimble Studio examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Nimble Studio.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

get-eula

The following code example shows how to use `get-eula`.

AWS CLI

To get information about your studio

The following `get-eula` example lists the information about an EULA.

```
aws nimble get-eula \  
  --eula-id "EULAid"
```

Output:

```
{  
  "eula": {  
    "content": "https://www.mozilla.org/en-US/MPL/2.0/",  
    "createdAt": "2021-04-20T16:45:23+00:00",  
    "eulaId": "gJZLygd-Srq_5NNbSfiaLg",  
    "name": "Mozilla-FireFox",  
    "updatedAt": "2021-04-20T16:45:23+00:00"  
  }  
}
```

For more information, see [Accept the EULA](#) in the *Amazon Nimble Studio User Guide*.

- For API details, see [GetEula](#) in *AWS CLI Command Reference*.

get-launch-profile-details

The following code example shows how to use `get-launch-profile-details`.

AWS CLI

To list the available widgets

The following `get-launch-profile-details` example lists the details about a launch profile.

```
aws nimble get-launch-profile-details \  
  --studio-id "StudioID" \  
  --launch-profile-id "LaunchProfileID"
```

Output:

```
{  
  "launchProfile": {  
    "arn": "arn:aws:nimble:us-west-2:123456789012:launch-profile/  
yeG7lDwNQEiwNTRT7DrV7Q",  
    "createdAt": "2022-01-27T21:18:59+00:00",  
    "createdBy": "AROA3002NEHCCYRNDDIFT:i-EXAMPLE11111",  
    "description": "The Launch Profile for the Render workers created by  
StudioBuilder.",  
    "ec2SubnetIds": [  
      "subnet-EXAMPLE11111"  
    ],  
    "launchProfileId": "yeG7lDwNQEiwNTRT7DrV7Q",  
    "launchProfileProtocolVersions": [  
      "2021-03-31"  
    ],  
    "name": "RenderWorker-Default",  
    "state": "READY",  
    "statusCode": "LAUNCH_PROFILE_CREATED",  
    "statusMessage": "Launch Profile has been created",  
    "streamConfiguration": {  
      "clipboardMode": "ENABLED",  
      "ec2InstanceTypes": [  
        "g4dn.4xlarge",  
        "g4dn.8xlarge"  
      ],  
      "maxSessionLengthInMinutes": 690,  
      "maxStoppedSessionLengthInMinutes": 0,  
      "streamingImageIds": [  
        "Cw_jXnp1QcSSXhE2hkNRoQ",  
        "YGXAqgoWTnCNSV8VP20sHQ"  
      ]  
    },  
    "studioComponentIds": [  
      "_hR_-RaAReS0jAnLakbX7Q",  
    ]  
  }  
}
```

```
        "vQ5w_TbIRayPkAZgcbyYRA",
        "ZQuMxN99Qfa_Js6ma9TwdA",
        "45Kj0SPPrzK20yvpCuQ6qw"
    ],
    "tags": {
        "resourceArn": "arn:aws:nimble:us-west-2:123456789012:launch-profile/
yeG71DwNQEiwNTRT7DrV7Q"
    },
    "updatedAt": "2022-01-27T21:19:13+00:00",
    "updatedBy": "AROA3002NEHCCYRNDDIFT:i-00b98256b04d9e989",
    "validationResults": [
        {
            "state": "VALIDATION_SUCCESS",
            "statusCode": "VALIDATION_SUCCESS",
            "statusMessage": "The validation succeeded.",
            "type": "VALIDATE_ACTIVE_DIRECTORY_STUDIO_COMPONENT"
        },
        {
            "state": "VALIDATION_SUCCESS",
            "statusCode": "VALIDATION_SUCCESS",
            "statusMessage": "The validation succeeded.",
            "type": "VALIDATE_SUBNET_ASSOCIATION"
        },
        {
            "state": "VALIDATION_SUCCESS",
            "statusCode": "VALIDATION_SUCCESS",
            "statusMessage": "The validation succeeded.",
            "type": "VALIDATE_NETWORK_ACL_ASSOCIATION"
        },
        {
            "state": "VALIDATION_SUCCESS",
            "statusCode": "VALIDATION_SUCCESS",
            "statusMessage": "The validation succeeded.",
            "type": "VALIDATE_SECURITY_GROUP_ASSOCIATION"
        }
    ]
},
"streamingImages": [
    {
        "arn": "arn:aws:nimble:us-west-2:123456789012:streaming-image/
Cw_jXnp1QcSSXhE2hkNRoQ",
        "description": "Base windows image for NimbleStudio",
        "ec2ImageId": "ami-EXAMPLE11111",
        "eulaIds": [
```



```

        "gJZLygd-Srq_5NNbSfiaLg",
        "ggK2eIw6RQyt8PIee0lD3g",
        "a-D9Wc0VQCKUfxAinCDxaw",
        "RvoNmVXiSrS4LhLTb6ybkw",
        "wtp85BcSTa2NZeNRnMKdjw",
        "Rl-J0fM5S12hyIiwWIV6hw"
    ],
    "name": "NimbleStudioWindowsStreamImage",
    "owner": "amazon",
    "platform": "WINDOWS",
    "state": "READY",
    "streamingImageId": "Cw_jXnp1QcSSXhE2hkNRoQ",
    "tags": {
        "resourceArn": "arn:aws:nimble:us-west-2:123456789012:streaming-
image/Cw_jXnp1QcSSXhE2hkNRoQ"
    }
},
{
    "arn": "arn:aws:nimble:us-west-2:123456789012:streaming-image/
YGXAqgoWTnCNSV8VP20sHQ",
    "description": "Base linux image for NimbleStudio",
    "ec2ImageId": "ami-EXAMPLE11111",
    "eulaIds": [
        "gJZLygd-Srq_5NNbSfiaLg",
        "ggK2eIw6RQyt8PIee0lD3g",
        "a-D9Wc0VQCKUfxAinCDxaw",
        "RvoNmVXiSrS4LhLTb6ybkw",
        "wtp85BcSTa2NZeNRnMKdjw",
        "Rl-J0fM5S12hyIiwWIV6hw"
    ],
    "name": "NimbleStudioLinuxStreamImage",
    "owner": "amazon",
    "platform": "LINUX",
    "state": "READY",
    "streamingImageId": "YGXAqgoWTnCNSV8VP20sHQ",
    "tags": {
        "resourceArn": "arn:aws:nimble:us-west-2:123456789012:streaming-
image/YGXAqgoWTnCNSV8VP20sHQ"
    }
}
],
"studioComponentSummaries": [
    {
        "description": "FSx for Windows",

```

```

        "name": "FSxWindows",
        "studioComponentId": "ZQuMxN99Qfa Js6ma9TwdA",
        "subtype": "AMAZON_FSX_FOR_WINDOWS",
        "type": "SHARED_FILE_SYSTEM"
    },
    {
        "description": "Instance configuration studio component.",
        "name": "InstanceConfiguration",
        "studioComponentId": "vQ5w_TbIRayPkAZgcbyYRA",
        "subtype": "CUSTOM",
        "type": "CUSTOM"
    },
    {
        "name": "ActiveDirectory",
        "studioComponentId": "_hR_-RaAReS0jAnLakbX7Q",
        "subtype": "AWS_MANAGED_MICROSOFT_AD",
        "type": "ACTIVE_DIRECTORY"
    },
    {
        "description": "Render farm running Deadline",
        "name": "RenderFarm",
        "studioComponentId": "45Kj0SPPRzK20yvpCuQ6qw",
        "subtype": "CUSTOM",
        "type": "COMPUTE_FARM"
    }
]
}

```

For more information, see [Creating launch profiles](#) in the *Amazon Nimble Studio User Guide*.

- For API details, see [GetLaunchProfileDetails](#) in *AWS CLI Command Reference*.

get-launch-profile

The following code example shows how to use `get-launch-profile`.

AWS CLI

To list the available widgets

The following `get-launch-profile` example lists information about a launch profile.

```
aws nimble get-launch-profile \
```

```
--studio-id "StudioID" \  
--launch-profile-id "LaunchProfileID"
```

Output:

```
{  
  "launchProfile": {  
    "arn": "arn:aws:nimble:us-west-2:123456789012:launch-profile/  
yeG7lDwNQEiwNTRT7DrV7Q",  
    "createdAt": "2022-01-27T21:18:59+00:00",  
    "createdBy": "ARO3002NEHCCYRNDIIFT:i-EXAMPLE11111",  
    "description": "The Launch Profile for the Render workers created by  
StudioBuilder.",  
    "ec2SubnetIds": [  
      "subnet-EXAMPLE11111"  
    ],  
    "launchProfileId": "yeG7lDwNQEiwNTRT7DrV7Q",  
    "launchProfileProtocolVersions": [  
      "2021-03-31"  
    ],  
    "name": "RenderWorker-Default",  
    "state": "READY",  
    "statusCode": "LAUNCH_PROFILE_CREATED",  
    "statusMessage": "Launch Profile has been created",  
    "streamConfiguration": {  
      "clipboardMode": "ENABLED",  
      "ec2InstanceTypes": [  
        "g4dn.4xlarge",  
        "g4dn.8xlarge"  
      ],  
      "maxSessionLengthInMinutes": 690,  
      "maxStoppedSessionLengthInMinutes": 0,  
      "streamingImageIds": [  
        "Cw_jXnp1QcSSXhE2hkNRoQ",  
        "YGXAqgoWTnCNSV8VP20sHQ"  
      ]  
    },  
    "studioComponentIds": [  
      "_hR_-RaAReS0jAnLakbX7Q",  
      "vQ5w_TbIRayPkAZgcbyYRA",  
      "ZQuMxN99Qfa_Js6ma9TwdA",  
      "45Kj0SPPRzK20yvpCuQ6qw"  
    ]  
  },  
}
```

```
"tags": {},
"updatedAt": "2022-01-27T21:19:13+00:00",
"updatedBy": "AROA3002NEHCCYRNDDIFT:i-00b98256b04d9e989",
"validationResults": [
  {
    "state": "VALIDATION_SUCCESS",
    "statusCode": "VALIDATION_SUCCESS",
    "statusMessage": "The validation succeeded.",
    "type": "VALIDATE_ACTIVE_DIRECTORY_STUDIO_COMPONENT"
  },
  {
    "state": "VALIDATION_SUCCESS",
    "statusCode": "VALIDATION_SUCCESS",
    "statusMessage": "The validation succeeded.",
    "type": "VALIDATE_SUBNET_ASSOCIATION"
  },
  {
    "state": "VALIDATION_SUCCESS",
    "statusCode": "VALIDATION_SUCCESS",
    "statusMessage": "The validation succeeded.",
    "type": "VALIDATE_NETWORK_ACL_ASSOCIATION"
  },
  {
    "state": "VALIDATION_SUCCESS",
    "statusCode": "VALIDATION_SUCCESS",
    "statusMessage": "The validation succeeded.",
    "type": "VALIDATE_SECURITY_GROUP_ASSOCIATION"
  }
]
}
```

For more information, see [Creating launch profiles](#) in the *Amazon Nimble Studio User Guide*.

- For API details, see [GetLaunchProfile](#) in *AWS CLI Command Reference*.

get-studio

The following code example shows how to use `get-studio`.

AWS CLI

To get information about your studio

The following `get-studio` example lists the studios in your AWS account.

```
aws nimble get-studio \  
  --studio-id "StudioID"
```

Output:

```
{  
  "studio": {  
    "adminRoleArn": "arn:aws:iam::123456789012:role/studio-admin-role",  
    "arn": "arn:aws:nimble:us-west-2:123456789012:studio/stid-EXAMPLE11111",  
    "createdAt": "2022-01-27T20:29:35+00:00",  
    "displayName": "studio-name",  
    "homeRegion": "us-west-2",  
    "ssoClientId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
    "state": "READY",  
    "statusCode": "STUDIO_CREATED",  
    "statusMessage": "The studio has been created successfully ",  
    "studioEncryptionConfiguration": {  
      "keyType": "AWS_OWNED_KEY"  
    },  
    "studioId": "us-west-2:stid-EXAMPLE11111",  
    "studioName": "studio-name",  
    "studioUrl": "https://studio-name.nimblestudio.us-west-2.amazonaws.com",  
    "tags": {},  
    "updatedAt": "2022-01-27T20:29:37+00:00",  
    "userRoleArn": "arn:aws:iam::123456789012:role/studio-user-role"  
  }  
}
```

For more information, see [What is Amazon Nimble Studio?](#) in the *Amazon Nimble Studio User Guide*.

- For API details, see [GetStudio](#) in *AWS CLI Command Reference*.

list-eula-acceptances

The following code example shows how to use `list-eula-acceptances`.

AWS CLI

To list the available widgets

The following `list-eula-acceptances` example lists the accepted EULAs in your AWS account.

```
aws nimble list-eula-acceptances \  
  --studio-id "StudioID"
```

Output:

```
{  
  "eulaAcceptances": [  
    {  
      "acceptedAt": "2022-01-28T17:44:35+00:00",  
      "acceptedBy": "92677b4b19-e9fd012a-94ad-4f16-9866-c69a63ab6486",  
      "accepteeId": "us-west-2:stid-nyoqq12fteqy1x48",  
      "eulaAcceptanceId": "V0JlpZQaSx6yHcUuX0qfQw",  
      "eulaId": "Rl-J0fM5S12hyIiwWIV6hw"  
    },  
    {  
      "acceptedAt": "2022-01-28T17:44:35+00:00",  
      "acceptedBy": "92677b4b19-e9fd012a-94ad-4f16-9866-c69a63ab6486",  
      "accepteeId": "us-west-2:stid-nyoqq12fteqy1x48",  
      "eulaAcceptanceId": "YY_uDFW-SVibc627qbug0Q",  
      "eulaId": "RvoNmVXiSrS4LhLTb6ybkw"  
    },  
    {  
      "acceptedAt": "2022-01-28T17:44:35+00:00",  
      "acceptedBy": "92677b4b19-e9fd012a-94ad-4f16-9866-c69a63ab6486",  
      "accepteeId": "us-west-2:stid-nyoqq12fteqy1x48",  
      "eulaAcceptanceId": "ov087PnhQ4-MpttiL5uN6Q",  
      "eulaId": "a-D9Wc0VQCKUfxAinCDxaw"  
    },  
    {  
      "acceptedAt": "2022-01-28T17:44:35+00:00",  
      "acceptedBy": "92677b4b19-e9fd012a-94ad-4f16-9866-c69a63ab6486",  
      "accepteeId": "us-west-2:stid-nyoqq12fteqy1x48",  
      "eulaAcceptanceId": "5YeXje4yR0amuTESGvqIAQ",  
      "eulaId": "gJZLygd-Srq_5NNbSfiaLg"  
    },  
    {  
      "acceptedAt": "2022-01-28T17:44:35+00:00",  
      "acceptedBy": "92677b4b19-e9fd012a-94ad-4f16-9866-c69a63ab6486",  
      "accepteeId": "us-west-2:stid-nyoqq12fteqy1x48",  
      "eulaAcceptanceId": "W1sIn8PtScqeJEn8sxxhgw",  
    }  
  ]  
}
```

```

        "eulaId": "ggK2eIw6RQyt8PIee01D3g"
    },
    {
        "acceptedAt": "2022-01-28T17:44:35+00:00",
        "acceptedBy": "92677b4b19-e9fd012a-94ad-4f16-9866-c69a63ab6486",
        "accepteeId": "us-west-2:stid-nyoqq12fteqy1x48",
        "eulaAcceptanceId": "Zq9KNEQPRMWJ7FolSoQgUA",
        "eulaId": "wtp85BcSTa2NZeNRnMKdjw"
    }
]
}

```

For more information, see [Accept the EULA](#) in the *Amazon Nimble Studio User Guide*.

- For API details, see [ListEulaAcceptances](#) in *AWS CLI Command Reference*.

list-eulas

The following code example shows how to use `list-eulas`.

AWS CLI

To list the available widgets

The following `list-eulas` example lists the EULAs in your AWS account.

```
aws nimble list-eulas
```

Output:

```

{
  "eulas": [
    {
      "content": "https://www.mozilla.org/en-US/MPL/2.0/",
      "createdAt": "2021-04-20T16:45:23+00:00",
      "eulaId": "gJZLygd-Srq_5NNbSfiaLg",
      "name": "Mozilla-FireFox",
      "updatedAt": "2021-04-20T16:45:23+00:00"
    },
    {
      "content": "https://www.awsthinkbox.com/end-user-license-agreement",
      "createdAt": "2021-04-20T16:45:24+00:00",
      "eulaId": "RvoNmVXiSrS4LhLTb6ybkw",

```

```
    "name": "Thinkbox-Deadline",
    "updatedAt": "2021-04-20T16:45:24+00:00"
  },
  {
    "content": "https://www.videolan.org/legal.html",
    "createdAt": "2021-04-20T16:45:24+00:00",
    "eulaId": "R1-J0fM5S12hyIiwWIV6hw",
    "name": "Videolan-VLC",
    "updatedAt": "2021-04-20T16:45:24+00:00"
  },
  {
    "content": "https://code.visualstudio.com/license",
    "createdAt": "2021-04-20T16:45:23+00:00",
    "eulaId": "ggK2eIw6RQyt8PIee0lD3g",
    "name": "Microsoft-VSCode",
    "updatedAt": "2021-04-20T16:45:23+00:00"
  },
  {
    "content": "https://darbyjohnston.github.io/DJV/legal.html#License",
    "createdAt": "2021-04-20T16:45:23+00:00",
    "eulaId": "wtp85BcSTa2NZeNRnMKdju",
    "name": "DJV-DJV",
    "updatedAt": "2021-04-20T16:45:23+00:00"
  },
  {
    "content": "https://www.sidefx.com/legal/license-agreement/",
    "createdAt": "2021-04-20T16:45:24+00:00",
    "eulaId": "uu2VDLo-QJeIGWwLBae_UA",
    "name": "SideFX-Houdini",
    "updatedAt": "2021-04-20T16:45:24+00:00"
  },
  {
    "content": "https://www.chaosgroup.com/eula",
    "createdAt": "2021-04-20T16:45:23+00:00",
    "eulaId": "L0HS4P3CRYKVXc2J2L07Vw",
    "name": "ChaosGroup-Vray",
    "updatedAt": "2021-04-20T16:45:23+00:00"
  },
  {
    "content": "https://www.foundry.com/eula",
    "createdAt": "2021-04-20T16:45:23+00:00",
    "eulaId": "SAuhfHmMSAeUuq3wsMiMlw",
    "name": "Foundry-Nuke",
    "updatedAt": "2021-04-20T16:45:23+00:00"
  }
```



```
    },
    {
      "content": "https://download.blender.org/release/GPL3-license.txt",
      "createdAt": "2021-04-20T16:45:23+00:00",
      "eulaId": "a-D9Wc0VQCKUfxAinCDxaw",
      "name": "BlenderFoundation-Blender",
      "updatedAt": "2021-04-20T16:45:23+00:00"
    }
  ]
}
```

For more information, see [Accept the EULA](#) in the *Amazon Nimble Studio User Guide*.

- For API details, see [ListEulas](#) in *AWS CLI Command Reference*.

list-launch-profiles

The following code example shows how to use `list-launch-profiles`.

AWS CLI

To list the available widgets

The following `list-launch-profiles` example lists the launch profiles in your AWS account.

```
aws nimble list-launch-profiles \
  --studio-id "StudioID"
```

Output:

```
{
  "launchProfiles": [
    {
      "arn": "arn:aws:nimble:us-west-2:123456789012:launch-profile/
yeG7lDwNQEiwNTRT7DrV7Q",
      "createdAt": "2022-01-27T21:18:59+00:00",
      "createdBy": "AROA3002NEHCCYRNDDIFT:i-EXAMPLE11111",
      "description": "The Launch Profile for the Render workers created by
StudioBuilder.",
      "ec2SubnetIds": [
        "subnet-EXAMPLE11111"
      ],
      "launchProfileId": "yeG7lDwNQEiwNTRT7DrV7Q",
```

```
"launchProfileProtocolVersions": [
  "2021-03-31"
],
"name": "RenderWorker-Default",
"state": "READY",
"statusCode": "LAUNCH_PROFILE_CREATED",
"statusMessage": "Launch Profile has been created",
"streamConfiguration": {
  "clipboardMode": "ENABLED",
  "ec2InstanceTypes": [
    "g4dn.4xlarge",
    "g4dn.8xlarge"
  ],
  "maxSessionLengthInMinutes": 690,
  "maxStoppedSessionLengthInMinutes": 0,
  "streamingImageIds": [
    "Cw_jXnp1QcSSXhE2hkNRoQ",
    "YGXAqgoWTnCNSV8VP20sHQ"
  ]
},
"studioComponentIds": [
  "_hR_-RaAReS0jAnLakbX7Q",
  "vQ5w_TbIRayPkAZgcbyYRA",
  "ZQuMxN99Qfa_Js6ma9TwdA",
  "45Kj0SPPRzK20yvpCuQ6qw"
],
"tags": {},
"updatedAt": "2022-01-27T21:19:13+00:00",
"updatedBy": "AROA3002NEHCCYRNDDIFT:i-EXAMPLE11111",
"validationResults": [
  {
    "state": "VALIDATION_SUCCESS",
    "statusCode": "VALIDATION_SUCCESS",
    "statusMessage": "The validation succeeded.",
    "type": "VALIDATE_ACTIVE_DIRECTORY_STUDIO_COMPONENT"
  },
  {
    "state": "VALIDATION_SUCCESS",
    "statusCode": "VALIDATION_SUCCESS",
    "statusMessage": "The validation succeeded.",
    "type": "VALIDATE_SUBNET_ASSOCIATION"
  },
  {
    "state": "VALIDATION_SUCCESS",
```

```

        "statusCode": "VALIDATION_SUCCESS",
        "statusMessage": "The validation succeeded.",
        "type": "VALIDATE_NETWORK_ACL_ASSOCIATION"
    },
    {
        "state": "VALIDATION_SUCCESS",
        "statusCode": "VALIDATION_SUCCESS",
        "statusMessage": "The validation succeeded.",
        "type": "VALIDATE_SECURITY_GROUP_ASSOCIATION"
    }
]
},
{
    "arn": "arn:aws:nimble:us-west-2:123456789012:launch-profile/
jDCIm1jRSaa9e44PZ3w7gg",
    "createdAt": "2022-01-27T21:19:26+00:00",
    "createdBy": "AROA3002NEHCCYRNDDIFT:i-EXAMPLE11111",
    "description": "This Workstation Launch Profile was created by
StudioBuilder",
    "ec2SubnetIds": [
        "subnet-046f4205ae535b2cc"
    ],
    "launchProfileId": "jDCIm1jRSaa9e44PZ3w7gg",
    "launchProfileProtocolVersions": [
        "2021-03-31"
    ],
    "name": "Workstation-Default",
    "state": "READY",
    "statusCode": "LAUNCH_PROFILE_CREATED",
    "statusMessage": "Launch Profile has been created",
    "streamConfiguration": {
        "clipboardMode": "ENABLED",
        "ec2InstanceTypes": [
            "g4dn.4xlarge",
            "g4dn.8xlarge"
        ],
        "maxSessionLengthInMinutes": 690,
        "maxStoppedSessionLengthInMinutes": 0,
        "streamingImageIds": [
            "Cw_jXnp1QcSSXhE2hkNRoQ",
            "YGXAqgoWTnCNSV8VP20sHQ"
        ]
    },
    "studioComponentIds": [

```

```

    "_hR_-RaAReS0jAnLakbX7Q",
    "vQ5w_TbIRayPkAZgcbyYRA",
    "ZQuMxN99Qfa_Js6ma9TwdA",
    "yJSbsHXAQYwk9FXLNusX1Q",
    "45Kj0SPPRzK20yvpCuQ6qw"
  ],
  "tags": {},
  "updatedAt": "2022-01-27T21:19:40+00:00",
  "updatedBy": "AR0A3002NEHCCYRNDDIFT:i-EXAMPLE11111",
  "validationResults": [
    {
      "state": "VALIDATION_SUCCESS",
      "statusCode": "VALIDATION_SUCCESS",
      "statusMessage": "The validation succeeded.",
      "type": "VALIDATE_ACTIVE_DIRECTORY_STUDIO_COMPONENT"
    },
    {
      "state": "VALIDATION_SUCCESS",
      "statusCode": "VALIDATION_SUCCESS",
      "statusMessage": "The validation succeeded.",
      "type": "VALIDATE_SUBNET_ASSOCIATION"
    },
    {
      "state": "VALIDATION_SUCCESS",
      "statusCode": "VALIDATION_SUCCESS",
      "statusMessage": "The validation succeeded.",
      "type": "VALIDATE_NETWORK_ACL_ASSOCIATION"
    },
    {
      "state": "VALIDATION_SUCCESS",
      "statusCode": "VALIDATION_SUCCESS",
      "statusMessage": "The validation succeeded.",
      "type": "VALIDATE_SECURITY_GROUP_ASSOCIATION"
    }
  ]
}

```

For more information, see [Creating launch profiles](#) in the *Amazon Nimble Studio User Guide*.

- For API details, see [ListLaunchProfiles](#) in *AWS CLI Command Reference*.

list-studio-components

The following code example shows how to use `list-studio-components`.

AWS CLI

To list the available widgets

The following `list-studio-components` example lists the studio components in your AWS account.

```
aws nimble list-studio-components \
  --studio-id "StudioID"
```

Output:

```
{
  "studioComponents": [
    {
      "arn": "arn:aws:nimble:us-west-2:123456789012:studio-component/
ZQuMxN99Qfa_Js6ma9TwdA",
      "configuration": {
        "sharedFileSystemConfiguration": {
          "fileSystemId": "fs-EXAMPLE11111",
          "linuxMountPoint": "/mnt/fsxshare",
          "shareName": "share",
          "windowsMountDrive": "Z"
        }
      },
      "createdAt": "2022-01-27T21:15:34+00:00",
      "createdBy": "AROA3002NEHCCYRNDIFT:i-EXAMPLE11111",
      "description": "FSx for Windows",
      "ec2SecurityGroupIds": [
        "sg-EXAMPLE11111"
      ],
      "name": "FSxWindows",
      "state": "READY",
      "statusCode": "STUDIO_COMPONENT_CREATED",
      "statusMessage": "Studio Component has been created",
      "studioComponentId": "ZQuMxN99Qfa_Js6ma9TwdA",
      "subtype": "AMAZON_FSX_FOR_WINDOWS",
      "tags": {},
      "type": "SHARED_FILE_SYSTEM",
    }
  ]
}
```

```
        "updatedAt": "2022-01-27T21:15:35+00:00",
        "updatedBy": "AROA3002NEHCCYRNDDIFT:i-EXAMPLE11111"
    },
    ...
}
```

For more information, see [How StudioBuilder works with Amazon Nimble Studio](#) in the *Amazon Nimble Studio User Guide*.

- For API details, see [ListStudioComponents](#) in *AWS CLI Command Reference*.

list-studio-members

The following code example shows how to use `list-studio-members`.

AWS CLI

To list the available widgets

The following `list-studio-members` example lists the available studio members in your AWS account.

```
aws nimble list-studio-members \
  --studio-id "StudioID"
```

Output:

```
{
  "members": [
    {
      "identityStoreId": "d-EXAMPLE11111",
      "persona": "ADMINISTRATOR",
      "principalId": "EXAMPLE11111-e9fd012a-94ad-4f16-9866-c69a63ab6486"
    }
  ]
}
```

For more information, see [Adding studio users](#) in the *Amazon Nimble Studio User Guide*.

- For API details, see [ListStudioMembers](#) in *AWS CLI Command Reference*.

list-studios

The following code example shows how to use `list-studios`.

AWS CLI

To list your studios

The following `list-studios` example lists the studios in your AWS account.

```
aws nimble list-studios
```

Output:

```
{
  "studios": [
    {
      "adminRoleArn": "arn:aws:iam::123456789012:role/studio-admin-role",
      "arn": "arn:aws:nimble:us-west-2:123456789012:studio/studio-id",
      "createdAt": "2022-01-27T20:29:35+00:00",
      "displayName": "studio-name",
      "homeRegion": "us-west-2",
      "ssoClientId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "state": "READY",
      "statusCode": "STUDIO_CREATED",
      "statusMessage": "The studio has been created successfully ",
      "studioEncryptionConfiguration": {
        "keyType": "AWS_OWNED_KEY"
      },
      "studioId": "us-west-2:studio-id",
      "studioName": "studio-name",
      "studioUrl": "https://studio-name.nimblestudio.us-west-2.amazonaws.com",
      "tags": {},
      "updatedAt": "2022-01-27T20:29:37+00:00",
      "userRoleArn": "arn:aws:iam::123456789012:role/studio-user-role"
    }
  ]
}
```

For more information, see [What is Amazon Nimble Studio?](#) in the *Amazon Nimble Studio User Guide*.

- For API details, see [ListStudios](#) in *AWS CLI Command Reference*.

OpenSearch Service examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with OpenSearch Service.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-elasticsearch-domain

The following code example shows how to use `create-elasticsearch-domain`.

AWS CLI

To create an Amazon Elasticsearch Service domain

The following `create-elasticsearch-domain` command creates a new Amazon Elasticsearch Service domain within a VPC and restricts access to a single user. Amazon ES infers the VPC ID from the specified subnet and security group IDs.

```
aws es create-elasticsearch-domain \
  --domain-name vpc-cli-example \
  --elasticsearch-version 6.2 \
  --elasticsearch-cluster-config
InstanceType=m4.large.elasticsearch,InstanceCount=1 \
  --ebs-options EBSEnabled=true,VolumeType=standard,VolumeSize=10 \
  --access-policies '{"Version": "2012-10-17", "Statement": [ { "Effect":
"Allow", "Principal": {"AWS": "arn:aws:iam::123456789012:root" }, "Action": "es:*",
"Resource": "arn:aws:es:us-west-1:123456789012:domain/vpc-cli-example/*" } ] }' \
```



```
--vpc-options SubnetIds=subnet-1a2a3a4a,SecurityGroupIds=sg-2a3a4a5a
```

Output:

```
{
  "DomainStatus": {
    "ElasticsearchClusterConfig": {
      "DedicatedMasterEnabled": false,
      "InstanceCount": 1,
      "ZoneAwarenessEnabled": false,
      "InstanceType": "m4.large.elasticsearch"
    },
    "DomainId": "123456789012/vpc-cli-example",
    "CognitoOptions": {
      "Enabled": false
    },
    "VPCOptions": {
      "SubnetIds": [
        "subnet-1a2a3a4a"
      ],
      "VPCId": "vpc-3a4a5a6a",
      "SecurityGroupIds": [
        "sg-2a3a4a5a"
      ],
      "AvailabilityZones": [
        "us-west-1c"
      ]
    },
    "Created": true,
    "Deleted": false,
    "EBSOptions": {
      "VolumeSize": 10,
      "VolumeType": "standard",
      "EBSEnabled": true
    },
    "Processing": true,
    "DomainName": "vpc-cli-example",
    "SnapshotOptions": {
      "AutomatedSnapshotStartHour": 0
    },
    "ElasticsearchVersion": "6.2",
    "AccessPolicies": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",\"Principal\":{\"AWS\":\"arn:aws:iam::123456789012:root\"},\"Action\":
```

```

\ "es:*\" , \ "Resource\" : \ "arn:aws:es:us-west-1:123456789012:domain/vpc-cli-example/*
\ "}]}" ,
  "AdvancedOptions": {
    "rest.action.multi.allow_explicit_index": "true"
  },
  "EncryptionAtRestOptions": {
    "Enabled": false
  },
  "ARN": "arn:aws:es:us-west-1:123456789012:domain/vpc-cli-example"
}
}

```

For more information, see [Creating and Managing Amazon Elasticsearch Service Domains](#) in the *Amazon Elasticsearch Service Developer Guide*.

- For API details, see [CreateElasticsearchDomain](#) in *AWS CLI Command Reference*.

describe-elasticsearch-domain-config

The following code example shows how to use `describe-elasticsearch-domain-config`.

AWS CLI

To get domain configuration details

The following `describe-elasticsearch-domain-config` example provides configuration details for a given domain, along with status information for each individual domain component.

```

aws es describe-elasticsearch-domain-config \
  --domain-name cli-example

```

Output:

```

{
  "DomainConfig": {
    "ElasticsearchVersion": {
      "Options": "7.4",
      "Status": {
        "CreationDate": 1589395034.946,
        "UpdateDate": 1589395827.325,
        "UpdateVersion": 8,
        "State": "Active",

```

```

        "PendingDeletion": false
    }
},
"ElasticsearchClusterConfig": {
    "Options": {
        "InstanceType": "c5.large.elasticsearch",
        "InstanceCount": 1,
        "DedicatedMasterEnabled": true,
        "ZoneAwarenessEnabled": false,
        "DedicatedMasterType": "c5.large.elasticsearch",
        "DedicatedMasterCount": 3,
        "WarmEnabled": true,
        "WarmType": "ultrawarm1.medium.elasticsearch",
        "WarmCount": 2
    },
    "Status": {
        "CreationDate": 1589395034.946,
        "UpdateDate": 1589395827.325,
        "UpdateVersion": 8,
        "State": "Active",
        "PendingDeletion": false
    }
},
"EBSOptions": {
    "Options": {
        "EBSEnabled": true,
        "VolumeType": "gp2",
        "VolumeSize": 10
    },
    "Status": {
        "CreationDate": 1589395034.946,
        "UpdateDate": 1589395827.325,
        "UpdateVersion": 8,
        "State": "Active",
        "PendingDeletion": false
    }
},
"AccessPolicies": {
    "Options": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",\"Principal\":{\"AWS\":\"*\"},\"Action\":\"es:*\",\"Resource\":\"arn:aws:es:us-east-1:123456789012:domain/cli-example/*\"}]}",
    "Status": {
        "CreationDate": 1589395034.946,
        "UpdateDate": 1589395827.325,

```

```
        "UpdateVersion": 8,
        "State": "Active",
        "PendingDeletion": false
    }
},
"SnapshotOptions": {
    "Options": {
        "AutomatedSnapshotStartHour": 0
    },
    "Status": {
        "CreationDate": 1589395034.946,
        "UpdateDate": 1589395827.325,
        "UpdateVersion": 8,
        "State": "Active",
        "PendingDeletion": false
    }
},
"VPCOptions": {
    "Options": {},
    "Status": {
        "CreationDate": 1591210426.162,
        "UpdateDate": 1591210426.162,
        "UpdateVersion": 18,
        "State": "Active",
        "PendingDeletion": false
    }
},
"CognitoOptions": {
    "Options": {
        "Enabled": false
    },
    "Status": {
        "CreationDate": 1591210426.163,
        "UpdateDate": 1591210426.163,
        "UpdateVersion": 18,
        "State": "Active",
        "PendingDeletion": false
    }
},
"EncryptionAtRestOptions": {
    "Options": {
        "Enabled": true,
        "KmsKeyId": "arn:aws:kms:us-
east-1:123456789012:key/1a2a3a4a-1a2a-1a2a-1a2a-1a2a3a4a5a6a"
```

```
    },
    "Status": {
      "CreationDate": 1589395034.946,
      "UpdateDate": 1589395827.325,
      "UpdateVersion": 8,
      "State": "Active",
      "PendingDeletion": false
    }
  },
  "NodeToNodeEncryptionOptions": {
    "Options": {
      "Enabled": true
    },
    "Status": {
      "CreationDate": 1589395034.946,
      "UpdateDate": 1589395827.325,
      "UpdateVersion": 8,
      "State": "Active",
      "PendingDeletion": false
    }
  },
  "AdvancedOptions": {
    "Options": {
      "rest.action.multi.allow_explicit_index": "true"
    },
    "Status": {
      "CreationDate": 1589395034.946,
      "UpdateDate": 1589395827.325,
      "UpdateVersion": 8,
      "State": "Active",
      "PendingDeletion": false
    }
  },
  "LogPublishingOptions": {
    "Options": {},
    "Status": {
      "CreationDate": 1591210426.164,
      "UpdateDate": 1591210426.164,
      "UpdateVersion": 18,
      "State": "Active",
      "PendingDeletion": false
    }
  },
  "DomainEndpointOptions": {
```

```

    "Options": {
      "EnforceHTTPS": true,
      "TLSSecurityPolicy": "Policy-Min-TLS-1-0-2019-07"
    },
    "Status": {
      "CreationDate": 1589395034.946,
      "UpdateDate": 1589395827.325,
      "UpdateVersion": 8,
      "State": "Active",
      "PendingDeletion": false
    }
  },
  "AdvancedSecurityOptions": {
    "Options": {
      "Enabled": true,
      "InternalUserDatabaseEnabled": true
    },
    "Status": {
      "CreationDate": 1589395034.946,
      "UpdateDate": 1589827485.577,
      "UpdateVersion": 14,
      "State": "Active",
      "PendingDeletion": false
    }
  }
}
}
}
}
}

```

For more information, see [Creating and Managing Amazon Elasticsearch Service Domains](#) in the *Amazon Elasticsearch Service Developer Guide*.

- For API details, see [DescribeElasticsearchDomainConfig](#) in *AWS CLI Command Reference*.

describe-elasticsearch-domain

The following code example shows how to use describe-elasticsearch-domain.

AWS CLI

To get details for a single domain

The following describe-elasticsearch-domain example provides configuration details for a given domain.

```
aws es describe-elasticsearch-domain \
  --domain-name cli-example
```

Output:

```
{
  "DomainStatus": {
    "DomainId": "123456789012/cli-example",
    "DomainName": "cli-example",
    "ARN": "arn:aws:es:us-east-1:123456789012:domain/cli-example",
    "Created": true,
    "Deleted": false,
    "Endpoint": "search-cli-example-1a2a3a4a5a6a7a8a9a0a.us-
east-1.es.amazonaws.com",
    "Processing": false,
    "UpgradeProcessing": false,
    "ElasticsearchVersion": "7.4",
    "ElasticsearchClusterConfig": {
      "InstanceType": "c5.large.elasticsearch",
      "InstanceCount": 1,
      "DedicatedMasterEnabled": true,
      "ZoneAwarenessEnabled": false,
      "DedicatedMasterType": "c5.large.elasticsearch",
      "DedicatedMasterCount": 3,
      "WarmEnabled": true,
      "WarmType": "ultrawarm1.medium.elasticsearch",
      "WarmCount": 2
    },
    "EBSOptions": {
      "EBSEnabled": true,
      "VolumeType": "gp2",
      "VolumeSize": 10
    },
    "AccessPolicies": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":"
    "\":\"Allow\",\"Principal\":{\"AWS\":\"*\"},\"Action\":\"es:*\",\"Resource\":"
    "\":\"arn:aws:es:us-east-1:123456789012:domain/cli-example/*\"}]}",
    "SnapshotOptions": {
      "AutomatedSnapshotStartHour": 0
    },
    "CognitoOptions": {
      "Enabled": false
    },
    "EncryptionAtRestOptions": {
```

```

    "Enabled": true,
    "KmsKeyId": "arn:aws:kms:us-
east-1:123456789012:key/1a2a3a4a-1a2a-1a2a-1a2a-1a2a3a4a5a6a"
  },
  "NodeToNodeEncryptionOptions": {
    "Enabled": true
  },
  "AdvancedOptions": {
    "rest.action.multi.allow_explicit_index": "true"
  },
  "ServiceSoftwareOptions": {
    "CurrentVersion": "R20200522",
    "NewVersion": "",
    "UpdateAvailable": false,
    "Cancelable": false,
    "UpdateStatus": "COMPLETED",
    "Description": "There is no software update available for this domain.",
    "AutomatedUpdateDate": 0.0
  },
  "DomainEndpointOptions": {
    "EnforceHTTPS": true,
    "TLSSecurityPolicy": "Policy-Min-TLS-1-0-2019-07"
  },
  "AdvancedSecurityOptions": {
    "Enabled": true,
    "InternalUserDatabaseEnabled": true
  }
}

```

For more information, see [Creating and Managing Amazon Elasticsearch Service Domains](#) in the *Amazon Elasticsearch Service Developer Guide*.

- For API details, see [DescribeElasticsearchDomain](#) in *AWS CLI Command Reference*.

describe-elasticsearch-domains

The following code example shows how to use describe-elasticsearch-domains.

AWS CLI

To get details for one or more domains

The following `describe-elasticsearch-domains` example provides configuration details for one or more domains.

```
aws es describe-elasticsearch-domains \  
  --domain-names cli-example-1 cli-example-2
```

Output:

```
{  
  "DomainStatusList": [{  
    "DomainId": "123456789012/cli-example-1",  
    "DomainName": "cli-example-1",  
    "ARN": "arn:aws:es:us-east-1:123456789012:domain/cli-example-1",  
    "Created": true,  
    "Deleted": false,  
    "Endpoint": "search-cli-example-1-1a2a3a4a5a6a7a8a9a0a.us-  
east-1.es.amazonaws.com",  
    "Processing": false,  
    "UpgradeProcessing": false,  
    "ElasticsearchVersion": "7.4",  
    "ElasticsearchClusterConfig": {  
      "InstanceType": "c5.large.elasticsearch",  
      "InstanceCount": 1,  
      "DedicatedMasterEnabled": true,  
      "ZoneAwarenessEnabled": false,  
      "DedicatedMasterType": "c5.large.elasticsearch",  
      "DedicatedMasterCount": 3,  
      "WarmEnabled": true,  
      "WarmType": "ultrawarm1.medium.elasticsearch",  
      "WarmCount": 2  
    },  
    "EBSOptions": {  
      "EBSEnabled": true,  
      "VolumeType": "gp2",  
      "VolumeSize": 10  
    },  
    "AccessPolicies": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",\"Principal\":{\"AWS\":\"*\"},\"Action\":\"es:*\",\"Resource\":\"arn:aws:es:us-east-1:123456789012:domain/cli-example-1/*\"}]}",  
    "SnapshotOptions": {  
      "AutomatedSnapshotStartHour": 0  
    },  
    "CognitoOptions": {
```

```

        "Enabled": false
    },
    "EncryptionAtRestOptions": {
        "Enabled": true,
        "KmsKeyId": "arn:aws:kms:us-
east-1:123456789012:key/1a2a3a4a-1a2a-1a2a-1a2a-1a2a3a4a5a6a"
    },
    "NodeToNodeEncryptionOptions": {
        "Enabled": true
    },
    "AdvancedOptions": {
        "rest.action.multi.allow_explicit_index": "true"
    },
    "ServiceSoftwareOptions": {
        "CurrentVersion": "R20200522",
        "NewVersion": "",
        "UpdateAvailable": false,
        "Cancellable": false,
        "UpdateStatus": "COMPLETED",
        "Description": "There is no software update available for this
domain.",
        "AutomatedUpdateDate": 0.0
    },
    "DomainEndpointOptions": {
        "EnforceHTTPS": true,
        "TLSSecurityPolicy": "Policy-Min-TLS-1-0-2019-07"
    },
    "AdvancedSecurityOptions": {
        "Enabled": true,
        "InternalUserDatabaseEnabled": true
    }
},
{
    "DomainId": "123456789012/cli-example-2",
    "DomainName": "cli-example-2",
    "ARN": "arn:aws:es:us-east-1:123456789012:domain/cli-example-2",
    "Created": true,
    "Deleted": false,
    "Processing": true,
    "UpgradeProcessing": false,
    "ElasticsearchVersion": "7.4",
    "ElasticsearchClusterConfig": {
        "InstanceType": "r5.large.elasticsearch",
        "InstanceCount": 1,

```

```

        "DedicatedMasterEnabled": false,
        "ZoneAwarenessEnabled": false,
        "WarmEnabled": false
    },
    "EBSOptions": {
        "EBSEnabled": true,
        "VolumeType": "gp2",
        "VolumeSize": 10
    },
    "AccessPolicies": "{\\"Version\\":\\"2012-10-17\\",\\"Statement\\":[{\\"Effect
\\":\\"Deny\\",\\"Principal\\":{\\"AWS\\":\\"*\\"},\\"Action\\":\\"es:*\\",\\"Resource\\":
\\"arn:aws:es:us-east-1:123456789012:domain/cli-example-2/*\\"}]}",
    "SnapshotOptions": {
        "AutomatedSnapshotStartHour": 0
    },
    "CognitoOptions": {
        "Enabled": false
    },
    "EncryptionAtRestOptions": {
        "Enabled": false
    },
    "NodeToNodeEncryptionOptions": {
        "Enabled": false
    },
    "AdvancedOptions": {
        "rest.action.multi.allow_explicit_index": "true"
    },
    "ServiceSoftwareOptions": {
        "CurrentVersion": "",
        "NewVersion": "",
        "UpdateAvailable": false,
        "Cancellable": false,
        "UpdateStatus": "COMPLETED",
        "Description": "There is no software update available for this
domain.",
        "AutomatedUpdateDate": 0.0
    },
    "DomainEndpointOptions": {
        "EnforceHTTPS": false,
        "TLSSecurityPolicy": "Policy-Min-TLS-1-0-2019-07"
    },
    "AdvancedSecurityOptions": {
        "Enabled": false,
        "InternalUserDatabaseEnabled": false
    }

```

```
    }  
  }  
]  
}
```

For more information, see [Creating and Managing Amazon Elasticsearch Service Domains](#) in the *Amazon Elasticsearch Service Developer Guide*.

- For API details, see [DescribeElasticsearchDomains](#) in *AWS CLI Command Reference*.

describe-reserved-elasticsearch-instances

The following code example shows how to use `describe-reserved-elasticsearch-instances`.

AWS CLI

To view all reserved instances

The following `describe-elasticsearch-domains` example provides a summary of all instances you have reserved in a region.

```
aws es describe-reserved-elasticsearch-instances
```

Output:

```
{  
  "ReservedElasticsearchInstances": [{  
    "FixedPrice": 100.0,  
    "ReservedElasticsearchInstanceOfferingId":  
    "1a2a3a4a5-1a2a-3a4a-5a6a-1a2a3a4a5a6a",  
    "ReservationName": "my-reservation",  
    "PaymentOption": "PARTIAL_UPFRONT",  
    "UsagePrice": 0.0,  
    "ReservedElasticsearchInstanceId": "9a8a7a6a-5a4a-3a2a-1a0a-9a8a7a6a5a4a",  
    "RecurringCharges": [{  
      "RecurringChargeAmount": 0.603,  
      "RecurringChargeFrequency": "Hourly"  
    }],  
    "State": "payment-pending",  
    "StartTime": 1522872571.229,  
  }],  
}
```

```
    "ElasticsearchInstanceCount": 3,  
    "Duration": 31536000,  
    "ElasticsearchInstanceType": "m4.2xlarge.elasticsearch",  
    "CurrencyCode": "USD"  
  }  
}
```

For more information, see [Reserved Instances](#) in the *Amazon Elasticsearch Service Developer Guide*.

- For API details, see [DescribeReservedElasticsearchInstances](#) in *AWS CLI Command Reference*.

list-domain-names

The following code example shows how to use `list-domain-names`.

AWS CLI

To list all domains

The following `list-domain-names` example provides a quick summary of all domains in the region.

```
aws es list-domain-names
```

Output:

```
{  
  "DomainNames": [{  
    "DomainName": "cli-example-1"  
  },  
  {  
    "DomainName": "cli-example-2"  
  }  
]  
}
```

For more information, see [Creating and Managing Amazon Elasticsearch Service Domains](#) in the *Amazon Elasticsearch Service Developer Guide*.

- For API details, see [ListDomainNames](#) in *AWS CLI Command Reference*.

AWS OpsWorks examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS OpsWorks.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

assign-instance

The following code example shows how to use `assign-instance`.

AWS CLI

To assign a registered instance to a layer

The following example assigns a registered instance to a custom layer.

```
aws opsworks --region us-east-1 assign-instance --instance-id 4d6d1710-ded9-42a1-b08e-b043ad7af1e2 --layer-ids 26cf1d32-6876-42fa-bbf1-9cad0b0b938
```

Output: None.

More Information

For more information, see *Assigning a Registered Instance to a Layer* in the *AWS OpsWorks User Guide*.

- For API details, see [AssignInstance](#) in *AWS CLI Command Reference*.

assign-volume

The following code example shows how to use `assign-volume`.

AWS CLI

To assign a registered volume to an instance

The following example assigns a registered Amazon Elastic Block Store (Amazon EBS) volume to an instance. The volume is identified by its volume ID, which is the GUID that AWS OpsWorks assigns when you register the volume with a stack, not the Amazon Elastic Compute Cloud (Amazon EC2) volume ID. Before you run `assign-volume`, you must first run `update-volume` to assign a mount point to the volume.

```
aws opsworks --region us-east-1 assign-volume --instance-id 4d6d1710-ded9-42a1-b08e-b043ad7af1e2 --volume-id 26cf1d32-6876-42fa-bbf1-9cad0bff938
```

Output: None.

More Information

For more information, see *Assigning Amazon EBS Volumes to an Instance* in the *AWS OpsWorks User Guide*.

- For API details, see [AssignVolume](#) in *AWS CLI Command Reference*.

associate-elastic-ip

The following code example shows how to use `associate-elastic-ip`.

AWS CLI

To associate an Elastic IP address with an instance

The following example associates an Elastic IP address with a specified instance.

```
aws opsworks --region us-east-1 associate-elastic-ip --instance-id dfe18b02-5327-493d-91a4-c5c0c448927f --elastic-ip 54.148.130.96
```

Output: None.

More Information

For more information, see Resource Management in the *AWS OpsWorks User Guide*.

- For API details, see [AssociateElasticIp](#) in *AWS CLI Command Reference*.

attach-elastic-load-balancer

The following code example shows how to use `attach-elastic-load-balancer`.

AWS CLI

To attach a load balancer to a layer

The following example attaches a load balancer, identified by its name, to a specified layer.

```
aws opsworks --region us-east-1 attach-elastic-load-balancer --elastic-load-balancer-name Java-LB --layer-id 888c5645-09a5-4d0e-95a8-812ef1db76a4
```

Output: None.

More Information

For more information, see Elastic Load Balancing in the *AWS OpsWorks User Guide*.

- For API details, see [AttachElasticLoadBalancer](#) in *AWS CLI Command Reference*.

create-app

The following code example shows how to use `create-app`.

AWS CLI

Example 1: To create an app

The following example creates a PHP app named SimplePHPApp from code stored in a GitHub repository. The command uses the shorthand form of the application source definition.

```
aws opsworks create-app \  
  --region us-east-1 \  
  --source-definition SimplePHPApp
```



```
--stack-id f6673d70-32e6-4425-8999-265dd002fec7 \  
--name SimplePHPApp \  
--type php \  
--app-source Type=git,Url=git://github.com/amazonwebservices/opsworks-demo-php-  
simple-app.git,Revision=version1
```

Output:

```
{  
  "AppId": "6cf5163c-a951-444f-a8f7-3716be75f2a2"  
}
```

Example 2: To create an app with an attached database

The following example creates a JSP app from code stored in .zip archive in a public S3 bucket. It attaches an RDS DB instance to serve as the app's data store. The application and database sources are defined in separate JSON files that are in the directory from which you run the command.

```
aws opsworks create-app \  
  --region us-east-1 \  
  --stack-id 8c428b08-a1a1-46ce-a5f8-feddc43771b8 \  
  --name SimpleJSP \  
  --type java \  
  --app-source file://appsource.json \  
  --data-sources file://datasource.json
```

The application source information is in `appsource.json` and contains the following.

```
{  
  "Type": "archive",  
  "Url": "https://s3.amazonaws.com/opsworks-demo-assets/simplejsp.zip"  
}
```

The database source information is in `datasource.json` and contains the following.

```
[  
  {  
    "Type": "RdsDbInstance",  
    "Arn": "arn:aws:rds:us-west-2:123456789012:db:clitestdb",
```

```

    "DatabaseName": "mydb"
  }
]

```

Note: For an RDS DB instance, you must first use `register-rds-db-instance` to register the instance with the stack. For MySQL App Server instances, set `Type` to `OpsworksMySQLInstance`. These instances are created by AWS OpsWorks, so they do not have to be registered.

Output:

```

{
  "AppId": "26a61ead-d201-47e3-b55c-2a7c666942f8"
}

```

For more information, see *Adding Apps* in the *AWS OpsWorks User Guide*.

- For API details, see [CreateApp](#) in *AWS CLI Command Reference*.

create-deployment

The following code example shows how to use `create-deployment`.

AWS CLI

Example 1: To deploy apps and run stack commands

The following examples show how to use the `create-deployment` command to deploy apps and run stack commands. Notice that the quote (") characters in the JSON object that specifies the command are all preceded by escape characters (\). Without the escape characters, the command might return an invalid JSON error.

The following `create-deployment` example deploys an app to a specified stack.

```

aws opsworks create-deployment \
  --stack-id cfb7e082-ad1d-4599-8e81-de1c39ab45bf \
  --app-id 307be5c8-d55d-47b5-bd6e-7bd417c6c7eb
  --command "{\"Name\": \"deploy\"}"

```

Output:

```
{
  "DeploymentId": "5746c781-df7f-4c87-84a7-65a119880560"
}
```

Example 2: To deploy a Rails App and Migrate the Database

The following `create-deployment` command deploys a Ruby on Rails app to a specified stack and migrates the database.

```
aws opsworks create-deployment \
  --stack-id cfb7e082-ad1d-4599-8e81-de1c39ab45bf \
  --app-id 307be5c8-d55d-47b5-bd6e-7bd417c6c7eb \
  --command "{\"Name\":\"deploy\", \"Args\":{\"migrate\":[\"true\"]}]\""
```

Output:

```
{
  "DeploymentId": "5746c781-df7f-4c87-84a7-65a119880560"
}
```

For more information on deployment, see [Deploying Apps](#) in the *AWS OpsWorks User Guide*.

Example 3: Run a Recipe

The following `create-deployment` command runs a custom recipe, `phpapp::appsetup`, on the instances in a specified stack.

```
aws opsworks create-deployment \
  --stack-id 935450cc-61e0-4b03-a3e0-160ac817d2bb \
  --command "{\"Name\":\"execute_recipes\", \"Args\":{\"recipes\":[\"phpapp::appsetup\"]}]\""
```

Output:

```
{
  "DeploymentId": "5cbaa7b9-4e09-4e53-aa1b-314fbd106038"
}
```

For more information, see [Run Stack Commands](#) in the *AWS OpsWorks User Guide*.

Example 4: Install Dependencies

The following `create-deployment` command installs dependencies, such as packages or Ruby gems, on the instances in a specified stack.

```
aws opsworks create-deployment \  
  --stack-id 935450cc-61e0-4b03-a3e0-160ac817d2bb \  
  --command "{\"Name\":\"install_dependencies\"}"
```

Output:

```
{  
  "DeploymentId": "aef5b255-8604-4928-81b3-9b0187f962ff"  
}
```

For more information, see [Run Stack Commands](#) in the *AWS OpsWorks User Guide*.

- For API details, see [CreateDeployment](#) in *AWS CLI Command Reference*.

create-instance

The following code example shows how to use `create-instance`.

AWS CLI

To create an instance

The following `create-instance` command creates an `m1.large` Amazon Linux instance named `myinstance1` in a specified stack. The instance is assigned to one layer.

```
aws opsworks --region us-east-1 create-instance --stack-id 935450cc-61e0-4b03-  
a3e0-160ac817d2bb --layer-ids 5c8c272a-f2d5-42e3-8245-5bf3927cb65b --hostname  
myinstance1 --instance-type m1.large --os "Amazon Linux"
```

To use an autogenerated name, call `get-hostname-suggestion`, which generates a hostname based on the theme that you specified when you created the stack. Then pass that name to the `hostname` argument.

Output:

```
{
  "InstanceId": "5f9adeaa-c94c-42c6-aeeef-28a5376002cd"
}
```

More Information

For more information, see *Adding an Instance to a Layer* in the *AWS OpsWorks User Guide*.

- For API details, see [CreateInstance](#) in *AWS CLI Command Reference*.

create-layer

The following code example shows how to use `create-layer`.

AWS CLI

To create a layer

The following `create-layer` command creates a PHP App Server layer named `MyPHPLayer` in a specified stack.

```
aws opsworks create-layer --region us-east-1 --stack-id
f6673d70-32e6-4425-8999-265dd002fec7 --type php-app --name MyPHPLayer --shortname
myphplayer
```

Output:

```
{
  "LayerId": "0b212672-6b4b-40e4-8a34-5a943cf2e07a"
}
```

More Information

For more information, see *How to Create a Layer* in the *AWS OpsWorks User Guide*.

- For API details, see [CreateLayer](#) in *AWS CLI Command Reference*.

create-server

The following code example shows how to use `create-server`.

AWS CLI

To create a server

The following `create-server` example creates a new Chef Automate server named `automate-06` in your default region. Note that defaults are used for most other settings, such as number of backups to retain, and maintenance and backup start times. Before you run a `create-server` command, complete prerequisites in [Getting Started with AWS OpsWorks for Chef Automate](#) in the *AWS Opsworks for Chef Automate User Guide*.

```
aws opsworks-cm create-server \  
  --engine "ChefAutomate" \  
  --instance-profile-arn "arn:aws:iam::012345678901:instance-profile/aws-opsworks-  
cm-ec2-role" \  
  --instance-type "t2.medium" \  
  --server-name "automate-06" \  
  --service-role-arn "arn:aws:iam::012345678901:role/aws-opsworks-cm-service-role"
```

Output:

```
{  
  "Server": {  
    "AssociatePublicIpAddress": true,  
    "BackupRetentionCount": 10,  
    "CreatedAt": 2019-12-29T13:38:47.520Z,  
    "DisableAutomatedBackup": FALSE,  
    "Endpoint": "https://opsworks-cm.us-east-1.amazonaws.com",  
    "Engine": "ChefAutomate",  
    "EngineAttributes": [  
      {  
        "Name": "CHEF_AUTOMATE_ADMIN_PASSWORD",  
        "Value": "1Example1"  
      }  
    ],  
    "EngineModel": "Single",  
    "EngineVersion": "2019-08",  
    "InstanceProfileArn": "arn:aws:iam::012345678901:instance-profile/aws-  
opsworks-cm-ec2-role",  
    "InstanceType": "t2.medium",  
    "PreferredBackupWindow": "Sun:02:00",  
    "PreferredMaintenanceWindow": "00:00",  
    "SecurityGroupIds": [ "sg-12345678" ],
```

```
"ServerArn": "arn:aws:iam::012345678901:instance/automate-06-1010V4UU2WRM2",
"ServerName": "automate-06",
"ServiceRoleArn": "arn:aws:iam::012345678901:role/aws-opsworks-cm-service-
role",
"Status": "CREATING",
"SubnetIds": [ "subnet-12345678" ]
}
}
```

For more information, see [CreateServer](#) in the *AWS OpsWorks for Chef Automate API Reference*.

- For API details, see [CreateServer](#) in *AWS CLI Command Reference*.

create-stack

The following code example shows how to use `create-stack`.

AWS CLI

To create a stack

The following `create-stack` command creates a stack named CLI Stack.

```
aws opsworks create-stack --name "CLI Stack" --stack-region "us-east-1" --service-
role-arn arn:aws:iam::123456789012:role/aws-opsworks-service-role --default-
instance-profile-arn arn:aws:iam::123456789012:instance-profile/aws-opsworks-ec2-
role --region us-east-1
```

The `service-role-arn` and `default-instance-profile-arn` parameters are required. You typically use the ones that AWS OpsWorks creates for you when you create your first stack. To get the Amazon Resource Names (ARNs) for your account, go to the IAM console, choose Roles in the navigation panel, choose the role or profile, and choose the Summary tab.

Output:

```
{
  "StackId": "f6673d70-32e6-4425-8999-265dd002fec7"
}
```

More Information

For more information, see *Create a New Stack* in the *AWS OpsWorks User Guide*.

- For API details, see [CreateStack](#) in *AWS CLI Command Reference*.

create-user-profile

The following code example shows how to use `create-user-profile`.

AWS CLI

To create a user profile

You import an AWS Identity and Access Manager (IAM) user into AWS OpsWorks by calling `create-user-profile` to create a user profile. The following example creates a user profile for the `cli-user-test` IAM user, who is identified by Amazon Resource Name (ARN). The example assigns the user an SSH username of `myusername` and enables self management, which allows the user to specify an SSH public key.

```
aws opsworks --region us-east-1 create-user-profile --iam-user-arn
arn:aws:iam::123456789102:user/cli-user-test --ssh-username myusername --allow-
self-management
```

Output:

```
{
  "IamUserArn": "arn:aws:iam::123456789102:user/cli-user-test"
}
```

Tip: This command imports an IAM user into AWS OpsWorks, but only with the permissions that are granted by the attached policies. You can grant per-stack AWS OpsWorks permissions by using the `set-permissions` command.

More Information

For more information, see *Importing Users into AWS OpsWorks* in the *AWS OpsWorks User Guide*.

- For API details, see [CreateUserProfile](#) in *AWS CLI Command Reference*.

delete-app

The following code example shows how to use `delete-app`.

AWS CLI

To delete an app

The following example deletes a specified app, which is identified by its app ID. You can obtain an app ID by going to the app's details page on the AWS OpsWorks console or by running the `describe-apps` command.

```
aws opsworks delete-app --region us-east-1 --app-id 577943b9-2ec1-4baf-
a7bf-1d347601edc5
```

Output: None.

More Information

For more information, see *Apps* in the *AWS OpsWorks User Guide*.

- For API details, see [DeleteApp](#) in *AWS CLI Command Reference*.

delete-instance

The following code example shows how to use `delete-instance`.

AWS CLI

To delete an instance

The following `delete-instance` example deletes a specified instance, which is identified by its instance ID. You can find an instance ID by opening the instance's details page in the AWS OpsWorks console, or by running the `describe-instances` command.

If the instance is online, you must first stop the instance by calling `stop-instance`, and then you must wait until the instance has stopped. Run `describe-instances` to check the instance status.

To remove the instance's Amazon EBS volumes or Elastic IP addresses, add the `--delete-volumes` or `--delete-elastic-ip` arguments, respectively.

```
aws opsworks delete-instance \  
  --region us-east-1 \  
  --instance-id i-12345678
```

```
--instance-id 3a21cfac-4a1f-4ce2-a921-b2cfba6f7771
```

This command produces no output.

For more information, see [Deleting AWS OpsWorks Instances](#) in the *AWS OpsWorks User Guide*.

- For API details, see [DeleteInstance](#) in *AWS CLI Command Reference*.

delete-layer

The following code example shows how to use `delete-layer`.

AWS CLI

To delete a layer

The following example deletes a specified layer, which is identified by its layer ID. You can obtain a layer ID by going to the layer's details page on the AWS OpsWorks console or by running the `describe-layers` command.

Note: Before deleting a layer, you must use `delete-instance` to delete all of the layer's instances.

```
aws opsworks delete-layer --region us-east-1 --layer-id a919454e-b816-4598-  
b29a-5796afb498ed
```

Output: None.

More Information

For more information, see [Deleting AWS OpsWorks Instances](#) in the *AWS OpsWorks User Guide*.

- For API details, see [DeleteLayer](#) in *AWS CLI Command Reference*.

delete-stack

The following code example shows how to use `delete-stack`.

AWS CLI

To delete a stack

The following example deletes a specified stack, which is identified by its stack ID. You can obtain a stack ID by clicking **Stack Settings** on the AWS OpsWorks console or by running the `describe-stacks` command.

Note: Before deleting a layer, you must use `delete-app`, `delete-instance`, and `delete-layer` to delete all of the stack's apps, instances, and layers.

```
aws opsworks delete-stack --region us-east-1 --stack-id
154a9d89-7e9e-433b-8de8-617e53756c84
```

Output: None.

More Information

For more information, see Shut Down a Stack in the *AWS OpsWorks User Guide*.

- For API details, see [DeleteStack](#) in *AWS CLI Command Reference*.

delete-user-profile

The following code example shows how to use `delete-user-profile`.

AWS CLI

To delete a user profile and remove an IAM user from AWS OpsWorks

The following example deletes the user profile for a specified AWS Identity and Access Management (IAM) user, who is identified by Amazon Resource Name (ARN). The operation removes the user from AWS OpsWorks, but does not delete the IAM user. You must use the IAM console, CLI, or API for that task.

```
aws opsworks --region us-east-1 delete-user-profile --iam-user-arn
arn:aws:iam::123456789102:user/cli-user-test
```

Output: None.

More Information

For more information, see Importing Users into AWS OpsWorks in the *AWS OpsWorks User Guide*.

- For API details, see [DeleteUserProfile](#) in *AWS CLI Command Reference*.

deregister-elastic-ip

The following code example shows how to use `deregister-elastic-ip`.

AWS CLI

To deregister an Elastic IP address from a stack

The following example deregisters an Elastic IP address, identified by its IP address, from its stack.

```
aws opsworks deregister-elastic-ip --region us-east-1 --elastic-ip 54.148.130.96
```

Output: None.

More Information

For more information, see Deregistering Elastic IP Addresses in the *AWS OpsWorks User Guide*.

- For API details, see [DeregisterElasticIp](#) in *AWS CLI Command Reference*.

deregister-instance

The following code example shows how to use `deregister-instance`.

AWS CLI

To deregister a registered instance from a stack

The following `deregister-instance` command deregisters a registered instance from its stack.

```
aws opsworks --region us-east-1 deregister-instance --instance-id 4d6d1710-ded9-42a1-b08e-b043ad7af1e2
```

Output: None.

More Information

For more information, see Deregistering a Registered Instance in the *AWS OpsWorks User Guide*.

- For API details, see [DeregisterInstance](#) in *AWS CLI Command Reference*.

deregister-rds-db-instance

The following code example shows how to use `deregister-rds-db-instance`.

AWS CLI

To deregister an Amazon RDS DB instance from a stack

The following example deregisters an RDS DB instance, identified by its ARN, from its stack.

```
aws opsworks deregister-rds-db-instance --region us-east-1 --rds-db-instance-arn
arn:aws:rds:us-west-2:123456789012:db:clitestdb
```

Output: None.

More Information

For more information, see *Deregistering Amazon RDS Instances* in the *ASW OpsWorks User Guide*.

instance ID: clitestdb Master usernames: cliuser Master PWD: some23!pwd DB Name: mydb
aws opsworks deregister-rds-db-instance --region us-east-1 --rds-db-instance-arn arn:aws:rds:us-west-2:645732743964:db:clitestdb

- For API details, see [DeregisterRdsDbInstance](#) in *AWS CLI Command Reference*.

deregister-volume

The following code example shows how to use `deregister-volume`.

AWS CLI

To deregister an Amazon EBS volume

The following example deregisters an EBS volume from its stack. The volume is identified by its volume ID, which is the GUID that AWS OpsWorks assigned when you registered the volume with the stack, not the EC2 volume ID.

```
aws opsworks deregister-volume --region us-east-1 --volume-id 5c48ef52-3144-4bf5-
beaa-fda4deb23d4d
```

Output: None.

More Information

For more information, see *Deregistering Amazon EBS Volumes* in the *AWS OpsWorks User Guide*.

- For API details, see [DeregisterVolume](#) in *AWS CLI Command Reference*.

describe-apps

The following code example shows how to use `describe-apps`.

AWS CLI

To describe apps

The following `describe-apps` command describes the apps in a specified stack.

```
aws opsworks describe-apps \  
  --stack-id 38ee91e2-abdc-4208-a107-0b7168b3cc7a \  
  --region us-east-1
```

Output:

```
{  
  "Apps": [  
    {  
      "StackId": "38ee91e2-abdc-4208-a107-0b7168b3cc7a",  
      "AppSource": {  
        "Url": "https://s3-us-west-2.amazonaws.com/opsworks-demo-assets/  
simplejsp.zip",  
        "Type": "archive"  
      },  
      "Name": "SimpleJSP",  
      "EnableSsl": false,  
      "SslConfiguration": {},  
      "AppId": "da1decc1-0dff-43ea-ad7c-bb667cd87c8b",  
      "Attributes": {  
        "RailsEnv": null,  
        "AutoBundleOnDeploy": "true",  
        "DocumentRoot": "ROOT"  
      },  
      "Shortname": "simplejsp",
```

```

        "Type": "other",
        "CreatedAt": "2013-08-01T21:46:54+00:00"
    }
]
}

```

For more information, see *Apps* in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeApps](#) in *AWS CLI Command Reference*.

describe-commands

The following code example shows how to use `describe-commands`.

AWS CLI

To describe commands

The following `describe-commands` command describes the commands in a specified instance.

```

aws opsworks describe-commands \
  --instance-id 8c2673b9-3fe5-420d-9cfa-78d875ee7687 \
  --region us-east-1

```

Output:

```

{
  "Commands": [
    {
      "Status": "successful",
      "CompletedAt": "2013-07-25T18:57:47+00:00",
      "InstanceId": "8c2673b9-3fe5-420d-9cfa-78d875ee7687",
      "DeploymentId": "6ed0df4c-9ef7-4812-8dac-d54a05be1029",
      "AcknowledgedAt": "2013-07-25T18:57:41+00:00",
      "LogUrl": "https://s3.amazonaws.com/<bucket-name>/logs/008c1a91-ec59-4d51-971d-3adff54b00cc?AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&Expires=1375394373&Signature=HkXil6UuNfxTCC37EPQAa462E1E%3D&response-cache-control=private&response-content-encoding=gzip&response-content-type=text%2Fplain",
      "Type": "undeploy",
      "CommandId": "008c1a91-ec59-4d51-971d-3adff54b00cc",
      "CreatedAt": "2013-07-25T18:57:34+00:00",
      "ExitCode": 0
    },
  ],
}

```

```

    {
      "Status": "successful",
      "CompletedAt": "2013-07-25T18:55:40+00:00",
      "InstanceId": "8c2673b9-3fe5-420d-9cfa-78d875ee7687",
      "DeploymentId": "19d3121e-d949-4ff2-9f9d-94eac087862a",
      "AcknowledgedAt": "2013-07-25T18:55:32+00:00",
      "LogUrl": "https://s3.amazonaws.com/<bucket-name>/
logs/899d3d64-0384-47b6-a586-33433aad117c?AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE
&Expires=1375394373&Signature=xMsJvtLuUqWmsr8s%2FAjVru0BtRs%3D&response-cache-
control=private&response-content-encoding=gzip&response-content-type=text%2Fplain",
      "Type": "deploy",
      "CommandId": "899d3d64-0384-47b6-a586-33433aad117c",
      "CreatedAt": "2013-07-25T18:55:29+00:00",
      "ExitCode": 0
    }
  ]
}

```

For more information, see AWS OpsWorks Lifecycle Events in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeCommands](#) in *AWS CLI Command Reference*.

describe-deployments

The following code example shows how to use describe-deployments.

AWS CLI

To describe deployments

The following describe-deployments command describes the deployments in a specified stack.

```
aws opsworks --region us-east-1 describe-deployments --stack-id 38ee91e2-abdc-4208-
a107-0b7168b3cc7a
```

Output:

```

{
  "Deployments": [
    {
      "StackId": "38ee91e2-abdc-4208-a107-0b7168b3cc7a",

```



```

    "Status": "successful",
    "CompletedAt": "2013-07-25T18:57:49+00:00",
    "DeploymentId": "6ed0df4c-9ef7-4812-8dac-d54a05be1029",
    "Command": {
      "Args": {},
      "Name": "undeploy"
    },
    "CreatedAt": "2013-07-25T18:57:34+00:00",
    "Duration": 15,
    "InstanceIds": [
      "8c2673b9-3fe5-420d-9cfa-78d875ee7687",
      "9e588a25-35b2-4804-bd43-488f85ebe5b7"
    ]
  },
  {
    "StackId": "38ee91e2-abdc-4208-a107-0b7168b3cc7a",
    "Status": "successful",
    "CompletedAt": "2013-07-25T18:56:41+00:00",
    "IamUserArn": "arn:aws:iam::123456789012:user/someuser",
    "DeploymentId": "19d3121e-d949-4ff2-9f9d-94eac087862a",
    "Command": {
      "Args": {},
      "Name": "deploy"
    },
    "InstanceIds": [
      "8c2673b9-3fe5-420d-9cfa-78d875ee7687",
      "9e588a25-35b2-4804-bd43-488f85ebe5b7"
    ],
    "Duration": 72,
    "CreatedAt": "2013-07-25T18:55:29+00:00"
  }
]
}

```

More Information

For more information, see *Deploying Apps* in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeDeployments](#) in *AWS CLI Command Reference*.

describe-elastic-ips

The following code example shows how to use `describe-elastic-ips`.

AWS CLI

To describe Elastic IP instances

The following `describe-elastic-ips` command describes the Elastic IP addresses in a specified instance.

```
aws opsworks --region us-east-1 describe-elastic-ips --instance-id b62f3e04-
e9eb-436c-a91f-d9e9a396b7b0
```

Output:

```
{
  "ElasticIps": [
    {
      "Ip": "192.0.2.0",
      "Domain": "standard",
      "Region": "us-west-2"
    }
  ]
}
```

More Information

For more information, see Instances in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeElasticIps](#) in *AWS CLI Command Reference*.

describe-elastic-load-balancers

The following code example shows how to use `describe-elastic-load-balancers`.

AWS CLI

To describe a stack's elastic load balancers

The following `describe-elastic-load-balancers` command describes a specified stack's load balancers.

```
aws opsworks --region us-west-2 describe-elastic-load-balancers --stack-id
6f4660e5-37a6-4e42-bfa0-1358ebd9c182
```

Output: This particular stack has one load balancer.

```
{
  "ElasticLoadBalancers": [
    {
      "SubnetIds": [
        "subnet-60e4ea04",
        "subnet-66e1c110"
      ],
      "Ec2InstanceIds": [],
      "ElasticLoadBalancerName": "my-balancer",
      "Region": "us-west-2",
      "LayerId": "344973cb-bf2b-4cd0-8d93-51cd819bab04",
      "AvailabilityZones": [
        "us-west-2a",
        "us-west-2b"
      ],
      "VpcId": "vpc-b319f9d4",
      "StackId": "6f4660e5-37a6-4e42-bfa0-1358ebd9c182",
      "DnsName": "my-balancer-2094040179.us-west-2.elb.amazonaws.com"
    }
  ]
}
```

More Information

For more information, see Apps in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeElasticLoadBalancers](#) in *AWS CLI Command Reference*.

describe-instances

The following code example shows how to use `describe-instances`.

AWS CLI

To describe instances

The following `describe-instances` command describes the instances in a specified stack:

```
aws opsworks --region us-east-1 describe-instances --stack-id 8c428b08-a1a1-46ce-
a5f8-feddc43771b8
```

Output: The following output example is for a stack with two instances. The first is a registered EC2 instance, and the second was created by AWS OpsWorks.

```
{
  "Instances": [
    {
      "StackId": "71c7ca72-55ae-4b6a-8ee1-a8dcdded3fa0f",
      "PrivateDns": "ip-10-31-39-66.us-west-2.compute.internal",
      "LayerIds": [
        "26cf1d32-6876-42fa-bbf1-9cadc0bff938"
      ],
      "EbsOptimized": false,
      "ReportedOs": {
        "Version": "14.04",
        "Name": "ubuntu",
        "Family": "debian"
      },
      "Status": "online",
      "InstanceId": "4d6d1710-ded9-42a1-b08e-b043ad7af1e2",
      "SshKeyName": "US-West-2",
      "InfrastructureClass": "ec2",
      "RootDeviceVolumeId": "vol-d08ec6c1",
      "SubnetId": "subnet-b8de0ddd",
      "InstanceType": "t1.micro",
      "CreatedAt": "2015-02-24T20:52:49+00:00",
      "AmiId": "ami-35501205",
      "Hostname": "ip-192-0-2-0",
      "Ec2InstanceId": "i-5cd23551",
      "PublicDns": "ec2-192-0-2-0.us-west-2.compute.amazonaws.com",
      "SecurityGroupIds": [
        "sg-c4d3f0a1"
      ],
      "Architecture": "x86_64",
      "RootDeviceType": "ebs",
      "InstallUpdatesOnBoot": true,
      "Os": "Custom",
      "VirtualizationType": "paravirtual",
      "AvailabilityZone": "us-west-2a",
      "PrivateIp": "10.31.39.66",
      "PublicIp": "192.0.2.06",
      "RegisteredBy": "arn:aws:iam::123456789102:user/AWS/OpsWorks/OpsWorks-
      EC2Register-i-5cd23551"
    },
  ],
}
```

```

{
  "StackId": "71c7ca72-55ae-4b6a-8ee1-a8dcdded3fa0f",
  "PrivateDns": "ip-10-31-39-158.us-west-2.compute.internal",
  "SshHostRsaKeyFingerprint": "69:6b:7b:8b:72:f3:ed:23:01:00:05:bc:9f:a4:60:c1",
  "LayerIds": [
    "26cf1d32-6876-42fa-bbf1-9cad0bfff938"
  ],
  "EbsOptimized": false,
  "ReportedOs": {},
  "Status": "booting",
  "InstanceId": "9b137a0d-2f5d-4cc0-9704-13da4b31fdcb",
  "SshKeyName": "US-West-2",
  "InfrastructureClass": "ec2",
  "RootDeviceVolumeId": "vol-e09dd5f1",
  "SubnetId": "subnet-b8de0ddd",
  "InstanceProfileArn": "arn:aws:iam::123456789102:instance-profile/aws-opsworks-ec2-role",
  "InstanceType": "c3.large",
  "CreatedAt": "2015-02-24T21:29:33+00:00",
  "AmiId": "ami-9fc29baf",
  "SshHostDsaKeyFingerprint": "fc:87:95:c3:f5:e1:3b:9f:d2:06:6e:62:9a:35:27:e8",
  "Ec2InstanceId": "i-8d2dca80",
  "PublicDns": "ec2-192-0-2-1.us-west-2.compute.amazonaws.com",
  "SecurityGroupIds": [
    "sg-b022add5",
    "sg-b122add4"
  ],
  "Architecture": "x86_64",
  "RootDeviceType": "ebs",
  "InstallUpdatesOnBoot": true,
  "Os": "Amazon Linux 2014.09",
  "VirtualizationType": "paravirtual",
  "AvailabilityZone": "us-west-2a",
  "Hostname": "custom11",
  "PrivateIp": "10.31.39.158",
  "PublicIp": "192.0.2.0"
}
]
}

```

More Information

For more information, see Instances in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeInstances](#) in *AWS CLI Command Reference*.

describe-layers

The following code example shows how to use describe-layers.

AWS CLI

To describe a stack's layers

The following describe-layers command describes the layers in a specified stack:

```
aws opsworks --region us-east-1 describe-layers --stack-id 38ee91e2-abdc-4208-
a107-0b7168b3cc7a
```

Output:

```
{
  "Layers": [
    {
      "StackId": "38ee91e2-abdc-4208-a107-0b7168b3cc7a",
      "Type": "db-master",
      "DefaultSecurityGroupNames": [
        "AWS-OpsWorks-DB-Master-Server"
      ],
      "Name": "MySQL",
      "Packages": [],
      "DefaultRecipes": {
        "Undeploy": [],
        "Setup": [
          "opsworks_initial_setup",
          "ssh_host_keys",
          "ssh_users",
          "mysql::client",
          "dependencies",
          "ebs",
          "opsworks_ganglia::client",
          "mysql::server",
          "dependencies",
          "deploy:mysql"
        ],
        "Configure": [
          "opsworks_ganglia::configure-client",
```

```
        "ssh_users",
        "agent_version",
        "deploy:mysql"
    ],
    "Shutdown": [
        "opsworks_shutdown::default",
        "mysql::stop"
    ],
    "Deploy": [
        "deploy::default",
        "deploy:mysql"
    ]
},
"CustomRecipes": {
    "Undeploy": [],
    "Setup": [],
    "Configure": [],
    "Shutdown": [],
    "Deploy": []
},
"EnableAutoHealing": false,
"LayerId": "41a20847-d594-4325-8447-171821916b73",
"Attributes": {
    "MysqlRootPasswordUbiquitous": "true",
    "RubygemsVersion": null,
    "RailsStack": null,
    "HaproxyHealthCheckMethod": null,
    "RubyVersion": null,
    "BundlerVersion": null,
    "HaproxyStatsPassword": null,
    "PassengerVersion": null,
    "MemcachedMemory": null,
    "EnableHaproxyStats": null,
    "ManageBundler": null,
    "NodejsVersion": null,
    "HaproxyHealthCheckUrl": null,
    "MysqlRootPassword": "*****FILTERED*****",
    "GangliaPassword": null,
    "GangliaUser": null,
    "HaproxyStatsUrl": null,
    "GangliaUrl": null,
    "HaproxyStatsUser": null
},
"Shortname": "db-master",
```

```
"AutoAssignElasticIps": false,
"CustomSecurityGroupIds": [],
"CreatedAt": "2013-07-25T18:11:19+00:00",
"VolumeConfigurations": [
  {
    "MountPoint": "/vol/mysql",
    "Size": 10,
    "NumberOfDisks": 1
  }
],
},
{
  "StackId": "38ee91e2-abdc-4208-a107-0b7168b3cc7a",
  "Type": "custom",
  "DefaultSecurityGroupNames": [
    "AWS-OpsWorks-Custom-Server"
  ],
  "Name": "TomCustom",
  "Packages": [],
  "DefaultRecipes": {
    "Undeploy": [],
    "Setup": [
      "opsworks_initial_setup",
      "ssh_host_keys",
      "ssh_users",
      "mysql::client",
      "dependencies",
      "ebs",
      "opsworks_ganglia::client"
    ],
    "Configure": [
      "opsworks_ganglia::configure-client",
      "ssh_users",
      "agent_version"
    ],
    "Shutdown": [
      "opsworks_shutdown::default"
    ],
    "Deploy": [
      "deploy::default"
    ]
  },
  "CustomRecipes": {
    "Undeploy": [],
```



```
    "Setup": [
      "tomcat::setup"
    ],
    "Configure": [
      "tomcat::configure"
    ],
    "Shutdown": [],
    "Deploy": [
      "tomcat::deploy"
    ]
  },
  "EnableAutoHealing": true,
  "LayerId": "e6cbcd29-d223-40fc-8243-2eb213377440",
  "Attributes": {
    "MysqlRootPasswordUbiquitous": null,
    "RubygemsVersion": null,
    "RailsStack": null,
    "HaproxyHealthCheckMethod": null,
    "RubyVersion": null,
    "BundlerVersion": null,
    "HaproxyStatsPassword": null,
    "PassengerVersion": null,
    "MemcachedMemory": null,
    "EnableHaproxyStats": null,
    "ManageBundler": null,
    "NodejsVersion": null,
    "HaproxyHealthCheckUrl": null,
    "MysqlRootPassword": null,
    "GangliaPassword": null,
    "GangliaUser": null,
    "HaproxyStatsUrl": null,
    "GangliaUrl": null,
    "HaproxyStatsUser": null
  },
  "Shortname": "tomcustom",
  "AutoAssignElasticIps": false,
  "CustomSecurityGroupIds": [],
  "CreatedAt": "2013-07-25T18:12:53+00:00",
  "VolumeConfigurations": []
}
]
```

More Information

For more information, see Layers in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeLayers](#) in *AWS CLI Command Reference*.

describe-load-based-auto-scaling

The following code example shows how to use `describe-load-based-auto-scaling`.

AWS CLI

To describe a layer's load-based scaling configuration

The following example describes a specified layer's load-based scaling configuration. The layer is identified by its layer ID, which you can find on the layer's details page or by running `describe-layers`.

```
aws opsworks describe-load-based-auto-scaling --region us-east-1 --layer-ids
6bec29c9-c866-41a0-aba5-fa3e374ce2a1
```

Output: The example layer has a single load-based instance.

```
{
  "LoadBasedAutoScalingConfigurations": [
    {
      "DownScaling": {
        "IgnoreMetricsTime": 10,
        "ThresholdsWaitTime": 10,
        "InstanceCount": 1,
        "CpuThreshold": 30.0
      },
      "Enable": true,
      "UpScaling": {
        "IgnoreMetricsTime": 5,
        "ThresholdsWaitTime": 5,
        "InstanceCount": 1,
        "CpuThreshold": 80.0
      },
      "LayerId": "6bec29c9-c866-41a0-aba5-fa3e374ce2a1"
    }
  ]
}
```

More Information

For more information, see *How Automatic Load-based Scaling Works* in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeLoadBasedAutoScaling](#) in *AWS CLI Command Reference*.

describe-my-user-profile

The following code example shows how to use `describe-my-user-profile`.

AWS CLI

To obtain a user's profile

The following example shows how to obtain the profile of the AWS Identity and Access Management (IAM) user that is running the command.

```
aws opsworks --region us-east-1 describe-my-user-profile
```

Output: For brevity, most of the user's SSH public key is replaced by an ellipsis (...).

```
{
  "UserProfile": {
    "IamUserArn": "arn:aws:iam::123456789012:user/myusername",
    "SshPublicKey": "ssh-rsa AAAAB3NzaC1yc2EAAAABJQ...3LQ4aX9jpxQw== rsa-
key-20141104",
    "Name": "myusername",
    "SshUsername": "myusername"
  }
}
```

More Information

For more information, see *Importing Users into AWS OpsWorks* in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeMyUserProfile](#) in *AWS CLI Command Reference*.

describe-permissions

The following code example shows how to use `describe-permissions`.

AWS CLI

To obtain a user's per-stack AWS OpsWorks permission level

The following example shows how to obtain an AWS Identity and Access Management (IAM) user's permission level on a specified stack.

```
aws opsworks --region us-east-1 describe-permissions --iam-user-arn
arn:aws:iam::123456789012:user/cli-user-test --stack-id d72553d4-8727-448c-9b00-
f024f0ba1b06
```

Output:

```
{
  "Permissions": [
    {
      "StackId": "d72553d4-8727-448c-9b00-f024f0ba1b06",
      "IamUserArn": "arn:aws:iam::123456789012:user/cli-user-test",
      "Level": "manage",
      "AllowSudo": true,
      "AllowSsh": true
    }
  ]
}
```

More Information

For more information, see *Granting Per-Stack Permissions Levels* in the *AWS OpsWorks User Guide*.

- For API details, see [DescribePermissions](#) in *AWS CLI Command Reference*.

describe-raid-arrays

The following code example shows how to use `describe-raid-arrays`.

AWS CLI

To describe RAID arrays

The following example describes the RAID arrays attached to the instances in a specified stack.

```
aws opsworks --region us-east-1 describe-raid-arrays --stack-id
d72553d4-8727-448c-9b00-f024f0ba1b06
```

Output: The following is the output for a stack with one RAID array.

```
{
  "RaidArrays": [
    {
      "StackId": "d72553d4-8727-448c-9b00-f024f0ba1b06",
      "AvailabilityZone": "us-west-2a",
      "Name": "Created for php-app1",
      "NumberOfDisks": 2,
      "InstanceId": "9f14adbc-ced5-43b6-bf01-e7d0db6cf2f7",
      "RaidLevel": 0,
      "VolumeType": "standard",
      "RaidArrayId": "f2d4e470-5972-4676-b1b8-bae41ec3e51c",
      "Device": "/dev/md0",
      "MountPoint": "/mnt/workspace",
      "CreatedAt": "2015-02-26T23:53:09+00:00",
      "Size": 100
    }
  ]
}
```

For more information, see EBS Volumes in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeRaidArrays](#) in *AWS CLI Command Reference*.

describe-rds-db-instances

The following code example shows how to use `describe-rds-db-instances`.

AWS CLI

To describe a stack's registered Amazon RDS instances

The following example describes the Amazon RDS instances registered with a specified stack.

```
aws opsworks --region us-east-1 describe-rds-db-instances --stack-id
d72553d4-8727-448c-9b00-f024f0ba1b06
```

Output: The following is the output for a stack with one registered RDS instance.

```
{
  "RdsDbInstances": [
    {
      "Engine": "mysql",
      "StackId": "d72553d4-8727-448c-9b00-f024f0ba1b06",
      "MissingOnRds": false,
      "Region": "us-west-2",
      "RdsDbInstanceArn": "arn:aws:rds:us-west-2:123456789012:db:clitestdb",
      "DbPassword": "*****FILTERED*****",
      "Address": "clitestdb.cd1qlk5uwd0k.us-west-2.rds.amazonaws.com",
      "DbUser": "cliuser",
      "DbInstanceIdentifier": "clitestdb"
    }
  ]
}
```

For more information, see Resource Management in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeRdsDbInstances](#) in *AWS CLI Command Reference*.

describe-stack-provisioning-parameters

The following code example shows how to use `describe-stack-provisioning-parameters`.

AWS CLI

To return the provisioning parameters for a stack

The following `describe-stack-provisioning-parameters` example returns the provisioning parameters for a specified stack. Provisioning parameters include settings such as the agent installation location and public key that OpsWorks uses to manage the agent on instances in a stack.

```
aws opsworks describe-stack-provisioning-parameters \
  --stack-id 62744d97-6faf-4ecb-969b-a086fEXAMPLE
```

Output:

```
{
  "AgentInstallerUrl": "https://opsworks-instance-agent-us-
west-2.s3.amazonaws.com/ID_number/opsworks-agent-installer.tgz",
```

```
"Parameters": {
  "agent_installer_base_url": "https://opsworks-instance-agent-us-
west-2.s3.amazonaws.com",
  "agent_installer_tgz": "opsworks-agent-installer.tgz",
  "assets_download_bucket": "opsworks-instance-assets-us-
west-2.s3.amazonaws.com",
  "charlie_public_key": "-----BEGIN PUBLIC KEY-----PUBLIC_KEY_EXAMPLE\n-----
END PUBLIC KEY-----",
  "instance_service_endpoint": "opsworks-instance-service.us-
west-2.amazonaws.com",
  "instance_service_port": "443",
  "instance_service_region": "us-west-2",
  "instance_service_ssl_verify_peer": "true",
  "instance_service_use_ssl": "true",
  "ops_works_endpoint": "opsworks.us-west-2.amazonaws.com",
  "ops_works_port": "443",
  "ops_works_region": "us-west-2",
  "ops_works_ssl_verify_peer": "true",
  "ops_works_use_ssl": "true",
  "verbose": "false",
  "wait_between_runs": "30"
}
```

For more information, see [Run Stack Commands](#) in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeStackProvisioningParameters](#) in *AWS CLI Command Reference*.

describe-stack-summary

The following code example shows how to use `describe-stack-summary`.

AWS CLI

To describe a stack's configuration

The following `describe-stack-summary` command returns a summary of the specified stack's configuration.

```
aws opsworks --region us-east-1 describe-stack-summary --stack-id 8c428b08-
a1a1-46ce-a5f8-feddc43771b8
```

Output:

```
{
  "StackSummary": {
    "StackId": "8c428b08-a1a1-46ce-a5f8-feddc43771b8",
    "InstancesCount": {
      "Booting": 1
    },
    "Name": "CLITest",
    "AppsCount": 1,
    "LayersCount": 1,
    "Arn": "arn:aws:opsworks:us-west-2:123456789012:stack/8c428b08-a1a1-46ce-a5f8-feddc43771b8/"
  }
}
```

More Information

For more information, see *Stacks* in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeStackSummary](#) in *AWS CLI Command Reference*.

describe-stacks

The following code example shows how to use `describe-stacks`.

AWS CLI

To describe stacks

The following `describe-stacks` command describes an account's stacks.

```
aws opsworks --region us-east-1 describe-stacks
```

Output:

```
{
  "Stacks": [
    {
      "ServiceRoleArn": "arn:aws:iam::444455556666:role/aws-opsworks-service-role",
      "StackId": "aeb7523e-7c8b-49d4-b866-03aae9d4fbcf",
      "DefaultRootDeviceType": "instance-store",
      "Name": "TomStack-sd",
      "ConfigurationManager": {
        "Version": "11.4",

```



```

    "Name": "Chef"
  },
  "UseCustomCookbooks": true,
  "CustomJson": "{\n  \"tomcat\": {\n    \"base_version\": 7,\n    \"java_opts\n\": \"-Djava.awt.headless=true -Xmx256m\"\n  },\n  \"datasources\": {\n    \"ROOT\":\n  \"jdbc/mydb\"\n  }\n}",
  "Region": "us-east-1",
  "DefaultInstanceProfileArn": "arn:aws:iam::444455556666:instance-profile/aws-opsworks-ec2-role",
  "CustomCookbooksSource": {
    "Url": "git://github.com/example-repo/tomcustom.git",
    "Type": "git"
  },
  "DefaultAvailabilityZone": "us-east-1a",
  "HostnameTheme": "Layer_Dependent",
  "Attributes": {
    "Color": "rgb(45, 114, 184)"
  },
  "DefaultOs": "Amazon Linux",
  "CreatedAt": "2013-08-01T22:53:42+00:00"
},
{
  "ServiceRoleArn": "arn:aws:iam::444455556666:role/aws-opsworks-service-role",
  "StackId": "40738975-da59-4c5b-9789-3e422f2cf099",
  "DefaultRootDeviceType": "instance-store",
  "Name": "MyStack",
  "ConfigurationManager": {
    "Version": "11.4",
    "Name": "Chef"
  },
  "UseCustomCookbooks": false,
  "Region": "us-east-1",
  "DefaultInstanceProfileArn": "arn:aws:iam::444455556666:instance-profile/aws-opsworks-ec2-role",
  "CustomCookbooksSource": {},
  "DefaultAvailabilityZone": "us-east-1a",
  "HostnameTheme": "Layer_Dependent",
  "Attributes": {
    "Color": "rgb(45, 114, 184)"
  },
  "DefaultOs": "Amazon Linux",
  "CreatedAt": "2013-10-25T19:24:30+00:00"
}
]

```

```
}
```

More Information

For more information, see *Stacks* in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeStacks](#) in *AWS CLI Command Reference*.

describe-timebased-auto-scaling

The following code example shows how to use `describe-timebased-auto-scaling`.

AWS CLI

To describe the time-based scaling configuration of an instance

The following example describes a specified instance's time-based scaling configuration. The instance is identified by its instance ID, which you can find on the instance's details page or by running `describe-instances`.

```
aws opsworks describe-time-based-auto-scaling --region us-east-1 --instance-ids
701f2ffe-5d8e-4187-b140-77b75f55de8d
```

Output: The example has a single time-based instance.

```
{
  "TimeBasedAutoScalingConfigurations": [
    {
      "InstanceId": "701f2ffe-5d8e-4187-b140-77b75f55de8d",
      "AutoScalingSchedule": {
        "Monday": {
          "11": "on",
          "10": "on",
          "13": "on",
          "12": "on"
        },
        "Tuesday": {
          "11": "on",
          "10": "on",
          "13": "on",
          "12": "on"
        }
      }
    }
  ]
}
```

```
    }
  }
}
]
```

More Information

For more information, see *How Automatic Time-based Scaling Works* in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeTimebasedAutoScaling](#) in *AWS CLI Command Reference*.

describe-user-profiles

The following code example shows how to use `describe-user-profiles`.

AWS CLI

To describe user profiles

The following `describe-user-profiles` command describes the account's user profiles.

```
aws opsworks --region us-east-1 describe-user-profiles
```

Output:

```
{
  "UserProfiles": [
    {
      "IamUserArn": "arn:aws:iam::123456789012:user/someuser",
      "SshPublicKey": "ssh-rsa AAAAB3NzaC1yc2EAAAABJQAAAQEak0uP7i80q3Cko...",
      "AllowSelfManagement": true,
      "Name": "someuser",
      "SshUsername": "someuser"
    },
    {
      "IamUserArn": "arn:aws:iam::123456789012:user/cli-user-test",
      "AllowSelfManagement": true,
      "Name": "cli-user-test",
      "SshUsername": "myusername"
    }
  ]
}
```

```
    }  
  ]  
}
```

More Information

For more information, see *Managing AWS OpsWorks Users* in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeUserProfiles](#) in *AWS CLI Command Reference*.

describe-volumes

The following code example shows how to use `describe-volumes`.

AWS CLI

To describe a stack's volumes

The following example describes a stack's EBS volumes.

```
aws opsworks --region us-east-1 describe-volumes --stack-id 8c428b08-a1a1-46ce-a5f8-  
feddc43771b8
```

Output:

```
{  
  "Volumes": [  
    {  
      "Status": "in-use",  
      "AvailabilityZone": "us-west-2a",  
      "Name": "CLITest",  
      "InstanceId": "dfe18b02-5327-493d-91a4-c5c0c448927f",  
      "VolumeType": "standard",  
      "VolumeId": "56b66fbd-e1a1-4aff-9227-70f77118d4c5",  
      "Device": "/dev/sdi",  
      "Ec2VolumeId": "vol-295c1638",  
      "MountPoint": "/mnt/myvolume",  
      "Size": 1  
    }  
  ]  
}
```

More Information

For more information, see Resource Management in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeVolumes](#) in *AWS CLI Command Reference*.

detach-elastic-load-balancer

The following code example shows how to use `detach-elastic-load-balancer`.

AWS CLI

To detach a load balancer from its layer

The following example detaches a load balancer, identified by its name, from its layer.

```
aws opsworks --region us-east-1 detach-elastic-load-balancer --elastic-load-balancer-name Java-LB --layer-id 888c5645-09a5-4d0e-95a8-812ef1db76a4
```

Output: None.

More Information

For more information, see Elastic Load Balancing in the *AWS OpsWorks User Guide*.

- For API details, see [DetachElasticLoadBalancer](#) in *AWS CLI Command Reference*.

disassociate-elastic-ip

The following code example shows how to use `disassociate-elastic-ip`.

AWS CLI

To disassociate an Elastic IP address from an instance

The following example disassociates an Elastic IP address from a specified instance.

```
aws opsworks --region us-east-1 disassociate-elastic-ip --elastic-ip 54.148.130.96
```

Output: None.

More Information

For more information, see Resource Management in the *AWS OpsWorks User Guide*.

- For API details, see [DisassociateElasticIp](#) in *AWS CLI Command Reference*.

get-hostname-suggestion

The following code example shows how to use `get-hostname-suggestion`.

AWS CLI

To get the next hostname for a layer

The following example gets the next generated hostname for a specified layer. The layer used for this example is a Java Application Server layer with one instance. The stack's hostname theme is the default, `Layer_Dependent`.

```
aws opsworks --region us-east-1 get-hostname-suggestion --layer-id
888c5645-09a5-4d0e-95a8-812ef1db76a4
```

Output:

```
{
  "Hostname": "java-app2",
  "LayerId": "888c5645-09a5-4d0e-95a8-812ef1db76a4"
}
```

More Information

For more information, see Create a New Stack in the *AWS OpsWorks User Guide*.

- For API details, see [GetHostnameSuggestion](#) in *AWS CLI Command Reference*.

reboot-instance

The following code example shows how to use `reboot-instance`.

AWS CLI

To reboot an instance

The following example reboots an instance.

```
aws opsworks --region us-east-1 reboot-instance --instance-id
dfe18b02-5327-493d-91a4-c5c0c448927f
```

Output: None.

More Information

For more information, see *Rebooting an Instance* in the *AWS OpsWorks User Guide*.

- For API details, see [RebootInstance](#) in *AWS CLI Command Reference*.

register-elastic-ip

The following code example shows how to use `register-elastic-ip`.

AWS CLI

To register an Elastic IP address with a stack

The following example registers an Elastic IP address, identified by its IP address, with a specified stack.

Note: The Elastic IP address must be in the same region as the stack.

```
aws opsworks register-elastic-ip --region us-east-1 --stack-id
d72553d4-8727-448c-9b00-f024f0ba1b06 --elastic-ip 54.148.130.96
```

Output

```
{
  "ElasticIp": "54.148.130.96"
}
```

More Information

For more information, see *Registering Elastic IP Addresses with a Stack* in the *OpsWorks User Guide*.

- For API details, see [RegisterElasticIp](#) in *AWS CLI Command Reference*.

register-rds-db-instance

The following code example shows how to use `register-rds-db-instance`.

AWS CLI

To register an Amazon RDS instance with a stack

The following example registers an Amazon RDS DB instance, identified by its Amazon Resource Name (ARN), with a specified stack. It also specifies the instance's master username and password. Note that AWS OpsWorks does not validate either of these values. If either one is incorrect, your application will not be able to connect to the database.

```
aws opsworks register-rds-db-instance --region us-east-1 --stack-id
d72553d4-8727-448c-9b00-f024f0ba1b06 --rds-db-instance-arn arn:aws:rds:us-
west-2:123456789012:db:clitestdb --db-user cliuser --db-password some23!pwd
```

Output: None.

More Information

For more information, see *Registering Amazon RDS Instances with a Stack* in the *AWS OpsWorks User Guide*.

- For API details, see [RegisterRdsDbInstance](#) in *AWS CLI Command Reference*.

register-volume

The following code example shows how to use `register-volume`.

AWS CLI

To register an Amazon EBS volume with a stack

The following example registers an Amazon EBS volume, identified by its volume ID, with a specified stack.

```
aws opsworks register-volume --region us-east-1 --stack-id d72553d4-8727-448c-9b00-
f024f0ba1b06 --ec-2-volume-id vol-295c1638
```

Output:


```
{
  "VolumeId": "ee08039c-7cb7-469f-be10-40fb7f0c05e8"
}
```

More Information

For more information, see [Registering Amazon EBS Volumes with a Stack](#) in the *AWS OpsWorks User Guide*.

- For API details, see [RegisterVolume](#) in *AWS CLI Command Reference*.

register

The following code example shows how to use `register`.

AWS CLI

To register instances with a stack

The following examples show a variety of ways to register instances with a stack that were created outside of AWS Opsworks. You can run `register` from the instance to be registered, or from a separate workstation. For more information, see [Registering Amazon EC2 and On-premises Instances](#) in the *AWS OpsWorks User Guide*.

Note: For brevity, the examples omit the `region` argument.

To register an Amazon EC2 instance

To indicate that you are registering an EC2 instance, set the `--infrastructure-class` argument to `ec2`.

The following example registers an EC2 instance with the specified stack from a separate workstation. The instance is identified by its EC2 ID, `i-12345678`. The example uses the workstation's default SSH username and attempts to log in to the instance using authentication techniques that do not require a password, such as a default private SSH key. If that fails, `register` queries for the password.

```
aws opsworks register --infrastructure-class=ec2 --stack-id 935450cc-61e0-4b03-
a3e0-160ac817d2bb i-12345678
```

The following example registers an EC2 instance with the specified stack from a separate workstation. It uses the `--ssh-username` and `--ssh-private-key` arguments to explicitly specify the SSH username and private key file that the command uses to log into the instance. `ec2-user` is the standard username for Amazon Linux instances. Use `ubuntu` for Ubuntu instances.

```
aws opsworks register --infrastructure-class=ec2 --stack-id 935450cc-61e0-4b03-
a3e0-160ac817d2bb --ssh-username ec2-user --ssh-private-key ssh_private_key
i-12345678
```

The following example registers the EC2 instance that is running the `register` command. Log in to the instance with SSH and run `register` with the `--local` argument instead of an instance ID or hostname.

```
aws opsworks register --infrastructure-class ec2 --stack-id 935450cc-61e0-4b03-
a3e0-160ac817d2bb --local
```

To register an on-premises instance

To indicate that you are registering an on-premises instance, set the `--infrastructure-class` argument to `on-premises`.

The following example registers an existing on-premises instance with a specified stack from a separate workstation. The instance is identified by its IP address, `192.0.2.3`. The example uses the workstation's default SSH username and attempts to log in to the instance using authentication techniques that do not require a password, such as a default private SSH key. If that fails, `register` queries for the password.

```
aws opsworks register --infrastructure-class on-premises --stack-id
935450cc-61e0-4b03-a3e0-160ac817d2bb 192.0.2.3
```

The following example registers an on-premises instance with a specified stack from a separate workstation. The instance is identified by its hostname, `host1`. The `--override-...` arguments direct AWS OpsWorks to display `webserver1` as the host name and `192.0.2.3` and `10.0.0.2` as the instance's public and private IP addresses, respectively.

```
aws opsworks register --infrastructure-class on-premises --stack-id
935450cc-61e0-4b03-a3e0-160ac817d2bb --override-hostname webserver1 --override-
public-ip 192.0.2.3 --override-private-ip 10.0.0.2 host1
```

The following example registers an on-premises instance with a specified stack from a separate workstation. The instance is identified by its IP address. `register` logs into the instance using the specified SSH username and private key file.

```
aws opsworks register --infrastructure-class on-premises --stack-id
935450cc-61e0-4b03-a3e0-160ac817d2bb --ssh-username admin --ssh-private-key
ssh_private_key 192.0.2.3
```

The following example registers an existing on-premises instance with a specified stack from a separate workstation. The command logs into the instance using a custom SSH command string that specifies the SSH password and the instance's IP address.

```
aws opsworks register --infrastructure-class on-premises --stack-id
935450cc-61e0-4b03-a3e0-160ac817d2bb --override-ssh "sshpass -p 'mypassword' ssh
your-user@192.0.2.3"
```

The following example registers the on-premises instance that is running the `register` command. Log in to the instance with SSH and run `register` with the `--local` argument instead of an instance ID or hostname.

```
aws opsworks register --infrastructure-class on-premises --stack-id
935450cc-61e0-4b03-a3e0-160ac817d2bb --local
```

Output: The following is typical output for registering an EC2 instance.

```
Warning: Permanently added '52.11.41.206' (ECDSA) to the list of known hosts.
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           %             %             Dload  Upload  Total  Spent    Left  Speed
100 6403k  100 6403k    0     0 2121k      0  0:00:03  0:00:03  --:--:-- 2121k
[Tue, 24 Feb 2015 20:48:37 +0000] opsworks-init: Initializing AWS OpsWorks
environment
[Tue, 24 Feb 2015 20:48:37 +0000] opsworks-init: Running on Ubuntu
[Tue, 24 Feb 2015 20:48:37 +0000] opsworks-init: Checking if OS is supported
[Tue, 24 Feb 2015 20:48:37 +0000] opsworks-init: Running on supported OS
[Tue, 24 Feb 2015 20:48:37 +0000] opsworks-init: Setup motd
[Tue, 24 Feb 2015 20:48:37 +0000] opsworks-init: Executing: ln -sf --backup /etc/
motd.opsworks-static /etc/motd
[Tue, 24 Feb 2015 20:48:37 +0000] opsworks-init: Enabling multiverse repositories
[Tue, 24 Feb 2015 20:48:37 +0000] opsworks-init: Customizing APT environment
[Tue, 24 Feb 2015 20:48:37 +0000] opsworks-init: Installing system packages
```

```
[Tue, 24 Feb 2015 20:48:37 +0000] opsworks-init: Executing: dpkg --configure -a
[Tue, 24 Feb 2015 20:48:37 +0000] opsworks-init: Executing with retry: apt-get
update
[Tue, 24 Feb 2015 20:49:13 +0000] opsworks-init: Executing: apt-get install -y ruby
ruby-dev libicu-dev libssl-dev libxslt-dev libxml2-dev libyaml-dev monit
[Tue, 24 Feb 2015 20:50:13 +0000] opsworks-init: Using assets bucket from
environment: 'opsworks-instance-assets-us-east-1.s3.amazonaws.com'.
[Tue, 24 Feb 2015 20:50:13 +0000] opsworks-init: Installing Ruby for the agent
[Tue, 24 Feb 2015 20:50:13 +0000] opsworks-init: Executing: /tmp/opsworks-
agent-installer.YgGq8wF3UUre6yDy/opsworks-agent-installer/opsworks-agent/bin/
installer_wrapper.sh -r -R opsworks-instance-assets-us-east-1.s3.amazonaws.com
[Tue, 24 Feb 2015 20:50:44 +0000] opsworks-init: Starting the installer
Instance successfully registered. Instance ID: 4d6d1710-ded9-42a1-b08e-b043ad7af1e2
Connection to 52.11.41.206 closed.
```

More Information

For more information, see [Registering an Instance with an AWS OpsWorks Stack in the AWS OpsWorks User Guide](#).

- For API details, see [Register](#) in *AWS CLI Command Reference*.

set-load-based-auto-scaling

The following code example shows how to use `set-load-based-auto-scaling`.

AWS CLI

To set the load-based scaling configuration for a layer

The following example enables load-based scaling for a specified layer and sets the configuration for that layer. You must use `create-instance` to add load-based instances to the layer.

```
aws opsworks --region us-east-1 set-load-based-auto-scaling --layer-id
523569ae-2faf-47ac-b39e-f4c4b381f36d --enable --up-scaling file://upscale.json --
down-scaling file://downscale.json
```

The example puts the upscaling threshold settings in a separate file in the working directory named `upscale.json`, which contains the following.

```
{
```

```
"InstanceCount": 2,  
"ThresholdsWaitTime": 3,  
"IgnoreMetricsTime": 3,  
"CpuThreshold": 85,  
"MemoryThreshold": 85,  
"LoadThreshold": 85  
}
```

The example puts the downscaling threshold settings in a separate file in the working directory named `downscale.json`, which contains the following.

```
{  
  "InstanceCount": 2,  
  "ThresholdsWaitTime": 3,  
  "IgnoreMetricsTime": 3,  
  "CpuThreshold": 35,  
  "MemoryThreshold": 30,  
  "LoadThreshold": 30  
}
```

Output: None.

More Information

For more information, see *Using Automatic Load-based Scaling* in the *AWS OpsWorks User Guide*.

- For API details, see [SetLoadBasedAutoScaling](#) in *AWS CLI Command Reference*.

set-permission

The following code example shows how to use `set-permission`.

AWS CLI

To grant per-stack AWS OpsWorks permission levels

When you import an AWS Identity and Access Management (IAM) user into AWS OpsWorks by calling `create-user-profile`, the user has only those permissions that are granted by the attached IAM policies. You can grant AWS OpsWorks permissions by modifying a user's policies. However, it is often easier to import a user and then use the `set-permission` command to

grant the user one of the standard permission levels for each stack to which the user will need access.

The following example grants permission for the specified stack for a user, who is identified by Amazon Resource Name (ARN). The example grants the user a Manage permissions level, with sudo and SSH privileges on the stack's instances.

```
aws opsworks set-permission --region us-east-1 --stack-id 71c7ca72-55ae-4b6a-8ee1-
a8dcdded3fa0f --level manage --iam-user-arn arn:aws:iam::123456789102:user/cli-user-
test --allow-ssh --allow-sudo
```

Output: None.

More Information

For more information, see [Granting AWS OpsWorks Users Per-Stack Permissions](#) in the *AWS OpsWorks User Guide*.

- For API details, see [SetPermission](#) in *AWS CLI Command Reference*.

set-time-based-auto-scaling

The following code example shows how to use `set-time-based-auto-scaling`.

AWS CLI

To set the time-based scaling configuration for a layer

The following example sets the time-based configuration for a specified instance. You must first use `create-instance` to add the instance to the layer.

```
aws opsworks --region us-east-1 set-time-based-auto-scaling --instance-id
69b6237c-08c0-4edb-a6af-78f3d01cedf2 --auto-scaling-schedule file://schedule.json
```

The example puts the schedule in a separate file in the working directory named `schedule.json`. For this example, the instance is on for a few hours around midday UTC (Coordinated Universal Time) on Monday and Tuesday.

```
{
  "Monday": {
    "10": "on",
```

```
    "11": "on",
    "12": "on",
    "13": "on"
  },
  "Tuesday": {
    "10": "on",
    "11": "on",
    "12": "on",
    "13": "on"
  }
}
```

Output: None.

More Information

For more information, see *Using Automatic Time-based Scaling* in the *AWS OpsWorks User Guide*.

- For API details, see [SetTimeBasedAutoScaling](#) in *AWS CLI Command Reference*.

start-instance

The following code example shows how to use `start-instance`.

AWS CLI

To start an instance

The following `start-instance` command starts a specified 24/7 instance.

```
aws opsworks start-instance --instance-id f705ee48-9000-4890-8bd3-20eb05825aaf
```

Output: None. Use `describe-instances` to check the instance's status.

Tip You can start every offline instance in a stack with one command by calling `start-stack`.

More Information

For more information, see *Manually Starting, Stopping, and Rebooting 24/7 Instances* in the *AWS OpsWorks User Guide*.

- For API details, see [StartInstance](#) in *AWS CLI Command Reference*.

start-stack

The following code example shows how to use `start-stack`.

AWS CLI

To start a stack's instances

The following example starts all of a stack's 24/7 instances. To start a particular instance, use `start-instance`.

```
aws opsworks --region us-east-1 start-stack --stack-id 8c428b08-a1a1-46ce-a5f8-  
feddc43771b8
```

Output: None.

More Information

For more information, see *Starting an Instance* in the *AWS OpsWorks User Guide*.

- For API details, see [StartStack](#) in *AWS CLI Command Reference*.

stop-instance

The following code example shows how to use `stop-instance`.

AWS CLI

To stop an instance

The following example stops a specified instance, which is identified by its instance ID. You can obtain an instance ID by going to the instance's details page on the AWS OpsWorks console or by running the `describe-instances` command.

```
aws opsworks stop-instance --region us-east-1 --instance-id 3a21cfac-4a1f-4ce2-a921-  
b2cfba6f7771
```

You can restart a stopped instance by calling `start-instance` or by deleting the instance by calling `delete-instance`.

Output: None.

More Information

For more information, see *Stopping an Instance in the AWS OpsWorks User Guide*.

- For API details, see [StopInstance](#) in *AWS CLI Command Reference*.

stop-stack

The following code example shows how to use `stop-stack`.

AWS CLI

To stop a stack's instances

The following example stops all of a stack's 24/7 instances. To stop a particular instance, use `stop-instance`.

```
aws opsworks --region us-east-1 stop-stack --stack-id 8c428b08-a1a1-46ce-a5f8-  
feddc43771b8
```

Output: No output.

More Information

For more information, see *Stopping an Instance in the AWS OpsWorks User Guide*.

- For API details, see [StopStack](#) in *AWS CLI Command Reference*.

unassign-instance

The following code example shows how to use `unassign-instance`.

AWS CLI

To unassign a registered instance from its layers

The following `unassign-instance` command unassigns an instance from its attached layers.

```
aws opsworks --region us-east-1 unassign-instance --instance-id 4d6d1710-ded9-42a1-  
b08e-b043ad7af1e2
```

Output: None.

More Information

For more information, see *Unassigning a Registered Instance in the AWS OpsWorks User Guide*.

- For API details, see [UnassignInstance](#) in *AWS CLI Command Reference*.

unassign-volume

The following code example shows how to use `unassign-volume`.

AWS CLI

To unassign a volume from its instance

The following example unassigns a registered Amazon Elastic Block Store (Amazon EBS) volume from its instance. The volume is identified by its volume ID, which is the GUID that AWS OpsWorks assigns when you register the volume with a stack, not the Amazon Elastic Compute Cloud (Amazon EC2) volume ID.

```
aws opsworks --region us-east-1 unassign-volume --volume-id 8430177d-52b7-4948-9c62-
e195af4703df
```

Output: None.

More Information

For more information, see *Unassigning Amazon EBS Volumes in the AWS OpsWorks User Guide*.

- For API details, see [UnassignVolume](#) in *AWS CLI Command Reference*.

update-app

The following code example shows how to use `update-app`.

AWS CLI

To update an app

The following example updates a specified app to change its name.

```
aws opsworks --region us-east-1 update-app --app-id 26a61ead-d201-47e3-
b55c-2a7c666942f8 --name NewAppName
```

Output: None.

More Information

For more information, see *Editing Apps* in the *AWS OpsWorks User Guide*.

- For API details, see [UpdateApp](#) in *AWS CLI Command Reference*.

update-elastic-ip

The following code example shows how to use `update-elastic-ip`.

AWS CLI

To update an Elastic IP address name

The following example updates the name of a specified Elastic IP address.

```
aws opsworks --region us-east-1 update-elastic-ip --elastic-ip 54.148.130.96 --name  
NewIPName
```

Output: None.

More Information

For more information, see *Resource Management* in the *AWS OpsWorks User Guide*.

- For API details, see [UpdateElasticIp](#) in *AWS CLI Command Reference*.

update-instance

The following code example shows how to use `update-instance`.

AWS CLI

To update an instance

The following example updates a specified instance's type.

```
aws opsworks --region us-east-1 update-instance --instance-id  
dfe18b02-5327-493d-91a4-c5c0c448927f --instance-type c3.xlarge
```

Output: None.

More Information

For more information, see *Editing the Instance Configuration in the AWS OpsWorks User Guide*.

- For API details, see [UpdateInstance](#) in *AWS CLI Command Reference*.

update-layer

The following code example shows how to use `update-layer`.

AWS CLI

To update a layer

The following example updates a specified layer to use Amazon EBS-optimized instances.

```
aws opsworks --region us-east-1 update-layer --layer-id
888c5645-09a5-4d0e-95a8-812ef1db76a4 --use-efs-optimized-instances
```

Output: None.

More Information

For more information, see *Editing an OpsWorks Layer's Configuration in the AWS OpsWorks User Guide*.

- For API details, see [UpdateLayer](#) in *AWS CLI Command Reference*.

update-my-user-profile

The following code example shows how to use `update-my-user-profile`.

AWS CLI

To update a user's profile

The following example updates the development user's profile to use a specified SSH public key. The user's AWS credentials are represented by the development profile in the credentials file (`~\.aws\credentials`), and the key is in a `.pem` file in the working directory.

```
aws opsworks --region us-east-1 --profile development update-my-user-profile --ssh-
public-key file://development_key.pem
```

Output: None.

More Information

For more information, see *Editing AWS OpsWorks User Settings* in the *AWS OpsWorks User Guide*.

- For API details, see [UpdateMyUserProfile](#) in *AWS CLI Command Reference*.

update-rds-db-instance

The following code example shows how to use `update-rds-db-instance`.

AWS CLI

To update a registered Amazon RDS DB instance

The following example updates an Amazon RDS instance's master password value. Note that this command does not change the RDS instance's master password, just the password that you provide to AWS OpsWorks. If this password does not match the RDS instance's password, your application will not be able to connect to the database.

```
aws opsworks --region us-east-1 update-rds-db-instance --db-password 123456789
```

Output: None.

More Information

For more information, see *Registering Amazon RDS Instances with a Stack* in the *AWS OpsWorks User Guide*.

- For API details, see [UpdateRdsDbInstance](#) in *AWS CLI Command Reference*.

update-volume

The following code example shows how to use `update-volume`.

AWS CLI

To update a registered volume

The following example updates a registered Amazon Elastic Block Store (Amazon EBS) volume's mount point. The volume is identified by its volume ID, which is the GUID that AWS OpsWorks assigns to the volume when you register it with a stack, not the Amazon Elastic Compute Cloud (Amazon EC2) volume ID.:

```
aws opsworks --region us-east-1 update-volume --volume-id 8430177d-52b7-4948-9c62-
e195af4703df --mount-point /mnt/myvol
```

Output: None.

More Information

For more information, see *Assigning Amazon EBS Volumes to an Instance in the AWS OpsWorks User Guide*.

- For API details, see [UpdateVolume](#) in *AWS CLI Command Reference*.

AWS OpsWorks CM examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS OpsWorks CM.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-node

The following code example shows how to use `associate-node`.

AWS CLI

To associate nodes

The following `associate-node` command associates a node named `i-44de882p` with a Chef Automate server named `automate-06`, meaning that the `automate-06` server manages the node, and communicates recipe commands to the node through `chef-client` agent software that is installed on the node by the `associate-node` command. Valid node names are EC2 instance IDs.:

```
aws opsworks-cm associate-node --server-name "automate-06" --node-name
  "i-43de882p" --engine-attributes "Name=CHEF_ORGANIZATION,Value='MyOrganization'
  Name=CHEF_NODE_PUBLIC_KEY,Value='Public_key_contents'"
```

The output returned by the command resembles the following. *Output:*

```
{
  "NodeAssociationStatusToken": "AHUY8wFe4pdXtZC5DiJa5S0Lp5o14DH//
  rHRqHDWXxwVoNBxcEy4V7R0NOFymh7E/1Hum0BPsemPQFE6dcGaiFk"
}
```

More Information

For more information, see *Adding Nodes Automatically in AWS OpsWorks for Chef Automate* in the *AWS OpsWorks User Guide*.

- For API details, see [AssociateNode](#) in *AWS CLI Command Reference*.

create-backup

The following code example shows how to use `create-backup`.

AWS CLI

To create backups

The following `create-backup` command starts a manual backup of a Chef Automate server named `automate-06` in the `us-east-1` region. The command adds a descriptive message to the backup in the `--description` parameter.

```
aws opsworks-cm create-backup \
```

```
--server-name 'automate-06' \  
--description "state of my infrastructure at launch"
```

The output shows you information similar to the following about the new backup.

Output:

```
{  
  "Backups": [  
    {  
      "BackupArn": "string",  
      "BackupId": "automate-06-20160729133847520",  
      "BackupType": "MANUAL",  
      "CreatedAt": "2016-07-29T13:38:47.520Z",  
      "Description": "state of my infrastructure at launch",  
      "Engine": "Chef",  
      "EngineModel": "Single",  
      "EngineVersion": "12",  
      "InstanceProfileArn": "arn:aws:iam::1019881987024:instance-profile/  
automate-06-1010V4UU2WRM2",  
      "InstanceType": "m4.large",  
      "KeyPair": "",  
      "PreferredBackupWindow": "",  
      "PreferredMaintenanceWindow": "",  
      "S3LogUrl": "https://s3.amazonaws.com/<bucket-name>/  
automate-06-20160729133847520",  
      "SecurityGroupIds": [ "sg-1a24c270" ],  
      "ServerName": "automate-06",  
      "ServiceRoleArn": "arn:aws:iam::1019881987024:role/aws-opsworks-cm-  
service-role.1114810729735",  
      "Status": "OK",  
      "StatusDescription": "",  
      "SubnetIds": [ "subnet-49436a18" ],  
      "ToolsVersion": "string",  
      "UserArn": "arn:aws:iam::1019881987024:user/opsworks-user"  
    }  
  ],  
}
```

For more information, see [Back Up and Restore an AWS OpsWorks for Chef Automate Server](#) in the *AWS OpsWorks User Guide*.

- For API details, see [CreateBackup](#) in *AWS CLI Command Reference*.

create-server

The following code example shows how to use create-server.

AWS CLI

To create a server

The following create-server example creates a new Chef Automate server named automate-06 in your default region. Note that defaults are used for most other settings, such as number of backups to retain, and maintenance and backup start times. Before you run a create-server command, complete prerequisites in [Getting Started with AWS OpsWorks for Chef Automate](#) in the *AWS Opsworks for Chef Automate User Guide*.

```
aws opsworks-cm create-server \  
  --engine "Chef" \  
  --engine-model "Single" \  
  --engine-version "12" \  
  --server-name "automate-06" \  
  --instance-profile-arn "arn:aws:iam::1019881987024:instance-profile/aws-opsworks-cm-ec2-role" \  
  --instance-type "t2.medium" \  
  --key-pair "amazon-test" \  
  --service-role-arn "arn:aws:iam::044726508045:role/aws-opsworks-cm-service-role"
```

The output shows you information similar to the following about the new server:

```
{  
  "Server": {  
    "BackupRetentionCount": 10,  
    "CreatedAt": 2016-07-29T13:38:47.520Z,  
    "DisableAutomatedBackup": FALSE,  
    "Endpoint": "https://opsworks-cm.us-east-1.amazonaws.com",  
    "Engine": "Chef",  
    "EngineAttributes": [  
      {  
        "Name": "CHEF_DELIVERY_ADMIN_PASSWORD",  
        "Value": "1Password1"  
      }  
    ],  
    "EngineModel": "Single",  
    "EngineVersion": "12",
```

```

    "InstanceProfileArn": "arn:aws:iam::1019881987024:instance-profile/aws-
opsworks-cm-ec2-role",
    "InstanceType": "t2.medium",
    "KeyPair": "amazon-test",
    "MaintenanceStatus": "",
    "PreferredBackupWindow": "Sun:02:00",
    "PreferredMaintenanceWindow": "00:00",
    "SecurityGroupIds": [ "sg-1a24c270" ],
    "ServerArn": "arn:aws:iam::1019881987024:instance/
automate-06-1010V4UU2WRM2",
    "ServerName": "automate-06",
    "ServiceRoleArn": "arn:aws:iam::1019881987024:role/aws-opsworks-cm-service-
role",
    "Status": "CREATING",
    "StatusReason": "",
    "SubnetIds": [ "subnet-49436a18" ]
  }
}

```

For more information, see [UpdateServer](#) in the *AWS OpsWorks for Chef Automate API Reference*.

- For API details, see [CreateServer](#) in *AWS CLI Command Reference*.

delete-backup

The following code example shows how to use delete-backup.

AWS CLI

To delete backups

The following delete-backup command deletes a manual or automated backup of a Chef Automate server, identified by the backup ID. This command is useful when you are approaching the maximum number of backups that you can save, or you want to minimize your Amazon S3 storage costs.:

```
aws opsworks-cm delete-backup --backup-id "automate-06-2016-11-19T23:42:40.240Z"
```

The output shows whether the backup deletion succeeded.

More Information

For more information, see [Back Up and Restore an AWS OpsWorks for Chef Automate Server](#) in the *AWS OpsWorks User Guide*.

- For API details, see [DeleteBackup](#) in *AWS CLI Command Reference*.

delete-server

The following code example shows how to use `delete-server`.

AWS CLI

To delete servers

The following `delete-server` command deletes a Chef Automate server, identified by the server's name. After the server is deleted, it is no longer returned by `DescribeServer` requests.:

```
aws opsworks-cm delete-server --server-name "automate-06"
```

The output shows whether the server deletion succeeded.

More Information

For more information, see [Delete an AWS OpsWorks for Chef Automate Server](#) in the *AWS OpsWorks User Guide*.

- For API details, see [DeleteServer](#) in *AWS CLI Command Reference*.

describe-account-attributes

The following code example shows how to use `describe-account-attributes`.

AWS CLI

To describe account attributes

The following `describe-account-attributes` command returns information about your account's usage of AWS OpsWorks for Chef Automate resources.:

```
aws opsworks-cm describe-account-attributes
```

The output for each account attribute entry returned by the command resembles the following.

Output:

```
{
  "Attributes": [
    {
      "Maximum": 5,
      "Name": "ServerLimit",
      "Used": 2
    }
  ]
}
```

More Information

For more information, see `DescribeAccountAttributes` in the *AWS OpsWorks for Chef Automate API Reference*.

- For API details, see [DescribeAccountAttributes](#) in *AWS CLI Command Reference*.

describe-backups

The following code example shows how to use `describe-backups`.

AWS CLI

To describe backups

The following `describe-backups` command returns information about all backups associated with your account in your default region.

```
aws opsworks-cm describe-backups
```

The output for each backup entry returned by the command resembles the following.

Output:

```
{
  "Backups": [
    {
      "BackupArn": "string",
```

```

    "BackupId": "automate-06-20160729133847520",
    "BackupType": "MANUAL",
    "CreatedAt": 2016-07-29T13:38:47.520Z,
    "Description": "state of my infrastructure at launch",
    "Engine": "Chef",
    "EngineModel": "Single",
    "EngineVersion": "12",
    "InstanceProfileArn": "arn:aws:iam::1019881987024:instance-profile/
automate-06-1010V4UU2WRM2",
    "InstanceType": "m4.large",
    "KeyPair": "",
    "PreferredBackupWindow": "",
    "PreferredMaintenanceWindow": "",
    "S3LogUrl": "https://s3.amazonaws.com/<bucket-name>/
automate-06-20160729133847520",
    "SecurityGroupIds": [ "sg-1a24c270" ],
    "ServerName": "automate-06",
    "ServiceRoleArn": "arn:aws:iam::1019881987024:role/aws-opsworks-cm-
service-role.1114810729735",
    "Status": "Successful",
    "StatusDescription": "",
    "SubnetIds": [ "subnet-49436a18" ],
    "ToolsVersion": "string",
    "UserArn": "arn:aws:iam::1019881987024:user/opsworks-user"
  },
}

```

For more information, see [Back Up and Restore an AWS OpsWorks for Chef Automate Server](#) in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeBackups](#) in *AWS CLI Command Reference*.

describe-events

The following code example shows how to use `describe-events`.

AWS CLI

To describe events

The following `describe-events` example returns information about all events that are associated with the specified Chef Automate server.

```
aws opsworks-cm describe-events \
  --server-name 'automate-06'
```

The output for each event entry returned by the command resembles the following example:

```
{
  "ServerEvents": [
    {
      "CreatedAt": 2016-07-29T13:38:47.520Z,
      "LogUrl": "https://s3.amazonaws.com/<bucket-name>/
automate-06-20160729133847520",
      "Message": "Updates successfully installed.",
      "ServerName": "automate-06"
    }
  ]
}
```

For more information, see [General Troubleshooting Tips](#) in the *AWS OpsWorks User Guide*.

- For API details, see [DescribeEvents](#) in *AWS CLI Command Reference*.

describe-node-association-status

The following code example shows how to use `describe-node-association-status`.

AWS CLI

To describe node association status

The following `describe-node-association-status` command returns the status of a request to associate a node with a Chef Automate server named `automate-06`:

```
aws opsworks-cm describe-node-association-status --server-
name "automate-06" --node-association-status-token "Af1JKl+/
GoKLZJBdDQEx0065CDi57b1Qe9nKM8joSok0pQ9xr8DqApBN9/106sLdSvlfDEKkEx+eoCHvjowHa0s="
```

The output for each account attribute entry returned by the command resembles the following.
Output:

```
{
  "NodeAssociationStatus": "IN_PROGRESS"
```

```
}
```

More Information

For more information, see `DescribeNodeAssociationStatus` in the *AWS OpsWorks for Chef Automate API Reference*.

- For API details, see [DescribeNodeAssociationStatus](#) in *AWS CLI Command Reference*.

describe-servers

The following code example shows how to use `describe-servers`.

AWS CLI

To describe servers

The following `describe-servers` command returns information about all servers that are associated with your account, and in your default region.:

```
aws opsworks-cm describe-servers
```

The output for each server entry returned by the command resembles the following. *Output*:

```
{
  "Servers": [
    {
      "BackupRetentionCount": 8,
      "CreatedAt": "2016-07-29T13:38:47.520Z",
      "DisableAutomatedBackup": FALSE,
      "Endpoint": "https://opsworks-cm.us-east-1.amazonaws.com",
      "Engine": "Chef",
      "EngineAttributes": [
        {
          "Name": "CHEF_DELIVERY_ADMIN_PASSWORD",
          "Value": "1Password1"
        }
      ],
      "EngineModel": "Single",
      "EngineVersion": "12",
      "InstanceProfileArn": "arn:aws:iam::1019881987024:instance-profile/
automate-06-1010V4UU2WRM2",

```

```

    "InstanceType": "m4.large",
    "KeyPair": "",
    "MaintenanceStatus": "SUCCESS",
    "PreferredBackupWindow": "03:00",
    "PreferredMaintenanceWindow": "Mon:09:00",
    "SecurityGroupIds": [ "sg-1a24c270" ],
    "ServerArn": "arn:aws:iam::1019881987024:instance/automate-06-1010V4UU2WRM2",
    "ServerName": "automate-06",
    "ServiceRoleArn": "arn:aws:iam::1019881987024:role/aws-opsworks-cm-service-
role.1114810729735",
    "Status": "HEALTHY",
    "StatusReason": "",
    "SubnetIds": [ "subnet-49436a18" ]
  }
]
}

```

More Information

For more information, see `DescribeServers` in the *AWS OpsWorks for Chef Automate API Guide*.

- For API details, see [DescribeServers](#) in *AWS CLI Command Reference*.

disassociate-node

The following code example shows how to use `disassociate-node`.

AWS CLI

To disassociate nodes

The following `disassociate-node` command disassociates a node named `i-44de882p`, removing the node from management by a Chef Automate server named `automate-06`. Valid node names are EC2 instance IDs.:

```

aws opsworks-cm disassociate-node --server-name "automate-06" --node-name
  "i-43de882p" --engine-attributes "Name=CHEF_ORGANIZATION,Value='MyOrganization'
  Name=CHEF_NODE_PUBLIC_KEY,Value='Public_key_contents'"

```

The output returned by the command resembles the following. *Output*:

```
{
```



```
"NodeAssociationStatusToken": "AHUY8wFe4pdXtZC5DiJa5S0Lp5o14DH//  
rHRqHDWXxwVoNBxcEy4V7R0N0Fymh7E/1Hum0BPsemPQFE6dcGaiFk"  
}
```

More Information

For more information, see [Delete an AWS OpsWorks for Chef Automate Server](#) in the *AWS OpsWorks User Guide*.

- For API details, see [DisassociateNode](#) in *AWS CLI Command Reference*.

restore-server

The following code example shows how to use `restore-server`.

AWS CLI

To restore a server

The following `restore-server` command performs an in-place restoration of a Chef Automate server named `automate-06` in your default region from a backup with an ID of `automate-06-2016-11-22T16:13:27.998Z`. Restoring a server restores connections to the nodes that the Chef Automate server was managing at the time that the specified backup was performed.

```
aws opsworks-cm restore-server --backup-id "automate-06-2016-11-22T16:13:27.998Z" --  
server-name "automate-06"
```

The output is the command ID only. *Output:*

```
(None)
```

More Information

For more information, see [Restore a Failed AWS OpsWorks for Chef Automate Server](#) in the *AWS OpsWorks User Guide*.

- For API details, see [RestoreServer](#) in *AWS CLI Command Reference*.

start-maintenance

The following code example shows how to use `start-maintenance`.

AWS CLI

To start maintenance

The following `start-maintenance` example manually starts maintenance on the specified Chef Automate or Puppet Enterprise server in your default region. This command is useful if an earlier, automated maintenance attempt failed, and the underlying cause of maintenance failure has been resolved.

```
aws opsworks-cm start-maintenance \  
  --server-name 'automate-06'
```

Output:

```
{  
  "Server": {  
    "AssociatePublicIpAddress": true,  
    "BackupRetentionCount": 10,  
    "ServerName": "automate-06",  
    "CreatedAt": 1569229584.842,  
    "CloudFormationStackArn": "arn:aws:cloudformation:us-  
west-2:123456789012:stack/aws-opsworks-cm-instance-automate-06-1606611794746/  
EXAMPLE0-31de-11eb-bdb0-0a5b0a1353b8",  
    "DisableAutomatedBackup": false,  
    "Endpoint": "automate-06-EXAMPLEEvr8gjfk5f.us-west-2.opsworks-cm.io",  
    "Engine": "ChefAutomate",  
    "EngineModel": "Single",  
    "EngineAttributes": [],  
    "EngineVersion": "2020-07",  
    "InstanceProfileArn": "arn:aws:iam::123456789012:instance-profile/aws-  
opsworks-cm-ec2-role",  
    "InstanceType": "m5.large",  
    "PreferredMaintenanceWindow": "Sun:01:00",  
    "PreferredBackupWindow": "Sun:15:00",  
    "SecurityGroupIds": [  
      "sg-EXAMPLE"  
    ],  
    "ServiceRoleArn": "arn:aws:iam::123456789012:role/service-role/aws-opsworks-  
cm-service-role",  
    "Status": "UNDER_MAINTENANCE",  
    "SubnetIds": [  
      "subnet-EXAMPLE"  
    ]  
  }  
}
```

```

    ],
    "ServerArn": "arn:aws:opsworks-cm:us-west-2:123456789012:server/
automate-06/0148382d-66b0-4196-8274-d1a2b6dff8d1"
  }
}

```

For more information, see [System Maintenance \(Puppet Enterprise servers\)](#) or [System Maintenance \(Chef Automate servers\)](#) in the *AWS OpsWorks User Guide*.

- For API details, see [StartMaintenance](#) in *AWS CLI Command Reference*.

update-server-engine-attributes

The following code example shows how to use `update-server-engine-attributes`.

AWS CLI

To update server engine attributes

The following `update-server-engine-attributes` command updates the value of the `CHEF_PIVOTAL_KEY` engine attribute for a Chef Automate server named `automate-06`. It is currently not possible to change the value of other engine attributes.

```

aws opsworks-cm update-server-engine-attributes \
  --attribute-name CHEF_PIVOTAL_KEY \
  --attribute-value "new key value" \
  --server-name "automate-06"

```

The output shows you information similar to the following about the updated server.

```

{
  "Server": {
    "BackupRetentionCount": 2,
    "CreatedAt": 2016-07-29T13:38:47.520Z,
    "DisableAutomatedBackup": FALSE,
    "Endpoint": "https://opsworks-cm.us-east-1.amazonaws.com",
    "Engine": "Chef",
    "EngineAttributes": [
      {
        "Name": "CHEF_PIVOTAL_KEY",
        "Value": "new key value"
      }
    ]
  }
}

```

```

    }
  ],
  "EngineModel": "Single",
  "EngineVersion": "12",
  "InstanceProfileArn": "arn:aws:iam::1019881987024:instance-profile/
automate-06-1010V4UU2WRM2",
  "InstanceType": "m4.large",
  "KeyPair": "",
  "MaintenanceStatus": "SUCCESS",
  "PreferredBackupWindow": "Mon:09:15",
  "PreferredMaintenanceWindow": "03:00",
  "SecurityGroupIds": [ "sg-1a24c270" ],
  "ServerArn": "arn:aws:iam::1019881987024:instance/
automate-06-1010V4UU2WRM2",
  "ServerName": "automate-06",
  "ServiceRoleArn": "arn:aws:iam::1019881987024:role/aws-opsworks-cm-service-
role.1114810729735",
  "Status": "HEALTHY",
  "StatusReason": "",
  "SubnetIds": [ "subnet-49436a18" ]
}
}

```

For more information, see [UpdateServerEngineAttributes](#) in the *AWS OpsWorks for Chef Automate API Reference*.

- For API details, see [UpdateServerEngineAttributes](#) in *AWS CLI Command Reference*.

update-server

The following code example shows how to use `update-server`.

AWS CLI

To update a server

The following `update-server` command updates the maintenance start time of the specified Chef Automate server in your default region. The `--preferred-maintenance-window` parameter is added to change the start day and time of server maintenance to Mondays at 9:15 a.m. UTC.:

```
aws opsworks-cm update-server \
```

```
--server-name "automate-06" \  
--preferred-maintenance-window "Mon:09:15"
```

The output shows you information similar to the following about the updated server.

```
{  
  "Server": {  
    "BackupRetentionCount": 8,  
    "CreatedAt": 2016-07-29T13:38:47.520Z,  
    "DisableAutomatedBackup": TRUE,  
    "Endpoint": "https://opsworks-cm.us-east-1.amazonaws.com",  
    "Engine": "Chef",  
    "EngineAttributes": [  
      {  
        "Name": "CHEF_DELIVERY_ADMIN_PASSWORD",  
        "Value": "1Password1"  
      }  
    ],  
    "EngineModel": "Single",  
    "EngineVersion": "12",  
    "InstanceProfileArn": "arn:aws:iam::1019881987024:instance-profile/  
automate-06-1010V4UU2WRM2",  
    "InstanceType": "m4.large",  
    "KeyPair": "",  
    "MaintenanceStatus": "OK",  
    "PreferredBackupWindow": "Mon:09:15",  
    "PreferredMaintenanceWindow": "03:00",  
    "SecurityGroupIds": [ "sg-1a24c270" ],  
    "ServerArn": "arn:aws:iam::1019881987024:instance/  
automate-06-1010V4UU2WRM2",  
    "ServerName": "automate-06",  
    "ServiceRoleArn": "arn:aws:iam::1019881987024:role/aws-opsworks-cm-service-  
role.1114810729735",  
    "Status": "HEALTHY",  
    "StatusReason": "",  
    "SubnetIds": [ "subnet-49436a18" ]  
  }  
}
```

For more information, see [UpdateServer](#) in the *AWS OpsWorks for Chef Automate API Reference*.

- For API details, see [UpdateServer](#) in *AWS CLI Command Reference*.

Organizations examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Organizations.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

accept-handshake

The following code example shows how to use `accept-handshake`.

AWS CLI

To accept a handshake from another account

Bill, the owner of an organization, has previously invited Juan's account to join his organization. The following example shows Juan's account accepting the handshake and thus agreeing to the invitation.

```
aws organizations accept-handshake --handshake-id h-examplehandshakeid111
```

The output shows the following:

```
{
  "Handshake": {
    "Action": "INVITE",
```

```

    "Arn": "arn:aws:organizations::111111111111:handshake/o-
exampleorgid/invite/h-examplehandshakeid111",
    "RequestedTimestamp": 1481656459.257,
    "ExpirationTimestamp": 1482952459.257,
    "Id": "h-examplehandshakeid111",
    "Parties": [
      {
        "Id": "o-exampleorgid",
        "Type": "ORGANIZATION"
      },
      {
        "Id": "juan@example.com",
        "Type": "EMAIL"
      }
    ],
    "Resources": [
      {
        "Resources": [
          {
            "Type": "MASTER_EMAIL",
            "Value": "bill@amazon.com"
          },
          {
            "Type": "MASTER_NAME",
            "Value": "Org Master Account"
          },
          {
            "Type": "ORGANIZATION_FEATURE_SET",
            "Value": "ALL"
          }
        ],
        "Type": "ORGANIZATION",
        "Value": "o-exampleorgid"
      },
      {
        "Type": "EMAIL",
        "Value": "juan@example.com"
      }
    ],
    "State": "ACCEPTED"
  }
}

```

- For API details, see [AcceptHandshake](#) in *AWS CLI Command Reference*.

attach-policy

The following code example shows how to use `attach-policy`.

AWS CLI

To attach a policy to a root, OU, or account

Example 1

The following example shows how to attach a service control policy (SCP) to an OU:

```
aws organizations attach-policy
    --policy-id p-examplepolicyid111
    --target-id ou-examplerootid111-exampleouid111
```

Example 2

The following example shows how to attach a service control policy directly to an account:

```
aws organizations attach-policy
    --policy-id p-examplepolicyid111
    --target-id 333333333333
```

- For API details, see [AttachPolicy](#) in *AWS CLI Command Reference*.

cancel-handshake

The following code example shows how to use `cancel-handshake`.

AWS CLI

To cancel a handshake sent from another account

Bill previously sent an invitation to Susan's account to join his organization. He changes his mind and decides to cancel the invitation before Susan accepts it. The following example shows Bill's cancellation:

```
aws organizations cancel-handshake --handshake-id h-examplehandshakeid111
```


The output includes a handshake object that shows that the state is now CANCELED:

```
{
  "Handshake": {
    "Id": "h-examplehandshakeid111",
    "State": "CANCELED",
    "Action": "INVITE",
    "Arn": "arn:aws:organizations::111111111111:handshake/o-
exampleorgid/invite/h-examplehandshakeid111",
    "Parties": [
      {
        "Id": "o-exampleorgid",
        "Type": "ORGANIZATION"
      },
      {
        "Id": "susan@example.com",
        "Type": "EMAIL"
      }
    ],
    "Resources": [
      {
        "Type": "ORGANIZATION",
        "Value": "o-exampleorgid",
        "Resources": [
          {
            "Type": "MASTER_EMAIL",
            "Value": "bill@example.com"
          },
          {
            "Type": "MASTER_NAME",
            "Value": "Master Account"
          },
          {
            "Type": "ORGANIZATION_FEATURE_SET",
            "Value": "CONSOLIDATED_BILLING"
          }
        ]
      },
      {
        "Type": "EMAIL",
        "Value": "anika@example.com"
      },
      {
        "Type": "NOTES",
```

```
        "Value": "This is a request for Susan's account to  
join Bob's organization."  
      }  
    ],  
    "RequestedTimestamp": 1.47008383521E9,  
    "ExpirationTimestamp": 1.47137983521E9  
  }  
}
```

- For API details, see [CancelHandshake](#) in *AWS CLI Command Reference*.

create-account

The following code example shows how to use create-account.

AWS CLI

To create a member account that is automatically part of the organization

The following example shows how to create a member account in an organization. The member account is configured with the name Production Account and the email address of susan@example.com. Organizations automatically creates an IAM role using the default name of OrganizationAccountAccessRole because the roleName parameter is not specified. Also, the setting that allows IAM users or roles with sufficient permissions to access account billing data is set to the default value of ALLOW because the iamUserAccessToBilling parameter is not specified. Organizations automatically sends Susan a "Welcome to AWS" email:

```
aws organizations create-account --email susan@example.com --account-name  
"Production Account"
```

The output includes a request object that shows that the status is now IN_PROGRESS:

```
{  
  "CreateAccountStatus": {  
    "State": "IN_PROGRESS",  
    "Id": "car-examplecreateaccountrequestid111"  
  }  
}
```

You can later query the current status of the request by providing the Id response value to the describe-create-account-status command as the value for the create-account-request-id parameter.

For more information, see *Creating an AWS Account in Your Organization* in the *AWS Organizations Users Guide*.

- For API details, see [CreateAccount](#) in *AWS CLI Command Reference*.

create-organization

The following code example shows how to use create-organization.

AWS CLI

Example 1: To create a new organization

Bill wants to create an organization using credentials from account 111111111111. The following example shows that the account becomes the master account in the new organization. Because he does not specify a features set, the new organization defaults to all features enabled and service control policies are enabled on the root.

```
aws organizations create-organization
```

The output includes an organization object with details about the new organization:

```
{
  "Organization": {
    "AvailablePolicyTypes": [
      {
        "Status": "ENABLED",
        "Type": "SERVICE_CONTROL_POLICY"
      }
    ],
    "MasterAccountId": "111111111111",
    "MasterAccountArn": "arn:aws:organizations::111111111111:account/o-
exampleorgid/111111111111",
    "MasterAccountEmail": "bill@example.com",
    "FeatureSet": "ALL",
    "Id": "o-exampleorgid",
    "Arn": "arn:aws:organizations::111111111111:organization/o-
exampleorgid"
```

```
}  
}
```

Example 2: To create a new organization with only consolidated billing features enabled

The following example creates an organization that supports only the consolidated billing features:

```
aws organizations create-organization --feature-set CONSOLIDATED_BILLING
```

The output includes an organization object with details about the new organization:

```
{  
  "Organization": {  
    "Arn": "arn:aws:organizations::111111111111:organization/o-  
exampleorgid",  
    "AvailablePolicyTypes": [],  
    "Id": "o-exampleorgid",  
    "MasterAccountArn": "arn:aws:organizations::111111111111:account/o-  
exampleorgid/111111111111",  
    "MasterAccountEmail": "bill@example.com",  
    "MasterAccountId": "111111111111",  
    "FeatureSet": "CONSOLIDATED_BILLING"  
  }  
}
```

For more information, see *Creating an Organization* in the *AWS Organizations Users Guide*.

- For API details, see [CreateOrganization](#) in *AWS CLI Command Reference*.

create-organizational-unit

The following code example shows how to use `create-organizational-unit`.

AWS CLI

To create an OU in a root or parent OU

The following example shows how to create an OU that is named `AccountingOU`:

```
aws organizations create-organizational-unit --parent-id r-examplerootid111 --name  
AccountingOU
```

The output includes an `OrganizationalUnit` object with details about the new OU:

```
{
  "OrganizationalUnit": {
    "Id": "ou-examplerootid111-exampleoid111",
    "Arn": "arn:aws:organizations::111111111111:ou/o-exampleorgid/ou-
    exemplerooid111-exampleoid111",
    "Name": "AccountingOU"
  }
}
```

- For API details, see [CreateOrganizationalUnit](#) in *AWS CLI Command Reference*.

create-policy

The following code example shows how to use `create-policy`.

AWS CLI

Example 1: To create a policy with a text source file for the JSON policy

The following example shows you how to create an service control policy (SCP) named `AllowAllS3Actions`. The policy contents are taken from a file on the local computer called `policy.json`.

```
aws organizations create-policy --content file://policy.json --name
AllowAllS3Actions, --type SERVICE_CONTROL_POLICY --description "Allows delegation
of all S3 actions"
```

The output includes a policy object with details about the new policy:

```
{
  "Policy": {
    "Content": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":
    \"Allow\",\"Action\":[\"s3:*\"],\"Resource\":[\"*\"]}]}",
    "PolicySummary": {
      "Arn": "arn:aws:organizations::o-exampleorgid:policy/
      service_control_policy/p-examplepolicyid111",
      "Description": "Allows delegation of all S3 actions",
      "Name": "AllowAllS3Actions",
      "Type": "SERVICE_CONTROL_POLICY"
    }
  }
}
```

```

    }
  }
}

```

Example 2: To create a policy with a JSON policy as a parameter

The following example shows you how to create the same SCP, this time by embedding the policy contents as a JSON string in the parameter. The string must be escaped with backslashes before the double quotes to ensure that they are treated as literals in the parameter, which itself is surrounded by double quotes:

```

aws organizations create-policy --content "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",\"Action\":[\"s3:*\"],\"Resource\":[\"*\"]}]}\" --name AllowAllS3Actions --type SERVICE_CONTROL_POLICY --description \"Allows delegation of all S3 actions\"

```

For more information about creating and using policies in your organization, see *Managing Organization Policies* in the *AWS Organizations User Guide*.

- For API details, see [CreatePolicy](#) in *AWS CLI Command Reference*.

decline-handshake

The following code example shows how to use `decline-handshake`.

AWS CLI

To decline a handshake sent from another account

The following example shows that Susan, an admin who is the owner of account 222222222222, declines an invitation to join Bill's organization. The `DeclineHandshake` operation returns a handshake object, showing that the state is now `DECLINED`:

```

aws organizations decline-handshake --handshake-id h-examplehandshakeid111

```

The output includes a handshake object that shows the new state of `DECLINED`:

```

{
  "Handshake": {
    "Id": "h-examplehandshakeid111",
    "State": "DECLINED",

```

```

    "Resources": [
      {
        "Type": "ORGANIZATION",
        "Value": "o-exampleorgid",
        "Resources": [
          {
            "Type": "MASTER_EMAIL",
            "Value": "bill@example.com"
          },
          {
            "Type": "MASTER_NAME",
            "Value": "Master Account"
          }
        ]
      },
      {
        "Type": "EMAIL",
        "Value": "susan@example.com"
      },
      {
        "Type": "NOTES",
        "Value": "This is an invitation to Susan's account
to join the Bill's organization."
      }
    ],
    "Parties": [
      {
        "Type": "EMAIL",
        "Id": "susan@example.com"
      },
      {
        "Type": "ORGANIZATION",
        "Id": "o-exampleorgid"
      }
    ],
    "Action": "INVITE",
    "RequestedTimestamp": 1470684478.687,
    "ExpirationTimestamp": 1471980478.687,
    "Arn": "arn:aws:organizations::111111111111:handshake/o-
exampleorgid/invite/h-examplehandshakeid111"
  }
}

```

- For API details, see [DeclineHandshake](#) in *AWS CLI Command Reference*.

delete-organization

The following code example shows how to use delete-organization.

AWS CLI

To delete an organization

The following example shows how to delete an organization. To perform this operation, you must be an admin of the master account in the organization. The example assumes that you previously removed all the member accounts, OUs, and policies from the organization:

```
aws organizations delete-organization
```

- For API details, see [DeleteOrganization](#) in *AWS CLI Command Reference*.

delete-organizational-unit

The following code example shows how to use delete-organizational-unit.

AWS CLI

To delete an OU

The following example shows how to delete an OU. The example assumes that you previously removed all accounts and other OUs from the OU:

```
aws organizations delete-organizational-unit --organizational-unit-id ou-  
examplerootid111-exampleouid111
```

- For API details, see [DeleteOrganizationalUnit](#) in *AWS CLI Command Reference*.

delete-policy

The following code example shows how to use delete-policy.

AWS CLI

To delete a policy

The following example shows how to delete a policy from an organization. The example assumes that you previously detached the policy from all entities:

```
aws organizations delete-policy --policy-id p-examplepolicyid111
```

- For API details, see [DeletePolicy](#) in *AWS CLI Command Reference*.

describe-account

The following code example shows how to use describe-account.

AWS CLI

To get the details about an account

The following example shows you how to request details about an account:

```
aws organizations describe-account --account-id 555555555555
```

The output shows an account object with the details about the account:

```
{
  "Account": {
    "Id": "555555555555",
    "Arn": "arn:aws:organizations::111111111111:account/o-
exampleorgid/555555555555",
    "Name": "Beta account",
    "Email": "anika@example.com",
    "JoinedMethod": "INVITED",
    "JoinedTimeStamp": 1481756563.134,
    "Status": "ACTIVE"
  }
}
```

- For API details, see [DescribeAccount](#) in *AWS CLI Command Reference*.

describe-create-account-status

The following code example shows how to use describe-create-account-status.

AWS CLI

To get the latest status about a request to create an account

The following example shows how to request the latest status for a previous request to create an account in an organization. The specified `--request-id` comes from the response of the original call to `create-account`. The account creation request shows by the `status` field that Organizations successfully completed the creation of the account.

Command:

```
aws organizations describe-create-account-status --create-account-request-id car-examplecreateaccountrequestid111
```

Output:

```
{
  "CreateAccountStatus": {
    "State": "SUCCEEDED",
    "AccountId": "555555555555",
    "AccountName": "Beta account",
    "RequestedTimestamp": 1470684478.687,
    "CompletedTimestamp": 1470684532.472,
    "Id": "car-examplecreateaccountrequestid111"
  }
}
```

- For API details, see [DescribeCreateAccountStatus](#) in *AWS CLI Command Reference*.

describe-handshake

The following code example shows how to use `describe-handshake`.

AWS CLI

To get information about a handshake

The following example shows you how to request details about a handshake. The handshake ID comes either from the original call to `InviteAccountToOrganization`, or from a call to `ListHandshakesForAccount` or `ListHandshakesForOrganization`:

```
aws organizations describe-handshake --handshake-id h-examplehandshakeid111
```

The output includes a handshake object that has all the details about the requested handshake:

```
{
  "Handshake": {
    "Id": "h-examplehandshakeid111",
    "State": "OPEN",
    "Resources": [
      {
        "Type": "ORGANIZATION",
        "Value": "o-exampleorgid",
        "Resources": [
          {
            "Type": "MASTER_EMAIL",
            "Value": "bill@example.com"
          },
          {
            "Type": "MASTER_NAME",
            "Value": "Master Account"
          }
        ]
      },
      {
        "Type": "EMAIL",
        "Value": "anika@example.com"
      }
    ],
    "Parties": [
      {
        "Type": "ORGANIZATION",
        "Id": "o-exampleorgid"
      },
      {
        "Type": "EMAIL",
        "Id": "anika@example.com"
      }
    ],
    "Action": "INVITE",
    "RequestedTimestamp": 1470158698.046,
    "ExpirationTimestamp": 1471454698.046,
    "Arn": "arn:aws:organizations::111111111111:handshake/o-exampleorgid/invite/h-examplehandshakeid111"
  }
}
```

```
}  
}
```

- For API details, see [DescribeHandshake](#) in *AWS CLI Command Reference*.

describe-organization

The following code example shows how to use `describe-organization`.

AWS CLI

To get information about the current organization

The following example shows you how to request details about an organization:

```
aws organizations describe-organization
```

The output includes an organization object that has the details about the organization:

```
{  
  "Organization": {  
    "MasterAccountArn": "arn:aws:organizations::111111111111:account/o-  
exampleorgid/111111111111",  
    "MasterAccountEmail": "bill@example.com",  
    "MasterAccountId": "111111111111",  
    "Id": "o-exampleorgid",  
    "FeatureSet": "ALL",  
    "Arn": "arn:aws:organizations::111111111111:organization/o-  
exampleorgid",  
    "AvailablePolicyTypes": [  
      {  
        "Status": "ENABLED",  
        "Type": "SERVICE_CONTROL_POLICY"  
      }  
    ]  
  }  
}
```

- For API details, see [DescribeOrganization](#) in *AWS CLI Command Reference*.

describe-organizational-unit

The following code example shows how to use `describe-organizational-unit`.

AWS CLI

To get information about an OU

The following `describe-organizational-unit` example requests details about an OU.

```
aws organizations describe-organizational-unit \  
  --organizational-unit-id ou-examplerootid111-exampleoid111
```

Output:

```
{  
  "OrganizationalUnit": {  
    "Name": "Accounting Group",  
    "Arn": "arn:aws:organizations::123456789012:ou/o-exampleorgid/ou-  
examplerootid111-exampleoid111",  
    "Id": "ou-examplerootid111-exampleoid111"  
  }  
}
```

- For API details, see [DescribeOrganizationalUnit](#) in *AWS CLI Command Reference*.

describe-policy

The following code example shows how to use `describe-policy`.

AWS CLI

To get information about a policy

The following example shows how to request information about a policy:

```
aws organizations describe-policy --policy-id p-examplepolicyid111
```

The output includes a policy object that contains details about the policy:

```
{  
  "Policy": {
```

```

        "Content": "{\n  \"Version\": \"2012-10-17\",\n  \"Statement\": [\n    {\n      \"Effect\": \"Allow\",\n      \"Action\": \"*\",\n      \"Resource\": \"*\n    ]\n  }",
        "PolicySummary": {
          "Arn": "arn:aws:organizations::111111111111:policy/o-
exampleorgid/service_control_policy/p-examplepolicyid111",
          "Type": "SERVICE_CONTROL_POLICY",
          "Id": "p-examplepolicyid111",
          "AwsManaged": false,
          "Name": "AllowAllS3Actions",
          "Description": "Enables admins to delegate S3 permissions"
        }
      }
    }
  }
}

```

- For API details, see [DescribePolicy](#) in *AWS CLI Command Reference*.

detach-policy

The following code example shows how to use detach-policy.

AWS CLI

To detach a policy from a root, OU, or account

The following example shows how to detach a policy from an OU:

```
aws organizations detach-policy --target-id ou-examplerootid111-exampleouid111 --
policy-id p-examplepolicyid111
```

- For API details, see [DetachPolicy](#) in *AWS CLI Command Reference*.

disable-policy-type

The following code example shows how to use disable-policy-type.

AWS CLI

To disable a policy type in a root

The following example shows how to disable the service control policy (SCP) policy type in a root:

```
aws organizations disable-policy-type --root-id r-examplerootid111 --policy-type
SERVICE_CONTROL_POLICY
```

The output shows that the PolicyTypes response element no longer includes SERVICE_CONTROL_POLICY:

```
{
  "Root": {
    "PolicyTypes": [],
    "Name": "Root",
    "Id": "r-examplerootid111",
    "Arn": "arn:aws:organizations::111111111111:root/o-exampleorgid/r-
examplerootid111"
  }
}
```

- For API details, see [DisablePolicyType](#) in *AWS CLI Command Reference*.

enable-all-features

The following code example shows how to use enable-all-features.

AWS CLI

To enable all features in an organization

This example shows the administrator asking all the invited accounts in the organization to approve enabled all features in the organization. AWS Organizations sends an email to the address that is registered with every invited member account asking the owner to approve the change to all features by accepting the handshake that is sent. After all invited member accounts accept the handshake, the organization administrator can finalize the change to all features, and those with appropriate permissions can create policies and apply them to roots, OUs, and accounts:

```
aws organizations enable-all-features
```

The output is a handshake object that is sent to all invited member accounts for approval:

```
{
  "Handshake": {
```

```

    "Action": "ENABLE_ALL_FEATURES",
    "Arn": "arn:aws:organizations::111111111111:handshake/o-exampleorgid/
enable_all_features/h-examplehandshakeid111",
    "ExpirationTimestamp": 1.483127868609E9,
    "Id": "h-examplehandshakeid111",
    "Parties": [
      {
        "id": "o-exampleorgid",
        "type": "ORGANIZATION"
      }
    ],
    "requestedTimestamp": 1.481831868609E9,
    "resources": [
      {
        "type": "ORGANIZATION",
        "value": "o-exampleorgid"
      }
    ],
    "state": "REQUESTED"
  }
}

```

- For API details, see [EnableAllFeatures](#) in *AWS CLI Command Reference*.

enable-policy-type

The following code example shows how to use `enable-policy-type`.

AWS CLI

To enable the use of a policy type in a root

The following example shows how to enable the service control policy (SCP) policy type in a root:

```
aws organizations enable-policy-type --root-id r-examplerootid111 --policy-type
SERVICE_CONTROL_POLICY
```

The output shows a root object with a `policyTypes` response element showing that SCPs are now enabled:

```
{
```



```

    "Root": {
      "PolicyTypes": [
        {
          "Status": "ENABLED",
          "Type": "SERVICE_CONTROL_POLICY"
        }
      ],
      "Id": "r-examplerootid111",
      "Name": "Root",
      "Arn": "arn:aws:organizations::111111111111:root/o-exampleorgid/r-examplerootid111"
    }
  }
}

```

- For API details, see [EnablePolicyType](#) in *AWS CLI Command Reference*.

invite-account-to-organization

The following code example shows how to use `invite-account-to-organization`.

AWS CLI

To invite an account to join an organization

The following example shows the master account owned by `bill@example.com` inviting the account owned by `juan@example.com` to join an organization:

```

aws organizations invite-account-to-organization --target '{"Type": "EMAIL", "Id": "juan@example.com"}' --notes "This is a request for Juan's account to join Bill's organization."

```

The output includes a handshake structure that shows what is sent to the invited account:

```

{
  "Handshake": {
    "Action": "INVITE",
    "Arn": "arn:aws:organizations::111111111111:handshake/o-exampleorgid/invite/h-examplehandshakeid111",
    "ExpirationTimestamp": 1482952459.257,
    "Id": "h-examplehandshakeid111",
    "Parties": [

```

```

        {
            "Id": "o-exampleorgid",
            "Type": "ORGANIZATION"
        },
        {
            "Id": "juan@example.com",
            "Type": "EMAIL"
        }
    ],
    "RequestedTimestamp": 1481656459.257,
    "Resources": [
        {
            "Resources": [
                {
                    "Type": "MASTER_EMAIL",
                    "Value": "bill@amazon.com"
                },
                {
                    "Type": "MASTER_NAME",
                    "Value": "Org Master Account"
                },
                {
                    "Type": "ORGANIZATION_FEATURE_SET",
                    "Value": "FULL"
                }
            ],
            "Type": "ORGANIZATION",
            "Value": "o-exampleorgid"
        },
        {
            "Type": "EMAIL",
            "Value": "juan@example.com"
        }
    ],
    "State": "OPEN"
}

```

- For API details, see [InviteAccountToOrganization](#) in *AWS CLI Command Reference*.

leave-organization

The following code example shows how to use leave-organization.

AWS CLI

To leave an organization as a member account

The following example shows the administrator of a member account requesting to leave the organization it is currently a member of:

```
aws organizations leave-organization
```

- For API details, see [LeaveOrganization](#) in *AWS CLI Command Reference*.

list-accounts-for-parent

The following code example shows how to use `list-accounts-for-parent`.

AWS CLI

To retrieve a list of all of the accounts in a specified parent root or OU

The following example shows how to request a list of the accounts in an OU:

```
aws organizations list-accounts-for-parent --parent-id ou-examplerootid111-  
exampleouid111
```

The output includes a list of account summary objects.

```
{  
  "Accounts": [  
    {  
      "Arn": "arn:aws:organizations::111111111111:account/o-  
exampleorgid/333333333333",  
      "JoinedMethod": "INVITED",  
      "JoinedTimestamp": 1481835795.536,  
      "Id": "333333333333",  
      "Name": "Development Account",  
      "Email": "juan@example.com",  
      "Status": "ACTIVE"  
    },  
    {  
      "Arn": "arn:aws:organizations::111111111111:account/o-  
exampleorgid/444444444444",  
      "JoinedMethod": "INVITED",
```

```
        "JoinedTimestamp": 1481835812.143,  
        "Id": "44444444444444",  
        "Name": "Test Account",  
        "Email": "anika@example.com",  
        "Status": "ACTIVE"  
    }  
]  
}
```

- For API details, see [ListAccountsForParent](#) in *AWS CLI Command Reference*.

list-accounts

The following code example shows how to use `list-accounts`.

AWS CLI

To retrieve a list of all of the accounts in an organization

The following example shows you how to request a list of the accounts in an organization:

```
aws organizations list-accounts
```

The output includes a list of account summary objects.

```
{  
  "Accounts": [  
    {  
      "Arn": "arn:aws:organizations::111111111111:account/o-  
exampleorgid/111111111111",  
      "JoinedMethod": "INVITED",  
      "JoinedTimestamp": 1481830215.45,  
      "Id": "111111111111",  
      "Name": "Master Account",  
      "Email": "bill@example.com",  
      "Status": "ACTIVE"  
    },  
    {  
      "Arn": "arn:aws:organizations::111111111111:account/o-  
exampleorgid/222222222222",  
      "JoinedMethod": "INVITED",  
      "JoinedTimestamp": 1481835741.044,  
      "Id": "222222222222",  
    }  
  ]  
}
```

```

        "Name": "Production Account",
        "Email": "alice@example.com",
        "Status": "ACTIVE"
    },
    {
        "Arn": "arn:aws:organizations::111111111111:account/o-
exampleorgid/333333333333",
        "JoinedMethod": "INVITED",
        "JoinedTimestamp": 1481835795.536,
        "Id": "333333333333",
        "Name": "Development Account",
        "Email": "juan@example.com",
        "Status": "ACTIVE"
    },
    {
        "Arn": "arn:aws:organizations::111111111111:account/o-
exampleorgid/444444444444",
        "JoinedMethod": "INVITED",
        "JoinedTimestamp": 1481835812.143,
        "Id": "444444444444",
        "Name": "Test Account",
        "Email": "anika@example.com",
        "Status": "ACTIVE"
    }
]
}

```

- For API details, see [ListAccounts](#) in *AWS CLI Command Reference*.

list-children

The following code example shows how to use `list-children`.

AWS CLI

To retrieve the child accounts and OUs of a parent OU or root

The following example you how to list the root or OU that contains that account 444444444444:

```
aws organizations list-children --child-type ORGANIZATIONAL_UNIT --parent-id ou-
exampleroottid111-exampleoid111
```

The output shows the two child OUs that are contained by the parent:

```
{
  "Children": [
    {
      "Id": "ou-examplerootid111-exampleouid111",
      "Type": "ORGANIZATIONAL_UNIT"
    },
    {
      "Id": "ou-examplerootid111-exampleouid222",
      "Type": "ORGANIZATIONAL_UNIT"
    }
  ]
}
```

- For API details, see [ListChildren](#) in *AWS CLI Command Reference*.

list-create-account-status

The following code example shows how to use `list-create-account-status`.

AWS CLI

Example 1: To retrieve a list of the account creation requests made in the current organization

The following example shows how to request a list of account creation requests for an organization that have completed successfully:

```
aws organizations list-create-account-status --states SUCCEEDED
```

The output includes an array of objects with information about each request.

```
{
  "CreateAccountStatuses": [
    {
      "AccountId": "4444444444444444",
      "AccountName": "Developer Test Account",
      "CompletedTimeStamp": 1481835812.143,
      "Id": "car-examplecreateaccountrequestid111",
      "RequestedTimeStamp": 1481829432.531,

```

```

        "State": "SUCCEEDED"
      }
    ]
  }

```

Example 2: To retrieve a list of the in progress account creation requests made in the current organization

The following example gets a list of in-progress account creation requests for an organization:

```
aws organizations list-create-account-status --states IN_PROGRESS
```

The output includes an array of objects with information about each request.

```

{
  "CreateAccountStatuses": [
    {
      "State": "IN_PROGRESS",
      "Id": "car-examplecreateaccountrequestid111",
      "RequestedTimeStamp": 1481829432.531,
      "AccountName": "Production Account"
    }
  ]
}

```

- For API details, see [ListCreateAccountStatus](#) in *AWS CLI Command Reference*.

list-handshakes-for-account

The following code example shows how to use `list-handshakes-for-account`.

AWS CLI

To retrieve a list of the handshakes sent to an account

The following example shows how to get a list of all handshakes that are associated with the account of the credentials that were used to call the operation:

```
aws organizations list-handshakes-for-account
```

The output includes a list of handshake structures with information about each handshake including its current state:

```
{
  "Handshake": {
    "Action": "INVITE",
    "Arn": "arn:aws:organizations::111111111111:handshake/o-
exampleorgid/invite/h-examplehandshakeid111",
    "ExpirationTimestamp": 1482952459.257,
    "Id": "h-examplehandshakeid111",
    "Parties": [
      {
        "Id": "o-exampleorgid",
        "Type": "ORGANIZATION"
      },
      {
        "Id": "juan@example.com",
        "Type": "EMAIL"
      }
    ],
    "RequestedTimestamp": 1481656459.257,
    "Resources": [
      {
        "Resources": [
          {
            "Type": "MASTER_EMAIL",
            "Value": "bill@amazon.com"
          },
          {
            "Type": "MASTER_NAME",
            "Value": "Org Master Account"
          },
          {
            "Type": "ORGANIZATION_FEATURE_SET",
            "Value": "FULL"
          }
        ],
        "Type": "ORGANIZATION",
        "Value": "o-exampleorgid"
      },
      {
        "Type": "EMAIL",
        "Value": "juan@example.com"
      }
    ]
  }
}
```



```

    }
    ],
    "State": "OPEN"
  }
}

```

- For API details, see [ListHandshakesForAccount](#) in *AWS CLI Command Reference*.

list-handshakes-for-organization

The following code example shows how to use `list-handshakes-for-organization`.

AWS CLI

To retrieve a list of the handshakes associated with an organization

The following example shows how to get a list of handshakes that are associated with the current organization:

```
aws organizations list-handshakes-for-organization
```

The output shows two handshakes. The first one is an invitation to Juan's account and shows a state of OPEN. The second is an invitation to Anika's account and shows a state of ACCEPTED:

```

{
  "Handshakes": [
    {
      "Action": "INVITE",
      "Arn": "arn:aws:organizations::111111111111:handshake/o-
exampleorgid/invite/h-examplehandshakeid111",
      "ExpirationTimestamp": 1482952459.257,
      "Id": "h-examplehandshakeid111",
      "Parties": [
        {
          "Id": "o-exampleorgid",
          "Type": "ORGANIZATION"
        },
        {
          "Id": "juan@example.com",
          "Type": "EMAIL"
        }
      ]
    },
  ],
}

```

```

    "RequestedTimestamp": 1481656459.257,
    "Resources": [
      {
        "Resources": [
          {
            "Type": "MASTER_EMAIL",
            "Value": "bill@amazon.com"
          },
          {
            "Type": "MASTER_NAME",
            "Value": "Org Master"
          },
          {
            "Type": "ORGANIZATION_FEATURE_SET",
            "Value": "FULL"
          }
        ],
        "Type": "ORGANIZATION",
        "Value": "o-exampleorgid"
      },
      {
        "Type": "EMAIL",
        "Value": "juan@example.com"
      },
      {
        "Type": "NOTES",
        "Value": "This is an invitation to Juan's
account to join Bill's organization."
      }
    ],
    "State": "OPEN"
  },
  {
    "Action": "INVITE",
    "State": "ACCEPTED",
    "Arn": "arn:aws:organizations::111111111111:handshake/o-
exampleorgid/invite/h-examplehandshakeid111",
    "ExpirationTimestamp": 1.471797437427E9,
    "Id": "h-examplehandshakeid222",
    "Parties": [
      {
        "Id": "o-exampleorgid",

```

```

        "Type": "ORGANIZATION"
      },
      {
        "Id": "anika@example.com",
        "Type": "EMAIL"
      }
    ],
    "RequestedTimestamp": 1.469205437427E9,
    "Resources": [
      {
        "Resources": [
          {
            "Type": "MASTER_EMAIL",
            "Value": "bill@example.com"
          },
          {
            "Type": "MASTER_NAME",
            "Value": "Master Account"
          }
        ],
        "Type": "ORGANIZATION",
        "Value": "o-exampleorgid"
      },
      {
        "Type": "EMAIL",
        "Value": "anika@example.com"
      },
      {
        "Type": "NOTES",
        "Value": "This is an invitation to Anika's
account to join Bill's organization."
      }
    ]
  }
]
}

```

- For API details, see [ListHandshakesForOrganization](#) in *AWS CLI Command Reference*.

list-organizational-units-for-parent

The following code example shows how to use `list-organizational-units-for-parent`.

AWS CLI

To retrieve a list of the OUs in a parent OU or root

The following example shows you how to get a list of OUs in a specified root:

```
aws organizations list-organizational-units-for-parent --parent-id r-  
examplerootid111
```

The output shows that the specified root contains two OUs and shows details of each:

```
{  
  "OrganizationalUnits": [  
    {  
      "Name": "AccountingDepartment",  
      "Arn": "arn:aws:organizations::o-exampleorgid:ou/r-  
examplerootid111/ou-examplerootid111-exampleouid111"  
    },  
    {  
      "Name": "ProductionDepartment",  
      "Arn": "arn:aws:organizations::o-exampleorgid:ou/r-  
examplerootid111/ou-examplerootid111-exampleouid222"  
    }  
  ]  
}
```

- For API details, see [ListOrganizationalUnitsForParent](#) in *AWS CLI Command Reference*.

list-parents

The following code example shows how to use `list-parents`.

AWS CLI

To list the parent OUs or roots for an account or child OU

The following example you how to list the root or parent OU that contains that account 444444444444:

```
aws organizations list-parents --child-id 444444444444
```

The output shows that the specified account is in the OU with specified ID:

```
{
  "Parents": [
    {
      "Id": "ou-examplerootid111-exampleoid111",
      "Type": "ORGANIZATIONAL_UNIT"
    }
  ]
}
```

- For API details, see [ListParents](#) in *AWS CLI Command Reference*.

list-policies-for-target

The following code example shows how to use `list-policies-for-target`.

AWS CLI

To retrieve a list of the SCPs attached directly to an account

The following example shows how to get a list of all service control policies (SCPs), as specified by the `Filter` parameter, that are directly attached to an account:

```
aws organizations list-policies-for-target --filter SERVICE_CONTROL_POLICY --target-id 44444444444444
```

The output includes a list of policy structures with summary information about the policies. The list does not include policies that apply to the account because of inheritance from its location in an OU hierarchy:

```
{
  "Policies": [
    {
      "Type": "SERVICE_CONTROL_POLICY",
      "Name": "AllowAllEC2Actions",
      "AwsManaged": false,
      "Id": "p-examplepolicyid222",
      "Arn": "arn:aws:organizations::o-exampleorgid:policy/service_control_policy/p-examplepolicyid222",
      "Description": "Enables account admins to delegate permissions for any EC2 actions to users and roles in their accounts."
    }
  ]
}
```

```
]
}
```

- For API details, see [ListPoliciesForTarget](#) in *AWS CLI Command Reference*.

list-policies

The following code example shows how to use `list-policies`.

AWS CLI

To retrieve a list of all policies in an organization of a certain type

The following example shows you how to get a list of SCPs, as specified by the `filter` parameter:

```
aws organizations list-policies --filter SERVICE_CONTROL_POLICY
```

The output includes a list of policies with summary information:

```
{
  "Policies": [
    {
      "Type": "SERVICE_CONTROL_POLICY",
      "Name": "AllowAllS3Actions",
      "AwsManaged": false,
      "Id": "p-examplepolicyid111",
      "Arn": "arn:aws:organizations::111111111111:policy/service_control_policy/p-examplepolicyid111",
      "Description": "Enables account admins to delegate permissions for any S3 actions to users and roles in their accounts."
    },
    {
      "Type": "SERVICE_CONTROL_POLICY",
      "Name": "AllowAllEC2Actions",
      "AwsManaged": false,
      "Id": "p-examplepolicyid222",
      "Arn": "arn:aws:organizations::111111111111:policy/service_control_policy/p-examplepolicyid222",
      "Description": "Enables account admins to delegate permissions for any EC2 actions to users and roles in their accounts."
    },
    {
```

```

        "AwsManaged": true,
        "Description": "Allows access to every operation",
        "Type": "SERVICE_CONTROL_POLICY",
        "Id": "p-FullAWSAccess",
        "Arn": "arn:aws:organizations::aws:policy/
service_control_policy/p-FullAWSAccess",
        "Name": "FullAWSAccess"
    }
]
}

```

- For API details, see [ListPolicies](#) in *AWS CLI Command Reference*.

list-roots

The following code example shows how to use `list-roots`.

AWS CLI

To retrieve a list of the roots in an organization

This example shows you how to get the list of roots for an organization:

```
aws organizations list-roots
```

The output includes a list of root structures with summary information:

```

{
  "Roots": [
    {
      "Name": "Root",
      "Arn": "arn:aws:organizations::111111111111:root/o-
exampleorgid/r-examplerootid111",
      "Id": "r-examplerootid111",
      "PolicyTypes": [
        {
          "Status": "ENABLED",
          "Type": "SERVICE_CONTROL_POLICY"
        }
      ]
    }
  ]
}

```

```
}

```

- For API details, see [ListRoots](#) in *AWS CLI Command Reference*.

list-targets-for-policy

The following code example shows how to use `list-targets-for-policy`.

AWS CLI

To retrieve a list of the roots, OUs, and accounts that a policy is attached to

The following example shows how to get a list of the roots, OUs, and accounts that the specified policy is attached to:

```
aws organizations list-targets-for-policy --policy-id p-FullAWSAccess
```

The output includes a list of attachment objects with summary information about the roots, OUs, and accounts the policy is attached to:

```
{
  "Targets": [
    {
      "Arn": "arn:aws:organizations::111111111111:root/o-
exampleorgid/r-examplerootid111",
      "Name": "Root",
      "TargetId": "r-examplerootid111",
      "Type": "ROOT"
    },
    {
      "Arn": "arn:aws:organizations::111111111111:account/o-
exampleorgid/333333333333;",
      "Name": "Developer Test Account",
      "TargetId": "333333333333",
      "Type": "ACCOUNT"
    },
    {
      "Arn": "arn:aws:organizations::111111111111:ou/o-
exampleorgid/ou-examplerootid111-exampleoid111",
      "Name": "Accounting",
      "TargetId": "ou-examplerootid111-exampleoid111",
      "Type": "ORGANIZATIONAL_UNIT"
    }
  ]
}
```



```
    ]  
  }  
}
```

- For API details, see [ListTargetsForPolicy](#) in *AWS CLI Command Reference*.

move-account

The following code example shows how to use `move-account`.

AWS CLI

To move an account between roots or OUs

The following example shows you how to move the master account in the organization from the root to an OU:

```
aws organizations move-account --account-id 333333333333 --source-parent-id r-  
examplerootid111 --destination-parent-id ou-examplerootid111-exampleouid111
```

- For API details, see [MoveAccount](#) in *AWS CLI Command Reference*.

remove-account-from-organization

The following code example shows how to use `remove-account-from-organization`.

AWS CLI

To remove an account from an organization as the master account

The following example shows you how to remove an account from an organization:

```
aws organizations remove-account-from-organization --account-id 333333333333
```

- For API details, see [RemoveAccountFromOrganization](#) in *AWS CLI Command Reference*.

update-organizational-unit

The following code example shows how to use `update-organizational-unit`.

AWS CLI

To rename an OU

This example shows you how to rename an OU: In this example, the OU is renamed "AccountingOU":

```
aws organizations update-organizational-unit --organizational-unit-id ou-
exampleroottid111-exampleoid111 --name AccountingOU
```

The output shows the new name:

```
{
  "OrganizationalUnit": {
    "Id": "ou-exampleroottid111-exampleoid111"
    "Name": "AccountingOU",
    "Arn": "arn:aws:organizations::111111111111:ou/o-exampleorgid/ou-
exampleroottid111-exampleoid111"
  }
}
```

- For API details, see [UpdateOrganizationalUnit](#) in *AWS CLI Command Reference*.

update-policy

The following code example shows how to use update-policy.

AWS CLI

Example 1: To rename a policy

The following update-policy example renames a policy and gives it a new description.

```
aws organizations update-policy \
  --policy-id p-examplepolicyid111 \
  --name Renamed-Policy \
  --description "This description replaces the original."
```

The output shows the new name and description.

```
{
  "Policy": {
```

```

    "Content": "{\n  \"Version\": \"2012-10-17\", \n  \"Statement\": {\n\n    \"Effect\": \"Allow\", \n    \"Action\": \"ec2:*\", \n    \"Resource\": \"*\"\n  }\n}\n",
    "PolicySummary": {
      "Id": "p-examplepolicyid111",
      "AwsManaged": false,
      "Arn": "arn:aws:organizations::111111111111:policy/o-exampleorgid/
service_control_policy/p-examplepolicyid111",
      "Description": "This description replaces the original.",
      "Name": "Renamed-Policy",
      "Type": "SERVICE_CONTROL_POLICY"
    }
  }
}

```

Example 2: To replace a policy's JSON text content

The following example shows you how to replace the JSON text of the SCP in the previous example with a new JSON policy text string that allows S3 instead of EC2:

```

aws organizations update-policy \
  --policy-id p-examplepolicyid111 \
  --content "{\"Version\": \"2012-10-17\", \"Statement\": { \"Effect\": \"Allow\",
  \"Action\": \"s3:*\", \"Resource\": \"*\" } }"

```

The output shows the new content:

```

{
  "Policy": {
    "Content": "{ \"Version\": \"2012-10-17\", \"Statement\": { \"Effect\":
  \"Allow\", \"Action\": \"s3:*\", \"Resource\": \"*\" } }",
    "PolicySummary": {
      "Arn": "arn:aws:organizations::111111111111:policy/o-exampleorgid/
service_control_policy/p-examplepolicyid111",
      "AwsManaged": false;
      "Description": "This description replaces the original.",
      "Id": "p-examplepolicyid111",
      "Name": "Renamed-Policy",
      "Type": "SERVICE_CONTROL_POLICY"
    }
  }
}

```

- For API details, see [UpdatePolicy](#) in *AWS CLI Command Reference*.

AWS Outposts examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Outposts.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

get-outpost-instance-types

The following code example shows how to use `get-outpost-instance-types`.

AWS CLI

To get the instance types on your Outpost

The following `get-outpost-instance-types` example gets the instance types for the specified Outpost.

```
aws outposts get-outpost-instance-types \
  --outpost-id op-0ab23c4567EXAMPLE
```

Output:

```
{
  "InstanceTypes": [
```

```
{
  "InstanceType": "c5d.large"
},
{
  "InstanceType": "i3en.24xlarge"
},
{
  "InstanceType": "m5d.large"
},
{
  "InstanceType": "r5d.large"
}
],
"OutpostId": "op-0ab23c4567EXAMPLE",
"OutpostArn": "arn:aws:outposts:us-west-2:123456789012:outpost/
op-0ab23c4567EXAMPLE"
}
```

For more information, see [Launch an instance on your Outpost](#) in the *AWS Outposts User Guide*.

- For API details, see [GetOutpostInstanceTypes](#) in *AWS CLI Command Reference*.

get-outpost

The following code example shows how to use get-outpost.

AWS CLI

To get Outpost details

The following get-outpost example displays the details for the specified Outpost.

```
aws outposts get-outpost \
  --outpost-id op-0ab23c4567EXAMPLE
```

Output:

```
{
  "Outpost": {
    "OutpostId": "op-0ab23c4567EXAMPLE",
    "OwnerId": "123456789012",
    "OutpostArn": "arn:aws:outposts:us-west-2:123456789012:outpost/
op-0ab23c4567EXAMPLE",
```

```
    "SiteId": "os-0ab12c3456EXAMPLE",
    "Name": "EXAMPLE",
    "LifecycleStatus": "ACTIVE",
    "AvailabilityZone": "us-west-2a",
    "AvailabilityZoneId": "usw2-az1",
    "Tags": {}
  }
}
```

For more information, see [Working with Outposts](#) in the *AWS Outposts User Guide*.

- For API details, see [GetOutpost](#) in *AWS CLI Command Reference*.

list-outposts

The following code example shows how to use `list-outposts`.

AWS CLI

To list Outposts

The following `list-outposts` example lists the Outposts in your AWS account.

```
aws outposts list-outposts
```

Output:

```
{
  "Outposts": [
    {
      "OutpostId": "op-0ab23c4567EXAMPLE",
      "OwnerId": "123456789012",
      "OutpostArn": "arn:aws:outposts:us-west-2:123456789012:outpost/
op-0ab23c4567EXAMPLE",
      "SiteId": "os-0ab12c3456EXAMPLE",
      "Name": "EXAMPLE",
      "Description": "example",
      "LifecycleStatus": "ACTIVE",
      "AvailabilityZone": "us-west-2a",
      "AvailabilityZoneId": "usw2-az1",
      "Tags": {
        "Name": "EXAMPLE"
      }
    }
  ]
}
```

```
    },
    {
      "OutpostId": "op-4fe3dc21baEXAMPLE",
      "OwnerId": "123456789012",
      "OutpostArn": "arn:aws:outposts:us-west-2:123456789012:outpost/
op-4fe3dc21baEXAMPLE",
      "SiteId": "os-0ab12c3456EXAMPLE",
      "Name": "EXAMPLE2",
      "LifeCycleStatus": "ACTIVE",
      "AvailabilityZone": "us-west-2a",
      "AvailabilityZoneId": "usw2-az1",
      "Tags": {}
    }
  ]
}
```

For more information, see [Working with Outposts](#) in the *AWS Outposts User Guide*.

- For API details, see [ListOutposts](#) in *AWS CLI Command Reference*.

list-sites

The following code example shows how to use `list-sites`.

AWS CLI

To list sites

The following `list-sites` example lists the available Outpost sites in your AWS account.

```
aws outposts list-sites
```

Output:

```
{
  "Sites": [
    {
      "SiteId": "os-0ab12c3456EXAMPLE",
      "AccountId": "123456789012",
      "Name": "EXAMPLE",
      "Description": "example",
      "Tags": {}
    }
  ]
}
```

```
]
}
```

For more information, see [Working with Outposts](#) in the *AWS Outposts User Guide*.

- For API details, see [ListSites](#) in *AWS CLI Command Reference*.

AWS Payment Cryptography examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Payment Cryptography.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-alias

The following code example shows how to use `create-alias`.

AWS CLI

To create an alias for a key

The following `create-alias` example creates an alias for a key.

```
aws payment-cryptography create-alias \  
  --alias-name alias/sampleAlias1 \  
  --key-arn arn:aws:payment-cryptography:us-east-2:123456789012:key/  
kwapwa6qaiif1lw2h
```


Output:

```
{
  "Alias": {
    "AliasName": "alias/sampleAlias1",
    "KeyArn": "arn:aws:payment-cryptography:us-west-2:123456789012:key/
kwapwa6qaiifllw2h"
  }
}
```

For more information, see [About aliases](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [CreateAlias](#) in *AWS CLI Command Reference*.

create-key

The following code example shows how to use `create-key`.

AWS CLI**To create a key**

The following `create-key` example generates a 2KEY TDES key you can use to generate and verify CVV/CVV2 values.

```
aws payment-cryptography create-key \
  --exportable \
  --key-attributes KeyAlgorithm=TDES_2KEY,
KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,
KeyModesOfUse={Generate=true,Verify=true}
```

Output:

```
{
  "Key": {
    "CreateTimestamp": "1686800690",
    "Enabled": true,
    "Exportable": true,
    "KeyArn": "arn:aws:payment-cryptography:us-west-2:123456789012:key/
kwapwa6qaiifllw2h",
    "KeyAttributes": {
      "KeyAlgorithm": "TDES_2KEY",
```

```
    "KeyClass": "SYMMETRIC_KEY",
    "KeyModesOfUse": {
      "Decrypt": false,
      "DeriveKey": false,
      "Encrypt": false,
      "Generate": true,
      "NoRestrictions": false,
      "Sign": false,
      "Unwrap": false,
      "Verify": true,
      "Wrap": false
    },
    "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY"
  },
  "KeyCheckValue": "F2E50F",
  "KeyCheckValueAlgorithm": "ANSI_X9_24",
  "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
  "KeyState": "CREATE_COMPLETE",
  "UsageStartTimestamp": "1686800690"
}
}
```

For more information, see [Generating keys](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [CreateKey](#) in *AWS CLI Command Reference*.

delete-alias

The following code example shows how to use delete-alias.

AWS CLI

To delete an alias

The following delete-alias example deletes an alias. It does not affect the key.

```
aws payment-cryptography delete-alias \
  --alias-name alias/sampleAlias1
```

This command produces no output.

For more information, see [About aliases](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [DeleteAlias](#) in *AWS CLI Command Reference*.

delete-key

The following code example shows how to use delete-key.

AWS CLI

To delete a key

The following delete-key example schedules a key for deletion after 7 days, which is the default waiting period.

```
aws payment-cryptography delete-key \  
  --key-identifier arn:aws:payment-cryptography:us-west-2:123456789012:key/  
  kwapwa6qaiifllw2h
```

Output:

```
{  
  "Key": {  
    "CreateTimestamp": "1686801198",  
    "DeletePendingTimestamp": "1687405998",  
    "Enabled": true,  
    "Exportable": true,  
    "KeyArn": "arn:aws:payment-cryptography:us-west-2:123456789012:key/  
kwapwa6qaiifllw2h",  
    "KeyAttributes": {  
      "KeyAlgorithm": "TDES_2KEY",  
      "KeyClass": "SYMMETRIC_KEY",  
      "KeyModesOfUse": {  
        "Decrypt": false,  
        "DeriveKey": false,  
        "Encrypt": false,  
        "Generate": true,  
        "NoRestrictions": false,  
        "Sign": false,  
        "Unwrap": false,  
        "Verify": true,  
        "Wrap": false  
      },  
      "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY"  
    },  
    "KeyCheckValue": "F2E50F",  
    "KeyCheckValueAlgorithm": "ANSI_X9_24",  
  }  
}
```

```

    "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
    "KeyState": "DELETE_PENDING",
    "UsageStartTimestamp": "1686801190"
  }
}

```

For more information, see [Deleting keys](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [DeleteKey](#) in *AWS CLI Command Reference*.

export-key

The following code example shows how to use `export-key`.

AWS CLI

To export a key

The following `export-key` example exports a key.

```

aws payment-cryptography export-key \
  --export-key-identifier arn:aws:payment-cryptography:us-west-2:123456789012:key/
lco3w6gask7zgu2l \
  --key-material '{"Tr34KeyBlock": { \
    "CertificateAuthorityPublicKeyIdentifier": "arn:aws:payment-cryptography:us-
west-2:123456789012:key/ftobshq7pvioc5fx", \
    "ExportToken": "export-token-cu4lg26ofcziixny", \
    "KeyBlockFormat": "X9_TR34_2012", \
    "WrappingKeyCertificate": file://wrapping-key-certificate.pem }}'

```

Contents of `wrapping-key-certificate.pem`:

```

LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUV2VENDQXFXZ0F3SUJBZ01SQ1ZZS8xMXFUK2svVz1RUDJQOE1V

```

Output:

```

{
  "WrappedKey": {
    "KeyMaterial":
    "308205A106092A864886F70D010702A08205923082058E020101310D300B06096086480165030402013082031F
    "WrappedKeyMaterialFormat": "TR34_KEY_BLOCK"
  }
}

```

```
}  
}
```

For more information, see [Export keys](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [ExportKey](#) in *AWS CLI Command Reference*.

get-alias

The following code example shows how to use `get-alias`.

AWS CLI

To get an alias

The following `get-alias` example returns the ARN of the key associated with the alias.

```
aws payment-cryptography get-alias \  
  --alias-name alias/sampleAlias1
```

Output:

```
{  
  "Alias": {  
    "AliasName": "alias/sampleAlias1",  
    "KeyArn": "arn:aws:payment-cryptography:us-west-2:123456789012:key/  
kwapwa6qaiifllw2h"  
  }  
}
```

For more information, see [About aliases](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [GetAlias](#) in *AWS CLI Command Reference*.

get-key

The following code example shows how to use `get-key`.

AWS CLI

To get the metadata of a key

The following `get-key` example returns the metadata of the key associated with the alias. This operation does not return cryptographic material.

```
aws payment-cryptography get-key \  
  --key-identifier alias/sampleAlias1
```

Output:

```
{  
  "Key": {  
    "CreateTimestamp": "1686800690",  
    "DeletePendingTimestamp": "1687405998",  
    "Enabled": true,  
    "Exportable": true,  
    "KeyArn": "arn:aws:payment-cryptography:us-west-2:123456789012:key/  
kwapwa6qaifllw2h",  
    "KeyAttributes": {  
      "KeyAlgorithm": "TDES_2KEY",  
      "KeyClass": "SYMMETRIC_KEY",  
      "KeyModesOfUse": {  
        "Decrypt": false,  
        "DeriveKey": false,  
        "Encrypt": false,  
        "Generate": true,  
        "NoRestrictions": false,  
        "Sign": false,  
        "Unwrap": false,  
        "Verify": true,  
        "Wrap": false  
      },  
      "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY"  
    },  
    "KeyCheckValue": "F2E50F",  
    "KeyCheckValueAlgorithm": "ANSI_X9_24",  
    "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",  
    "KeyState": "DELETE_PENDING",  
    "UsageStartTimestamp": "1686801190"  
  }  
}
```

For more information, see [Get keys](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [GetKey](#) in *AWS CLI Command Reference*.

get-parameters-for-export

The following code example shows how to use `get-parameters-for-export`.

AWS CLI

To initialize the export process

The following `get-parameters-for-export` example generates a key pair, signs the key, and then returns the certificate and certificate root.

```
aws payment-cryptography get-parameters-for-export \
  --signing-key-algorithm RSA_2048 \
  --key-material-type TR34_KEY_BLOCK
```

Output:

```
{
  "ExportToken": "export-token-ep5cwyzone7oya53",
  "ParametersValidUntilTimestamp": "1687415640",
  "SigningKeyAlgorithm": "RSA_2048",
  "SigningKeyCertificate":

  "MIICiTCCAfICCD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
  VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
  b24xFDASBgNVBA5TC0lBTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWx1ZAd
  BgkqhkiG9w0BCQEWEG5vb25lQGFTYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
  MTIwNDI1MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYD
  VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBA5TC0lBTSBDb25z
  b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWx1ZAdBgkqhkiG9w0BCQEWEG5vb25lQGFT
  YXpvbi5jb20wZGZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
  21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
  rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
  Ibb30hjZncvcQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
  nUHVxYUntneD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
  FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJ10ZxBHjJnyp3780D8uTs7fLvjx79LjStB
  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=",
  "SigningKeyCertificateChain":
  "MIICiTCCAfICCD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
  VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
  b24xFDASBgNVBA5TC0lBTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWx1ZAd
  BgkqhkiG9w0BCQEWEG5vb25lQGFTYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
  MTIwNDI1MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYD
```

```
VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAwTC01BTSBDb25z
b2x1MRIwEAYDVQQDEwLUZXN0Q21sYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGft
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLyGVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb30hjZnczvQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE="
}
```

For more information, see [Export keys](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [GetParametersForExport](#) in *AWS CLI Command Reference*.

get-parameters-for-import

The following code example shows how to use `get-parameters-for-import`.

AWS CLI

To initialize the import process

The following `get-parameters-for-import` example generates a key pair, signs the key, and then returns the certificate and certificate root.

```
aws payment-cryptography get-parameters-for-import \
  --key-material-type TR34_KEY_BLOCK \
  --wrapping-key-algorithm RSA_2048
```

Output:

```
{
  "ImportToken": "import-token-qgmafpa7nt2kfbb",
  "ParametersValidUntilTimestamp": "1687415640",
  "WrappingKeyAlgorithm": "RSA_2048",
  "WrappingKeyCertificate":
  "MIICiTCCAFICCCQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAKGA1UEBhMC
  VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
  b24xFDASBgNVBAwTC01BTSBDb25zb2x1MRIwEAYDVQQDEwLUZXN0Q21sYWMxHzAd
  BgkqhkiG9w0BCQEWEG5vb25lQGftYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
  MTIwNDI0MjA0NTIxWjCBiDELMAKGA1UEBhMCMVVMxCzAJBgNVBAGTAldBMRAwDgYD
  VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAwTC01BTSBDb25z
```



```

b2x1MRIwEAYDVQQDEw1UZsXN0Q21sYWMxHzAdBgkqhkiG9w0BCQEWEG5vb251QGft
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb30hjZncvcQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVvxYUntneD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
FFbjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=",
"WrappingKeyCertificateChain":
"NIICiTCCAfICCD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBIDELMAKGA1UEBhMC
VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBA5TC0lBTSBDb25zb2x1MRIwEAYDVQQDEw1UZsXN0Q21sYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb251QGftYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAKGA1UEBhMCMVVMxCzAJBgNVBAGTAldBMRAwDgYD
VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBA5TC0lBTSBDb25z
b2x1MRIwEAYDVQQDEw1UZsXN0Q21sYWMxHzAdBgkqhkiG9w0BCQEWEG5vb251QGft
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb30hjZncvcQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVvxYUntneD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
FFbjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE="
}

```

For more information, see [Import keys](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [GetParametersForImport](#) in *AWS CLI Command Reference*.

get-public-key-certificate

The following code example shows how to use `get-public-key-certificate`.

AWS CLI

To return the public key

The following `get-public-key-certificate` example returns the public key portion of a key pair.

```

aws payment-cryptography get-public-key-certificate \
  --key-identifier arn:aws:payment-cryptography:us-east-2:123456789012:key/
  kwapwa6qaiifllw2h

```

Output:

```
{
  "KeyCertificate":
  "MIICiTCCAfICCD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMakGA1UEBhMCMC
  VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
  b24xFDASBgNVBA5TC01BTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWx1eHAd
  BgkqhkiG9w0BCQEWEG5vb251QGftYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
  MTIwNDI0MjA0NTIxWjCBiDELMakGA1UEBhMCMCVMxCzAJBgNVBAGTAldBMRAwDgYD
  VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBA5TC01BTSBDb25z
  b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWx1eHAdBgkqhkiG9w0BCQEWEG5vb251QGft
  YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
  21uUSfwfEvySWtC2XADZ4nB+BLyGVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
  rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
  Ibb30hjZncvcQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
  nUHVvXyUntneD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
  FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjStB
  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=",
  "KeyCertificateChain":
  "MIICiTCCAfICCD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMakGA1UEBhMCMC
  VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
  b24xFDASBgNVBA5TC01BTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWx1eHAd
  BgkqhkiG9w0BCQEWEG5vb251QGftYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
  MTIwNDI0MjA0NTIxWjCBiDELMakGA1UEBhMCMCVMxCzAJBgNVBAGTAldBMRAwDgYD
  VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBA5TC01BTSBDb25z
  b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWx1eHAdBgkqhkiG9w0BCQEWEG5vb251QGft
  YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
  21uUSfwfEvySWtC2XADZ4nB+BLyGVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
  rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
  Ibb30hjZncvcQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
  nUHVvXyUntneD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
  FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjStB
  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE="
}
```

For more information, see [Get the public key/certificate associated with a key pair](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [GetPublicKeyCertificate](#) in *AWS CLI Command Reference*.

import-key

The following code example shows how to use `import-key`.

AWS CLI

To import a TR-34 key

The following `import-key` example imports a TR-34 key.

```
aws payment-cryptography import-key \
  --key-material='{ "Tr34KeyBlock": {" \
    CertificateAuthorityPublicKeyIdentifier": "arn:aws:payment-
cryptography:us-west-2:123456789012:key/rmm5wn2q564njinjm", \
    "ImportToken": "import-token-5ott6ho5nts7bbc", \
    "KeyBlockFormat": "X9_TR34_2012", \
    "SigningKeyCertificate": file://signing-key-certificate.pem, \
    "WrappedKeyBlock": file://wrapped-key-block.pem }}}
```

Contents of `signing-key-certificate.pem`:

```
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUV2RENDQXFTZ0F3SUJBZ01RYWVCK25IbE1WZU1PR1ZiNjU1Q2Jz
```

Contents of `wrapped-key-block.pem`:

```
3082059806092A864886F70D010702A082058930820585020101310D300B06096086480165030402013082031606
```

Output:

```
{
  "Key": {
    "CreateTimestamp": "2023-06-09T16:56:27.621000-07:00",
    "Enabled": true,
    "KeyArn": "arn:aws:payment-cryptography:us-west-2:123456789012:key/
bzmvgyx dg3sktwxd",
    "KeyAttributes": {
      "KeyAlgorithm": "TDES_2KEY",
      "KeyClass": "SYMMETRIC_KEY",
      "KeyModesOfUse": {
        "Decrypt": false,
        "DeriveKey": false,
        "Encrypt": false,
        "Generate": true,
        "NoRestrictions": false,
        "Sign": false,
        "Unwrap": false,

```

```
        "Verify": true,  
        "Wrap": false  
    },  
    "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY"  
},  
"KeyCheckValue": "D9B20E",  
"KeyCheckValueAlgorithm": "ANSI_X9_24",  
"KeyOrigin": "EXTERNAL",  
"KeyState": "CREATE_COMPLETE",  
"UsageStartTimestamp": "2023-06-09T16:56:27.621000-07:00"  
}  
}
```

For more information, see [Import keys](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [ImportKey](#) in *AWS CLI Command Reference*.

list-aliases

The following code example shows how to use `list-aliases`.

AWS CLI

To get a list of aliases

The following `list-aliases` example shows all of the aliases in your account in this Region.

```
aws payment-cryptography list-aliases
```

Output:

```
{  
  "Aliases": [  
    {  
      "AliasName": "alias/sampleAlias1",  
      "KeyArn": "arn:aws:payment-cryptography:us-east-2:123456789012:key/  
kwapwa6qaifllw2h"  
    },  
    {  
      "AliasName": "alias/sampleAlias2",  
      "KeyArn": "arn:aws:payment-cryptography:us-east-2:123456789012:key/  
kwapwa6qaifllw2h"  
    }  
  ]  
}
```

```
    }  
  ]  
}
```

For more information, see [About aliases](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [ListAliases](#) in *AWS CLI Command Reference*.

list-keys

The following code example shows how to use `list-keys`.

AWS CLI

To get a list of keys

The following `list-keys` example shows all of the keys in your account in this Region.

```
aws payment-cryptography list-keys
```

Output:

```
{  
  "Keys": [  
    {  
      "CreateTimestamp": "1666506840",  
      "Enabled": false,  
      "Exportable": true,  
      "KeyArn": "arn:aws:payment-cryptography:us-east-2:123456789012:key/  
kwapwa6qaifllw2h",  
      "KeyAttributes": {  
        "KeyAlgorithm": "TDES_3KEY",  
        "KeyClass": "SYMMETRIC_KEY",  
        "KeyModesOfUse": {  
          "Decrypt": true,  
          "DeriveKey": false,  
          "Encrypt": true,  
          "Generate": false,  
          "NoRestrictions": false,  
          "Sign": false,  
          "Unwrap": true,  
          "Verify": false,  
          "Wrap": true  
        }  
      }  
    }  
  ]  
}
```

```

        },
        "KeyUsage": "TR31_P1_PIN_GENERATION_KEY"
    },
    "KeyCheckValue": "369D",
    "KeyCheckValueAlgorithm": "ANSI_X9_24",
    "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
    "KeyState": "CREATE_COMPLETE",
    "UsageStopTimestamp": "1666938840"
}
]
}

```

For more information, see [List keys](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [ListKeys](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To get the list of tags for a key

The following `list-tags-for-resource` example gets the tags for a key.

```

aws payment-cryptography list-tags-for-resource \
  --resource-arn arn:aws:payment-cryptography:us-east-2:123456789012:key/
  kwapwa6qaif1lw2h

```

Output:

```

{
  "Tags": [
    {
      "Key": "BIN",
      "Value": "20151120"
    },
    {
      "Key": "Project",
      "Value": "Production"
    }
  ]
}

```

```
}
```

For more information, see [Managing key tags with API operations](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

restore-key

The following code example shows how to use `restore-key`.

AWS CLI

To restore a key that is scheduled for deletion

The following `restore-key` example cancels the deletion of a key.

```
aws payment-cryptography restore-key \  
  --key-identifier arn:aws:payment-cryptography:us-east-2:123456789012:key/  
  kwapwa6qaif1lw2h
```

Output:

```
{  
  "Key": {  
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:123456789012:key/  
kwapwa6qaif1lw2h",  
    "KeyAttributes": {  
      "KeyUsage": "TR31_V2_VISA_PIN_VERIFICATION_KEY",  
      "KeyClass": "SYMMETRIC_KEY",  
      "KeyAlgorithm": "TDES_3KEY",  
      "KeyModesOfUse": {  
        "Encrypt": false,  
        "Decrypt": false,  
        "Wrap": false,  
        "Unwrap": false,  
        "Generate": true,  
        "Sign": false,  
        "Verify": true,  
        "DeriveKey": false,  
        "NoRestrictions": false  
      }  
    }  
  }  
}
```

```

    },
    "KeyCheckValue": "",
    "KeyCheckValueAlgorithm": "ANSI_X9_24",
    "Enabled": false,
    "Exportable": true,
    "KeyState": "CREATE_COMPLETE",
    "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
    "CreateTimestamp": "1686800690",
    "UsageStopTimestamp": "1687405998"
  }
}

```

For more information, see [Deleting keys](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [RestoreKey](#) in *AWS CLI Command Reference*.

start-key-usage

The following code example shows how to use start-key-usage.

AWS CLI

To enable a key

The following start-key-usage example enables a key to be used.

```

aws payment-cryptography start-key-usage \
  --key-identifier arn:aws:payment-cryptography:us-east-2:123456789012:key/
  kwapwa6qaifllw2h

```

Output:

```

{
  "Key": {
    "CreateTimestamp": "1686800690",
    "Enabled": true,
    "Exportable": true,
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
alsuwxug3pgy6xh",
    "KeyAttributes": {
      "KeyAlgorithm": "TDES_3KEY",
      "KeyClass": "SYMMETRIC_KEY",
      "KeyModesOfUse": {

```



```

        "Decrypt": true,
        "DeriveKey": false,
        "Encrypt": true,
        "Generate": false,
        "NoRestrictions": false,
        "Sign": false,
        "Unwrap": true,
        "Verify": false,
        "Wrap": true
    },
    "KeyUsage": "TR31_P1_PIN_GENERATION_KEY"
},
"KeyCheckValue": "369D",
"KeyCheckValueAlgorithm": "ANSI_X9_24",
"KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
"KeyState": "CREATE_COMPLETE",
"UsageStartTimestamp": "1686800690"
}
}

```

For more information, see [Enabling and disabling keys](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [StartKeyUsage](#) in *AWS CLI Command Reference*.

stop-key-usage

The following code example shows how to use stop-key-usage.

AWS CLI

To disable a key

The following stop-key-usage example disables a key.

```

aws payment-cryptography stop-key-usage \
  --key-identifier arn:aws:payment-cryptography:us-east-2:123456789012:key/
  kwapwa6qaifllw2h

```

Output:

```

{
  "Key": {

```

```

    "CreateTimestamp": "1686800690",
    "Enabled": true,
    "Exportable": true,
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
alsuwfxug3pgy6xh",
    "KeyAttributes": {
      "KeyAlgorithm": "TDES_3KEY",
      "KeyClass": "SYMMETRIC_KEY",
      "KeyModesOfUse": {
        "Decrypt": true,
        "DeriveKey": false,
        "Encrypt": true,
        "Generate": false,
        "NoRestrictions": false,
        "Sign": false,
        "Unwrap": true,
        "Verify": false,
        "Wrap": true
      },
      "KeyUsage": "TR31_P1_PIN_GENERATION_KEY"
    },
    "KeyCheckValue": "369D",
    "KeyCheckValueAlgorithm": "ANSI_X9_24",
    "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
    "KeyState": "CREATE_COMPLETE",
    "UsageStartTimestamp": "1686800690"
  }
}

```

For more information, see [Enabling and disabling keys](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [StopKeyUsage](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To tag a key

The following tag-resource example tags a key.

```
aws payment-cryptography tag-resource \  
  --resource-arn arn:aws:payment-cryptography:us-east-2:123456789012:key/  
kwapwa6qaif1lw2h \  
  --tags Key=sampleTag,Value=sampleValue
```

This command produces no output.

For more information, see [Managing key tags](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag from a key

The following `untag-resource` example removes a tag from a key.

```
aws payment-cryptography untag-resource \  
  --resource-arn arn:aws:payment-cryptography:us-east-2:123456789012:key/  
kwapwa6qaif1lw2h \  
  --tag-keys sampleTag
```

This command produces no output.

For more information, see [Managing key tags](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-alias

The following code example shows how to use `update-alias`.

AWS CLI

To update an alias

The following `update-alias` example associates the alias with a different key.

```
aws payment-cryptography update-alias \  
  --resource-arn arn:aws:payment-cryptography:us-east-2:123456789012:key/  
kwapwa6qaif1lw2h \  
  --alias sampleAlias \  
  --key sampleKey
```

```
--alias-name alias/sampleAlias1 \  
--key-arn arn:aws:payment-cryptography:us-east-2:123456789012:key/  
tqv5yij6wtxx64pi
```

Output:

```
{  
  "Alias": {  
    "AliasName": "alias/sampleAlias1",  
    "KeyArn": "arn:aws:payment-cryptography:us-west-2:123456789012:key/  
tqv5yij6wtxx64pi "  
  }  
}
```

For more information, see [About aliases](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [UpdateAlias](#) in *AWS CLI Command Reference*.

AWS Payment Cryptography Data Plane examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Payment Cryptography Data Plane.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

decrypt-data

The following code example shows how to use `decrypt-data`.

AWS CLI

To decrypt ciphertext

The following `decrypt-data` example decrypts ciphertext data using a symmetric key. For this operation, the key must have `KeyModesOfUse` set to `Decrypt` and `KeyUsage` set to `TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY`.

```
aws payment-cryptography-data decrypt-data \  
  --key-identifier arn:aws:payment-cryptography:us-east-2:123456789012:key/  
kwapwa6qaifllw2h \  
  --cipher-text 33612AB9D6929C3A828EB6030082B2BD \  
  --decryption-attributes 'Symmetric={Mode=CBC}'
```

Output:

```
{  
  "KeyArn": "arn:aws:payment-cryptography:us-east-2:123456789012:key/  
kwapwa6qaifllw2h",  
  "KeyCheckValue": "71D7AE",  
  "PlainText": "31323334313233343132333431323334"  
}
```

For more information, see [Decrypt data](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [DecryptData](#) in *AWS CLI Command Reference*.

encrypt-data

The following code example shows how to use `encrypt-data`.

AWS CLI

To encrypt data

The following `encrypt-data` example encrypts plaintext data using a symmetric key. For this operation, the key must have `KeyModesOfUse` set to `Encrypt` and `KeyUsage` set to `TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY`.

```
aws payment-cryptography-data encrypt-data \  
  --key-identifier arn:aws:payment-cryptography:us-east-2:123456789012:key/  
kwapwa6qaifllw2h \  
  --plaintext 31323334313233343132333431323334
```

```
--plain-text 31323334313233343132333431323334 \  
--encryption-attributes 'Symmetric={Mode=CBC}'
```

Output:

```
{  
  "KeyArn": "arn:aws:payment-cryptography:us-east-2:123456789012:key/  
kwapwa6qaif1lw2h",  
  "KeyCheckValue": "71D7AE",  
  "CipherText": "33612AB9D6929C3A828EB6030082B2BD"  
}
```

For more information, see [Encrypt data](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [EncryptData](#) in *AWS CLI Command Reference*.

generate-card-validation-data

The following code example shows how to use `generate-card-validation-data`.

AWS CLI

To generate a CVV

The following `generate-card-validation-data` example generates a CVV/CVV2.

```
aws payment-cryptography-data generate-card-validation-data \  
  --key-identifier arn:aws:payment-cryptography:us-east-2:123456789012:key/  
kwapwa6qaif1lw2h \  
  --primary-account-number=171234567890123 \  
  --generation-attributes CardVerificationValue2={CardExpiryDate=0123}
```

Output:

```
{  
  "KeyArn": "arn:aws:payment-cryptography:us-east-2:123456789012:key/  
kwapwa6qaif1lw2h",  
  "KeyCheckValue": "CADD1",  
  "ValidationData": "801"  
}
```

For more information, see [Generate card data](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [GenerateCardValidationData](#) in *AWS CLI Command Reference*.

generate-mac

The following code example shows how to use generate-mac.

AWS CLI

To generate a MAC

The following generate-card-validation-data example generates a Hash-Based Message Authentication Code (HMAC) for card data authentication using the algorithm HMAC_SHA256 and an HMAC encryption key. The key must have KeyUsage set to TR31_M7_HMAC_KEY and KeyModesOfUse to Generate.

```
aws payment-cryptography-data generate-mac \  
  --key-identifier arn:aws:payment-cryptography:us-east-2:123456789012:key/  
kwapwa6qaif1lw2h \  
  --message-data  
  "3b313038383439303031303733393431353d32343038323236303030373030303f33" \  
  --generation-attributes Algorithm=HMAC_SHA256
```

Output:

```
{  
  "KeyArn": "arn:aws:payment-cryptography:us-east-2:123456789012:key/  
kwapwa6qaif1lw2h,  
  "KeyCheckValue": "2976E7",  
  "Mac": "ED87F26E961C6D0DDB78DA5038AA2BDDEA0DCE03E5B5E96BDDD494F4A7AA470C"  
}
```

For more information, see [Generate MAC](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [GenerateMac](#) in *AWS CLI Command Reference*.

generate-pin-data

The following code example shows how to use generate-pin-data.

AWS CLI

To generate a PIN

The following `generate-card-validation-data` example generate a new random PIN using the Visa PIN scheme.

```
aws payment-cryptography-data generate-pin-data \  
  --generation-key-identifier arn:aws:payment-cryptography:us-  
east-2:111122223333:key/37y2tsl45p5zjbh2 \  
  --encryption-key-identifier arn:aws:payment-cryptography:us-  
east-2:111122223333:key/ivi5ksfsuplneuyt \  
  --primary-account-number 171234567890123 \  
  --pin-block-format ISO_FORMAT_0 \  
  --generation-attributes VisaPin={PinVerificationKeyIndex=1}
```

Output:

```
{  
  "GenerationKeyArn": "arn:aws:payment-cryptography:us-  
east-2:111122223333:key/37y2tsl45p5zjbh2",  
  "GenerationKeyCheckValue": "7F2363",  
  "EncryptionKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/  
ivi5ksfsuplneuyt",  
  "EncryptionKeyCheckValue": "7CC9E2",  
  "EncryptedPinBlock": "AC17DC148BDA645E",  
  "PinData": {  
    "VerificationValue": "5507"  
  }  
}
```

For more information, see [Generate PIN data](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [GeneratePinData](#) in *AWS CLI Command Reference*.

re-encrypt-data

The following code example shows how to use `re-encrypt-data`.

AWS CLI

To re-encrypt data with a different key

The following `re-encrypt-data` example decrypts cipher text that was encrypted using an AES symmetric key and re-encrypts it using a Derived Unique Key Per Transaction (DUKPT) key.


```
aws payment-cryptography-data re-encrypt-data \
  --incoming-key-identifier arn:aws:payment-cryptography:us-
west-2:111122223333:key/hyv7ymboitd4vfy \
  --outgoing-key-identifier arn:aws:payment-cryptography:us-
west-2:111122223333:key/jl6ythkcvzesbxen \
  --cipher-text
4D2B0BDBA192D5AEFEAA5B3EC28E4A65383C313FFA25140101560F75FE1B99F27192A90980AB9334 \
  --incoming-encryption-attributes
"Dukpt={Mode=ECB,KeySerialNumber=0123456789111111}" \
  --outgoing-encryption-attributes '{"Symmetric": {"Mode": "ECB"}}'
```

Output:

```
{
  "CipherText":
  "F94959DA30EEFF0C035483C6067667CF6796E3C1AD28C2B61F9CFEB772A8DD41C0D6822931E0D3B1",
  "KeyArn": "arn:aws:payment-cryptography:us-west-2:111122223333:key/
jl6ythkcvzesbxen",
  "KeyCheckValue": "2E8CD9"
}
```

For more information, see [Encrypt and decrypt data](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [ReEncryptData](#) in *AWS CLI Command Reference*.

translate-pin-data

The following code example shows how to use `translate-pin-data`.

AWS CLI

To translate PIN data

The following `translate-pin-data` example translates a PIN from PEK TDES encryption using ISO 0 PIN block to an AES ISO 4 PIN Block using the DUKPT algorithm.

```
aws payment-cryptography-data translate-pin-data \
  --encrypted-pin-block "AC17DC148BDA645E" \
  --incoming-translation-
attributes=IsoFormat0='{PrimaryAccountNumber=171234567890123}' \
```

```
--incoming-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt \
--outgoing-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/4pmyquwjs3yj4vwe \
--outgoing-translation-attributes
IsoFormat4="{PrimaryAccountNumber=171234567890123}" \
--outgoing-dukpt-attributes KeySerialNumber="FFFF9876543210E00008"
```

Output:

```
{
  "PinBlock": "1F4209C670E49F83E75CC72E81B787D9",
  "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt
  "KeyCheckValue": "7CC9E2"
}
```

For more information, see [Translate PIN data](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [TranslatePinData](#) in *AWS CLI Command Reference*.

verify-auth-request-cryptogram

The following code example shows how to use `verify-auth-request-cryptogram`.

AWS CLI

To verify an auth request

The following `verify-auth-request-cryptogram` example verifies an Authorization Request Cryptogram (ARQC).

```
aws payment-cryptography-data verify-auth-request-cryptogram \
--auth-request-cryptogram F6E1BD1E6037FB3E \
--auth-response-attributes '{"ArpcMethod1": {"AuthResponseCode": "1111"}}' \
--key-identifier arn:aws:payment-cryptography:us-west-2:111122223333:key/
pboipdfzd4mdklya \
--major-key-derivation-mode "EMV_OPTION_A" \
--session-key-derivation-attributes '{"EmvCommon":
{"ApplicationTransactionCounter": "1234", "PanSequenceNumber":
"01", "PrimaryAccountNumber": "471234567890123"}}' \
--transaction-data "123456789ABCDEF"
```

Output:

```
{
  "AuthResponseValue": "D899B8C6FBF971AA",
  "KeyArn": "arn:aws:payment-cryptography:us-west-2:111122223333:key/
pboipdfzd4mdklya",
  "KeyCheckValue": "985792"
}
```

For more information, see [Verify auth request \(ARQC\) cryptogram](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [VerifyAuthRequestCryptogram](#) in *AWS CLI Command Reference*.

verify-card-validation-data

The following code example shows how to use `verify-card-validation-data`.

AWS CLI**To validate a CVV**

The following `verify-card-validation-data` example validates a CVV/CVV2 for a PAN.

```
aws payment-cryptography-data verify-card-validation-data \
  --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi \
  --primary-account-number=171234567890123 \
  --verification-attributes CardVerificationValue2={CardExpiryDate=0123} \
  --validation-data 801
```

Output:

```
{
  "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
  "KeyCheckValue": "CADD1"
}
```

For more information, see [Verify card data](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [VerifyCardValidationData](#) in *AWS CLI Command Reference*.

verify-mac

The following code example shows how to use `verify-mac`.

AWS CLI

To verify a MAC

The following `verify-mac` example verifies a Hash-Based Message Authentication Code (HMAC) for card data authentication using the algorithm HMAC_SHA256 and an HMAC encryption key.

```
aws payment-cryptography-data verify-mac \  
  --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/  
qno151ghrzunce6 \  
  --message-data  
  "3b343038383439303031303733393431353d32343038323236303030373030303f33" \  
  --verification-attributes='Algorithm=HMAC_SHA256' \  
  --mac ED87F26E961C6D0DDB78DA5038AA2BDDEA0DCE03E5B5E96BDDD494F4A7AA470C
```

Output:

```
{  
  "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/  
qno151ghrzunce6,  
  "KeyCheckValue": "2976E7",  
}
```

For more information, see [Verify MAC](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [VerifyMac](#) in *AWS CLI Command Reference*.

verify-pin-data

The following code example shows how to use `verify-pin-data`.

AWS CLI

To verify a PIN

The following `verify-pin-data` example validates a PIN for a PAN.

```
aws payment-cryptography-data verify-pin-data \  
  --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/  
qno151ghrzunce6 \  
  --message-data  
  "3b343038383439303031303733393431353d32343038323236303030373030303f33" \  
  --verification-attributes='Algorithm=HMAC_SHA256' \  
  --mac ED87F26E961C6D0DDB78DA5038AA2BDDEA0DCE03E5B5E96BDDD494F4A7AA470C
```

```
--verification-key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/37y2tsl45p5zjbh2 \  
--encryption-key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/ivi5ksfsuplneuyt \  
--primary-account-number 171234567890123 \  
--pin-block-format ISO_FORMAT_0 \  
--verification-attributes  
VisaPin="{PinVerificationKeyIndex=1,VerificationValue=5507}" \  
--encrypted-pin-block AC17DC148BDA645E
```

Output:

```
{  
  "VerificationKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/37y2tsl45p5zjbh2",  
  "VerificationKeyCheckValue": "7F2363",  
  "EncryptionKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/ivi5ksfsuplneuyt",  
  "EncryptionKeyCheckValue": "7CC9E2",  
}
```

For more information, see [Verify PIN data](#) in the *AWS Payment Cryptography User Guide*.

- For API details, see [VerifyPinData](#) in *AWS CLI Command Reference*.

Amazon Pinpoint examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon Pinpoint.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-app

The following code example shows how to use create-app.

AWS CLI

Example 1: To create an application

The following create-app example creates a new application (project).

```
aws pinpoint create-app \  
  --create-application-request Name=ExampleCorp
```

Output:

```
{  
  "ApplicationResponse": {  
    "Arn": "arn:aws:mobiletargeting:us-  
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example",  
    "Id": "810c7aab86d42fb2b56c8c966example",  
    "Name": "ExampleCorp",  
    "tags": {}  
  }  
}
```

Example 2: To create an application that is tagged

The following create-app example creates a new application (project) and associates a tag (key and value) with the application.

```
aws pinpoint create-app \  
  --create-application-request Name=ExampleCorp,tags={"Stack"="Test"}
```

Output:

```
{  
  "ApplicationResponse": {  
    "Arn": "arn:aws:mobiletargeting:us-  
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example",  
    "Id": "810c7aab86d42fb2b56c8c966example",
```

```
    "Name": "ExampleCorp",
    "tags": {
      "Stack": "Test"
    }
  }
}
```

- For API details, see [CreateApp](#) in *AWS CLI Command Reference*.

create-sms-template

The following code example shows how to use `create-sms-template`.

AWS CLI

Creates a message template for messages that are sent through the SMS channel

The following `create-sms-template` example creates a SMS message template.

```
aws pinpoint create-sms-template \
  --template-name TestTemplate \
  --sms-template-request file://myfile.json \
  --region us-east-1
```

Contents of `myfile.json`:

```
{
  "Body": "hello\n how are you?\n food is good",
  "TemplateDescription": "Test SMS Template"
}
```

Output:

```
{
  "CreateTemplateMessageBody": {
    "Arn": "arn:aws:mobiletargeting:us-east-1:AIDACKCEVSQ6C2EXAMPLE:templates/
TestTemplate/SMS",
    "Message": "Created",
    "RequestID": "8c36b17f-a0b0-400f-ac21-29e9b62a975d"
  }
}
```

For more information, see [Amazon Pinpoint message templates](#) in the *Amazon Pinpoint User Guide*.

- For API details, see [CreateSmsTemplate](#) in *AWS CLI Command Reference*.

delete-app

The following code example shows how to use `delete-app`.

AWS CLI

To delete an application

The following `delete-app` example deletes an application (project).

```
aws pinpoint delete-app \  
  --application-id 810c7aab86d42fb2b56c8c966example
```

Output:

```
{  
  "ApplicationResponse": {  
    "Arn": "arn:aws:mobiletargeting:us-  
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example",  
    "Id": "810c7aab86d42fb2b56c8c966example",  
    "Name": "ExampleCorp",  
    "tags": {}  
  }  
}
```

- For API details, see [DeleteApp](#) in *AWS CLI Command Reference*.

get-apns-channel

The following code example shows how to use `get-apns-channel`.

AWS CLI

To retrieve information about the status and settings of the APNs channel for an application

The following `get-apns-channel` example retrieves information about the status and settings of the APNs channel for an application.


```
aws pinpoint get-apns-channel \  
  --application-id 9ab1068eb0a6461c86cce7f27ce0efd7 \  
  --region us-east-1
```

Output:

```
{  
  "APNSChannelResponse": {  
    "ApplicationId": "9ab1068eb0a6461c86cce7f27ce0efd7",  
    "CreationDate": "2019-05-09T21:54:45.082Z",  
    "DefaultAuthenticationMethod": "CERTIFICATE",  
    "Enabled": true,  
    "HasCredential": true,  
    "HasTokenKey": false,  
    "Id": "apns",  
    "IsArchived": false,  
    "LastModifiedDate": "2019-05-09T22:04:01.067Z",  
    "Platform": "APNS",  
    "Version": 2  
  }  
}
```

- For API details, see [GetApnsChannel](#) in *AWS CLI Command Reference*.

get-app

The following code example shows how to use `get-app`.

AWS CLI

To retrieve information about an application (project)

The following `get-app` example retrieves information about an application (project).

```
aws pinpoint get-app \  
  --application-id 810c7aab86d42fb2b56c8c966example \  
  --region us-east-1
```

Output:

```
{
```

```
"ApplicationResponse": {
  "Arn": "arn:aws:mobiletargeting:us-
east-1:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example",
  "Id": "810c7aab86d42fb2b56c8c966example",
  "Name": "ExampleCorp",
  "tags": {
    "Year": "2019",
    "Stack": "Production"
  }
}
```

- For API details, see [GetApp](#) in *AWS CLI Command Reference*.

get-apps

The following code example shows how to use `get-apps`.

AWS CLI

To retrieve information about all of your applications

The following `get-apps` example retrieves information about all of your applications (projects).

```
aws pinpoint get-apps
```

Output:

```
{
  "ApplicationsResponse": {
    "Item": [
      {
        "Arn": "arn:aws:mobiletargeting:us-
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example",
        "Id": "810c7aab86d42fb2b56c8c966example",
        "Name": "ExampleCorp",
        "tags": {
          "Year": "2019",
          "Stack": "Production"
        }
      },
      {
```

```

        "Arn": "arn:aws:mobiletargeting:us-
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/42d8c7eb0990a57ba1d5476a3example",
        "Id": "42d8c7eb0990a57ba1d5476a3example",
        "Name": "AnyCompany",
        "tags": {}
    },
    {
        "Arn": "arn:aws:mobiletargeting:us-
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/80f5c382b638ffe5ad12376bbexample",
        "Id": "80f5c382b638ffe5ad12376bbexample",
        "Name": "ExampleCorp_Test",
        "tags": {
            "Year": "2019",
            "Stack": "Test"
        }
    }
],
    "NextToken":
    "eyJJdcmVhdGlvbkRhdGUiOiIyMDE5LTA3LTE2VDE0jM40jUzLjkwM1oiLCJBY2NvdW50SWQ0iI1MTIzOTcxODM4Nz"
}
}

```

The presence of the `NextToken` response value indicates that there is more output available. Call the command again and supply that value as the `NextToken` input parameter.

- For API details, see [GetApps](#) in *AWS CLI Command Reference*.

get-campaign

The following code example shows how to use `get-campaign`.

AWS CLI

To retrieve information about the status, configuration, and other settings for a campaign

The following `get-campaign` example retrieves information about the status, configuration, and other settings for a campaign.

```

aws pinpoint get-campaign \
  --application-id 6e0b7591a90841d2b5d93fa11143e5a7 \
  --campaign-id a1e63c6cc0eb43ed826ffcc3cc90b30d \
  --region us-east-1

```

Output:

```
{
  "CampaignResponse": {
    "AdditionalTreatments": [],
    "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",
    "Arn": "arn:aws:mobiletargeting:us-east-1:AIDACKCEVSQ6C2EXAMPLE:apps/6e0b7591a90841d2b5d93fa11143e5a7/campaigns/a1e63c6cc0eb43ed826ffcc3cc90b30d",
    "CreationDate": "2019-10-08T18:40:16.581Z",
    "Description": " ",
    "HoldoutPercent": 0,
    "Id": "a1e63c6cc0eb43ed826ffcc3cc90b30d",
    "IsPaused": false,
    "LastModifiedDate": "2019-10-08T18:40:16.581Z",
    "Limits": {
      "Daily": 0,
      "MaximumDuration": 60,
      "MessagesPerSecond": 50,
      "Total": 0
    },
    "MessageConfiguration": {
      "EmailMessage": {
        "FromAddress": "sender@example.com",
        "HtmlBody": "<!DOCTYPE html>\n <html lang=\"en\">\n <head>\n <meta http-equiv=\"Content-Type\" content=\"text/html; charset=utf-8\" />\n</head>\n<body>Hello</body>\n</html>",
        "Title": "PinpointDemo"
      }
    },
    "Name": "MyCampaign",
    "Schedule": {
      "IsLocalTime": false,
      "StartTime": "IMMEDIATE",
      "Timezone": "utc"
    },
    "SegmentId": "b66c9e42f71444b2aa2e0ffc1df28f60",
    "SegmentVersion": 1,
    "State": {
      "CampaignStatus": "COMPLETED"
    },
    "tags": {},
    "TemplateConfiguration": {},
    "Version": 1
  }
}
```

```
}  
}
```

- For API details, see [GetCampaign](#) in *AWS CLI Command Reference*.

get-campaigns

The following code example shows how to use `get-campaigns`.

AWS CLI

To retrieves information about the status, configuration, and other settings for all the campaigns that are associated with an application

The following `get-campaigns` example retrieves information about the status, configuration, and other settings for all the campaigns that are associated with an application.

```
aws pinpoint get-campaigns \  
  --application-id 6e0b7591a90841d2b5d93fa11143e5a7 \  
  --region us-east-1
```

Output:

```
{  
  "CampaignsResponse": {  
    "Item": [  
      {  
        "AdditionalTreatments": [],  
        "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",  
        "Arn": "arn:aws:mobiletargeting:us-  
east-1:AIDACKCEVSQ6C2EXAMPLE:apps/6e0b7591a90841d2b5d93fa11143e5a7/  
campaigns/7e1280344c8f4a9aa40a00b006fe44f1",  
        "CreationDate": "2019-10-08T18:40:22.905Z",  
        "Description": " ",  
        "HoldoutPercent": 0,  
        "Id": "7e1280344c8f4a9aa40a00b006fe44f1",  
        "IsPaused": false,  
        "LastModifiedDate": "2019-10-08T18:40:22.905Z",  
        "Limits": {},  
        "MessageConfiguration": {  
          "EmailMessage": {  
            "FromAddress": "sender@example.com",
```

```

        "HtmlBody": "<!DOCTYPE html>\n    <html lang=\"en
\n>\n    <head>\n    <meta http-equiv=\"Content-Type\" content=\"text/html;
charset=utf-8\" />\n</head>\n<body>Hello</body>\n</html>",
        "Title": "PINpointDemo Test"
    }
},
"Name": "MyCampaign1",
"Schedule": {
    "IsLocalTime": false,
    "QuietTime": {},
    "StartTime": "IMMEDIATE",
    "Timezone": "UTC"
},
"SegmentId": "b66c9e42f71444b2aa2e0ffc1df28f60",
"SegmentVersion": 1,
"State": {
    "CampaignStatus": "COMPLETED"
},
"tags": {},
"TemplateConfiguration": {},
"Version": 1
},
{
    "AdditionalTreatments": [],
    "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",
    "Arn": "arn:aws:mobiletargeting:us-
east-1:AIDACKCEVSQ6C2EXAMPLE:apps/6e0b7591a90841d2b5d93fa11143e5a7/campaigns/
a1e63c6cc0eb43ed826ffcc3cc90b30d",
    "CreationDate": "2019-10-08T18:40:16.581Z",
    "Description": " ",
    "HoldoutPercent": 0,
    "Id": "a1e63c6cc0eb43ed826ffcc3cc90b30d",
    "IsPaused": false,
    "LastModifiedDate": "2019-10-08T18:40:16.581Z",
    "Limits": {
        "Daily": 0,
        "MaximumDuration": 60,
        "MessagesPerSecond": 50,
        "Total": 0
    },
    "MessageConfiguration": {
        "EmailMessage": {
            "FromAddress": "sender@example.com",

```

```

        "HtmlBody": "<!DOCTYPE html>\n    <html lang=\"en
\n>\n    <head>\n    <meta http-equiv=\"Content-Type\" content=\"text/html;
charset=utf-8\" />\n</head>\n<body>Demo</body>\n</html>",
        "Title": "PinpointDemo"
    }
},
"Name": "MyCampaign2",
"Schedule": {
    "IsLocalTime": false,
    "StartTime": "IMMEDIATE",
    "Timezone": "utc"
},
"SegmentId": "b66c9e42f71444b2aa2e0ffc1df28f60",
"SegmentVersion": 1,
"State": {
    "CampaignStatus": "COMPLETED"
},
"tags": {},
"TemplateConfiguration": {},
"Version": 1
}
]
}
}
}

```

- For API details, see [GetCampaigns](#) in *AWS CLI Command Reference*.

get-channels

The following code example shows how to use `get-channels`.

AWS CLI

To retrieves information about the history and status of each channel for an application

The following `get-channels` example retrieves information about the history and status of each channel for an application.

```

aws pinpoint get-channels \
  --application-id 6e0b7591a90841d2b5d93fa11143e5a7 \
  --region us-east-1

```

Output:

```
{
  "ChannelsResponse": {
    "Channels": {
      "GCM": {
        "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",
        "CreationDate": "2019-10-08T18:28:23.182Z",
        "Enabled": true,
        "HasCredential": true,
        "Id": "gcm",
        "IsArchived": false,
        "LastModifiedDate": "2019-10-08T18:28:23.182Z",
        "Version": 1
      },
      "SMS": {
        "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",
        "CreationDate": "2019-10-08T18:39:18.511Z",
        "Enabled": true,
        "Id": "sms",
        "IsArchived": false,
        "LastModifiedDate": "2019-10-08T18:39:18.511Z",
        "Version": 1
      },
      "EMAIL": {
        "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",
        "CreationDate": "2019-10-08T18:27:23.990Z",
        "Enabled": true,
        "Id": "email",
        "IsArchived": false,
        "LastModifiedDate": "2019-10-08T18:27:23.990Z",
        "Version": 1
      },
      "IN_APP": {
        "Enabled": true,
        "IsArchived": false,
        "Version": 0
      }
    }
  }
}
```

- For API details, see [GetChannels](#) in *AWS CLI Command Reference*.

get-email-channel

The following code example shows how to use `get-email-channel`.

AWS CLI

To retrieve information about the status and settings of the Email channel for an application

The following `get-email-channel` example retrieves status and settings of the Email channel for an application.

```
aws pinpoint get-email-channel \  
  --application-id 6e0b7591a90841d2b5d93fa11143e5a7 \  
  --region us-east-1
```

Output:

```
{  
  "EmailChannelResponse": {  
    "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",  
    "CreationDate": "2019-10-08T18:27:23.990Z",  
    "Enabled": true,  
    "FromAddress": "sender@example.com",  
    "Id": "email",  
    "Identity": "arn:aws:ses:us-east-1:AIDACKCEVSQ6C2EXAMPLE:identity/  
sender@example.com",  
    "IsArchived": false,  
    "LastModifiedDate": "2019-10-08T18:27:23.990Z",  
    "MessagesPerSecond": 1,  
    "Platform": "EMAIL",  
    "RoleArn": "arn:aws:iam::AIDACKCEVSQ6C2EXAMPLE:role/pinpoint-events",  
    "Version": 1  
  }  
}
```

- For API details, see [GetEmailChannel](#) in *AWS CLI Command Reference*.

get-endpoint

The following code example shows how to use `get-endpoint`.

AWS CLI

To retrieve information about the settings and attributes of a specific endpoint for an application

The following `get-endpoint` example retrieves information about the settings and attributes of a specific endpoint for an application.

```
aws pinpoint get-endpoint \  
  --application-id 611e3e3cdd47474c9c1399a505665b91 \  
  --endpoint-id testendpoint \  
  --region us-east-1
```

Output:

```
{  
  "EndpointResponse": {  
    "Address": "+11234567890",  
    "ApplicationId": "611e3e3cdd47474c9c1399a505665b91",  
    "Attributes": {},  
    "ChannelType": "SMS",  
    "CohortId": "63",  
    "CreationDate": "2019-01-28T23:55:11.534Z",  
    "EffectiveDate": "2021-08-06T00:04:51.763Z",  
    "EndpointStatus": "ACTIVE",  
    "Id": "testendpoint",  
    "Location": {  
      "Country": "USA"  
    },  
    "Metrics": {  
      "SmsDelivered": 1.0  
    },  
    "OptOut": "ALL",  
    "RequestId": "a204b1f2-7e26-48a7-9c80-b49a2143489d",  
    "User": {  
      "UserAttributes": {  
        "Age": [  
          "24"  
        ]  
      },  
      "UserId": "testuser"  
    }  
  }  
}
```

```
}
```

- For API details, see [GetEndpoint](#) in *AWS CLI Command Reference*.

get-gcm-channel

The following code example shows how to use `get-gcm-channel`.

AWS CLI

To retrieve information about the status and settings of the GCM channel for an application

The following `get-gcm-channel` example retrieves information about the status and settings of the GCM channel for an application.

```
aws pinpoint get-gcm-channel \  
  --application-id 6e0b7591a90841d2b5d93fa11143e5a7 \  
  --region us-east-1
```

Output:

```
{  
  "GCMChannelResponse": {  
    "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",  
    "CreationDate": "2019-10-08T18:28:23.182Z",  
    "Enabled": true,  
    "HasCredential": true,  
    "Id": "gcm",  
    "IsArchived": false,  
    "LastModifiedDate": "2019-10-08T18:28:23.182Z",  
    "Platform": "GCM",  
    "Version": 1  
  }  
}
```

- For API details, see [GetGcmChannel](#) in *AWS CLI Command Reference*.

get-sms-channel

The following code example shows how to use `get-sms-channel`.

AWS CLI

To retrieve information about the status and settings of the SMS channel for an application

The following `get-sms-channel` example retrieves status and settings of the sms channel for an application.

```
aws pinpoint get-sms-channel \  
  --application-id 6e0b7591a90841d2b5d93fa11143e5a7 \  
  --region us-east-1
```

Output:

```
{  
  "SMSChannelResponse": {  
    "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",  
    "CreationDate": "2019-10-08T18:39:18.511Z",  
    "Enabled": true,  
    "Id": "sms",  
    "IsArchived": false,  
    "LastModifiedDate": "2019-10-08T18:39:18.511Z",  
    "Platform": "SMS",  
    "PromotionalMessagesPerSecond": 20,  
    "TransactionalMessagesPerSecond": 20,  
    "Version": 1  
  }  
}
```

- For API details, see [GetSmsChannel](#) in *AWS CLI Command Reference*.

get-sms-template

The following code example shows how to use `get-sms-template`.

AWS CLI

Retrieves the content and settings of a message template for messages that are sent through the SMS channel

The following `get-sms-template` example retrieves the content and settings of a SMS message template.

```
aws pinpoint get-sms-template \  
  --template-name TestTemplate \  
  --region us-east-1
```

Output:

```
{  
  "SMSTemplateResponse": {  
    "Arn": "arn:aws:mobiletargeting:us-east-1:AIDACKCEVSQ6C2EXAMPLE:templates/  
TestTemplate/SMS",  
    "Body": "hello\n how are you?\n food is good",  
    "CreationDate": "2023-06-20T21:37:30.124Z",  
    "LastModifiedDate": "2023-06-20T21:37:30.124Z",  
    "tags": {},  
    "TemplateDescription": "Test SMS Template",  
    "TemplateName": "TestTemplate",  
    "TemplateType": "SMS",  
    "Version": "1"  
  }  
}
```

For more information, see [Amazon Pinpoint message templates](#) in the *Amazon Pinpoint User Guide*.

- For API details, see [GetSmsTemplate](#) in *AWS CLI Command Reference*.

get-voice-channel

The following code example shows how to use `get-voice-channel`.

AWS CLI**To retrieve information about the status and settings of the voice channel for an application**

The following `get-voice-channel` example retrieves status and settings of the voice channel for an application.

```
aws pinpoint get-voice-channel \  
  --application-id 6e0b7591a90841d2b5d93fa11143e5a7 \  
  --region us-east-1
```

Output:

```
{
  "VoiceChannelResponse": {
    "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",
    "CreationDate": "2022-04-28T00:17:03.836Z",
    "Enabled": true,
    "Id": "voice",
    "IsArchived": false,
    "LastModifiedDate": "2022-04-28T00:17:03.836Z",
    "Platform": "VOICE",
    "Version": 1
  }
}
```

- For API details, see [GetVoiceChannel](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To retrieve a list of tags for a resource

The following `list-tags-for-resource` example retrieves all the tags (key names and values) that are associated with the specified resource.

```
aws pinpoint list-tags-for-resource \
  --resource-arn arn:aws:mobiletargeting:us-
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example
```

Output:

```
{
  "TagsModel": {
    "tags": {
      "Year": "2019",
      "Stack": "Production"
    }
  }
}
```

For more information, see 'Tagging Amazon Pinpoint Resources <<https://docs.aws.amazon.com/pinpoint/latest/developerguide/tagging-resources.html>>'__ in the *Amazon Pinpoint Developer Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

phone-number-validate

The following code example shows how to use `phone-number-validate`.

AWS CLI

Retrieves information about a phone number

The following `phone-number-validate` retrieves information about a phone number.

```
aws pinpoint phone-number-validate \  
  --number-validate-request PhoneNumber="+12065550142" \  
  --region us-east-1
```

Output:

```
{  
  "NumberValidateResponse": {  
    "Carrier": "ExampleCorp Mobile",  
    "City": "Seattle",  
    "CleansedPhoneNumberE164": "+12065550142",  
    "CleansedPhoneNumberNational": "2065550142",  
    "Country": "United States",  
    "CountryCodeIso2": "US",  
    "CountryCodeNumeric": "1",  
    "OriginalPhoneNumber": "+12065550142",  
    "PhoneType": "MOBILE",  
    "PhoneTypeCode": 0,  
    "Timezone": "America/Los_Angeles",  
    "ZipCode": "98101"  
  }  
}
```

For more information, see [Amazon Pinpoint SMS channel](#) in the *Amazon Pinpoint User Guide*.

- For API details, see [PhoneNumberValidate](#) in *AWS CLI Command Reference*.

send-messages

The following code example shows how to use send-messages.

AWS CLI

To send SMS message using the endpoint of an application

The following send-messages example sends a direct message for an application with an endpoint.

```
aws pinpoint send-messages \  
  --application-id 611e3e3cdd47474c9c1399a505665b91 \  
  --message-request file://myfile.json \  
  --region us-west-2
```

Contents of myfile.json:

```
{  
  "MessageConfiguration": {  
    "SMSMessage": {  
      "Body": "hello, how are you?"  
    }  
  },  
  "Endpoints": {  
    "testendpoint": {}  
  }  
}
```

Output:

```
{  
  "MessageResponse": {  
    "ApplicationId": "611e3e3cdd47474c9c1399a505665b91",  
    "EndpointResult": {  
      "testendpoint": {  
        "Address": "+12345678900",  
        "DeliveryStatus": "SUCCESSFUL",  
        "MessageId": "itnuqhai5alf1n6ahv3udc05n7hhddr6gb31q6g0",  
        "StatusCode": 200,  
        "StatusMessage": "MessageId:  
itnuqhai5alf1n6ahv3udc05n7hhddr6gb31q6g0"      }  
    }  
  }  
}
```



```
    }
  },
  "RequestId": "c7e23264-04b2-4a46-b800-d24923f74753"
}
}
```

For more information, see [Amazon Pinpoint SMS channel](#) in the *Amazon Pinpoint User Guide*.

- For API details, see [SendMessages](#) in *AWS CLI Command Reference*.

send-users-messages

The following code example shows how to use `send-users-messages`.

AWS CLI

To send SMS message for an user of an application

The following `send-users-messages` example sends a direct message for an user of an application.

```
aws pinpoint send-users-messages \
  --application-id 611e3e3cdd47474c9c1399a505665b91 \
  --send-users-message-request file://myfile.json \
  --region us-west-2
```

Contents of `myfile.json`:

```
{
  "MessageConfiguration": {
    "SMSMessage": {
      "Body": "hello, how are you?"
    }
  },
  "Users": {
    "testuser": {}
  }
}
```

Output:

```
{
```

```

"SendUsersMessageResponse": {
  "ApplicationId": "611e3e3cdd47474c9c1399a505665b91",
  "RequestId": "e0b12cf5-2359-11e9-bb0b-d5fb91876b25",
  "Result": {
    "testuser": {
      "testuserendpoint": {
        "DeliveryStatus": "SUCCESSFUL",
        "MessageId": "7qu4hk5bqhda3i7i2n4pjf98qcu8b7p45ifsmo0",
        "StatusCode": 200,
        "StatusMessage": "MessageId:
7qu4hk5bqhda3i7i2n4pjf98qcu8b7p45ifsmo0",
        "Address": "+12345678900"
      }
    }
  }
}

```

For more information, see [Amazon Pinpoint SMS channel](#) in the *Amazon Pinpoint User Guide*.

- For API details, see [SendUsersMessages](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To add tags to a resource

The following example adds two tags (key names and values) to a resource.

```

aws pinpoint list-tags-for-resource \
  --resource-arn arn:aws:mobiletargeting:us-
east-1:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example \
  --tags-model tags={Stack=Production,Year=2019}

```

This command produces no output.

For more information, see 'Tagging Amazon Pinpoint Resources <<https://docs.aws.amazon.com/pinpoint/latest/developerguide/tagging-resources.html>>'__ in the *Amazon Pinpoint Developer Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

Example 1: To remove a tag from a resource

The following `untag-resource` example removes the specified tag (key name and value) from a resource.

```
aws pinpoint untag-resource \  
  --resource-arn arn:aws:mobiletargeting:us-  
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example \  
  --tag-keys Year
```

This command produces no output.

Example 2: To remove multiple tags from a resource

The following `untag-resource` example removes the specified tags (key names and values) from a resource.

```
aws pinpoint untag-resource \  
  --resource-arn arn:aws:mobiletargeting:us-  
east-1:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example \  
  --tag-keys Year Stack
```

This command produces no output.

For more information, see 'Tagging Amazon Pinpoint Resources <<https://docs.aws.amazon.com/pinpoint/latest/developerguide/tagging-resources.html>>'__ in the *Amazon Pinpoint Developer Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-sms-channel

The following code example shows how to use `update-sms-channel`.

AWS CLI

To enable SMS channel or to update the status and settings of the SMS channel for an application.

The following `update-sms-channel` example enables SMS channel for an SMS channel for an application.

```
aws pinpoint update-sms-channel \  
  --application-id 611e3e3cdd47474c9c1399a505665b91 \  
  --sms-channel-request Enabled=true \  
  --region us-west-2
```

Output:

```
{  
  "SMSChannelResponse": {  
    "ApplicationId": "611e3e3cdd47474c9c1399a505665b91",  
    "CreationDate": "2019-01-28T23:25:25.224Z",  
    "Enabled": true,  
    "Id": "sms",  
    "IsArchived": false,  
    "LastModifiedDate": "2023-05-18T23:22:50.977Z",  
    "Platform": "SMS",  
    "PromotionalMessagesPerSecond": 20,  
    "TransactionalMessagesPerSecond": 20,  
    "Version": 3  
  }  
}
```

For more information, see [Amazon Pinpoint SMS channel](#) in the *Amazon Pinpoint User Guide*.

- For API details, see [UpdateSmsChannel](#) in *AWS CLI Command Reference*.

Amazon Polly examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon Polly.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

delete-lexicon

The following code example shows how to use `delete-lexicon`.

AWS CLI

To delete a lexicon

The following `delete-lexicon` example deletes the specified lexicon.

```
aws polly delete-lexicon \  
  --name w3c
```

This command produces no output.

For more information, see [Using the DeleteLexicon operation](#) in the *Amazon Polly Developer Guide*.

- For API details, see [DeleteLexicon](#) in *AWS CLI Command Reference*.

get-lexicon

The following code example shows how to use `get-lexicon`.

AWS CLI

To retrieve the content of a lexicon

The following `get-lexicon` example retrieves the content of the specified pronunciation lexicon.

```
aws polly get-lexicon \
  --name w3c
```

Output:

```
{
  "Lexicon": {
    "Content": "<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n<lexicon version=
\n\"1.0\" \n      xmlns=      \"http://www.w3.org/2005/01/pronunciation-lexicon
\n\" \n      xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\" \n
xsi:schemaLocation=\"http://www.w3.org/2005/01/pronunciation-lexicon \n
http://www.w3.org/TR/2007/CR-pronunciation-lexicon-20071212/pls.xsd\" \n
  alphabet=\"ipa\" \n      xml:lang=\"en-US\">\n  <lexeme>\n    <grapheme>W3C</
grapheme>\n      <alias>World Wide Web Consortium</alias>\n  </lexeme>\n</lexicon>
\n",
    "Name": "w3c"
  },
  "LexiconAttributes": {
    "Alphabet": "ipa",
    "LanguageCode": "en-US",
    "LastModified": 1603908910.99,
    "LexiconArn": "arn:aws:polly:us-west-2:880185128111:lexicon/w3c",
    "LexemesCount": 1,
    "Size": 492
  }
}
```

For more information, see [Using the GetLexicon operation](#) in the *Amazon Polly Developer Guide*.

- For API details, see [GetLexicon](#) in *AWS CLI Command Reference*.

get-speech-synthesis-task

The following code example shows how to use `get-speech-synthesis-task`.

AWS CLI

To get information about a speech synthesis task

The following `get-speech-synthesis-task` example retrieves information about the specified speech synthesis task.

```
aws polly get-speech-synthesis-task \  
  --task-id 70b61c0f-57ce-4715-a247-cae8729dcce9
```

Output:

```
{  
  "SynthesisTask": {  
    "TaskId": "70b61c0f-57ce-4715-a247-cae8729dcce9",  
    "TaskStatus": "completed",  
    "OutputUri": "https://s3.us-west-2.amazonaws.com/my-s3-  
bucket/70b61c0f-57ce-4715-a247-cae8729dcce9.mp3",  
    "CreationTime": 1603911042.689,  
    "RequestCharacters": 1311,  
    "OutputFormat": "mp3",  
    "TextType": "text",  
    "VoiceId": "Joanna"  
  }  
}
```

For more information, see [Creating long audio files](#) in the *Amazon Polly Developer Guide*.

- For API details, see [GetSpeechSynthesisTask](#) in *AWS CLI Command Reference*.

list-lexicons

The following code example shows how to use `list-lexicons`.

AWS CLI

To list your lexicons

The following `list-lexicons` example lists your pronunciation lexicons.

```
aws polly list-lexicons
```

Output:

```
{  
  "Lexicons": [  
    {  
      "Name": "w3c",
```

```

        "Attributes": {
            "Alphabet": "ipa",
            "LanguageCode": "en-US",
            "LastModified": 1603908910.99,
            "LexiconArn": "arn:aws:polly:us-east-2:123456789012:lexicon/w3c",
            "LexemesCount": 1,
            "Size": 492
        }
    ]
}

```

For more information, see [Using the ListLexicons operation](#) in the *Amazon Polly Developer Guide*.

- For API details, see [ListLexicons](#) in *AWS CLI Command Reference*.

list-speech-synthesis-tasks

The following code example shows how to use `list-speech-synthesis-tasks`.

AWS CLI

To list your speech synthesis tasks

The following `list-speech-synthesis-tasks` example lists your speech synthesis tasks.

```
aws polly list-speech-synthesis-tasks
```

Output:

```

{
  "SynthesisTasks": [
    {
      "TaskId": "70b61c0f-57ce-4715-a247-cae8729dcce9",
      "TaskStatus": "completed",
      "OutputUri": "https://s3.us-west-2.amazonaws.com/my-s3-bucket/70b61c0f-57ce-4715-a247-cae8729dcce9.mp3",
      "CreationTime": 1603911042.689,
      "RequestCharacters": 1311,
      "OutputFormat": "mp3",
      "TextType": "text",
    }
  ]
}

```



```

        "VoiceId": "Joanna"
      }
    ]
  }

```

For more information, see [Creating long audio files](#) in the *Amazon Polly Developer Guide*.

- For API details, see [ListSpeechSynthesisTasks](#) in *AWS CLI Command Reference*.

put-lexicon

The following code example shows how to use `put-lexicon`.

AWS CLI

To store a lexicon

The following `put-lexicon` example stores the specified pronunciation lexicon. The example `.pls` file specifies a W3C PLS-compliant lexicon.

```

aws polly put-lexicon \
  --name w3c \
  --content file://example.pls

```

Contents of `example.pls`

```

{
  <?xml version="1.0" encoding="UTF-8"?>
  <lexicon version="1.0"
    xmlns="http://www.w3.org/2005/01/pronunciation-lexicon"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.w3.org/2005/01/pronunciation-lexicon
      http://www.w3.org/TR/2007/CR-pronunciation-lexicon-20071212/pls.xsd"
    alphabet="ipa"
    xml:lang="en-US">
    <lexeme>
      <grapheme>W3C</grapheme>
      <alias>World Wide Web Consortium</alias>
    </lexeme>
  </lexicon>
}

```

This command produces no output.

For more information, see [Using the PutLexicon operation](#) in the *Amazon Polly Developer Guide*.

- For API details, see [PutLexicon](#) in *AWS CLI Command Reference*.

start-speech-synthesis-task

The following code example shows how to use `start-speech-synthesis-task`.

AWS CLI

To synthesize text

The following `start-speech-synthesis-task` example synthesizes the text in `text_file.txt` and stores the resulting MP3 file in the specified bucket.

```
aws polly start-speech-synthesis-task \  
  --output-format mp3 \  
  --output-s3-bucket-name my-s3-bucket \  
  --text file://text_file.txt \  
  --voice-id Joanna
```

Output:

```
{  
  "SynthesisTask": {  
    "TaskId": "70b61c0f-57ce-4715-a247-cae8729dcce9",  
    "TaskStatus": "scheduled",  
    "OutputUri": "https://s3.us-east-2.amazonaws.com/my-s3-  
bucket/70b61c0f-57ce-4715-a247-cae8729dcce9.mp3",  
    "CreationTime": 1603911042.689,  
    "RequestCharacters": 1311,  
    "OutputFormat": "mp3",  
    "TextType": "text",  
    "VoiceId": "Joanna"  
  }  
}
```

For more information, see [Creating long audio files](#) in the *Amazon Polly Developer Guide*.

- For API details, see [StartSpeechSynthesisTask](#) in *AWS CLI Command Reference*.

AWS Price List examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Price List.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

describe-services

The following code example shows how to use `describe-services`.

AWS CLI

To retrieve service metadata

This example retrieves the metadata for the Amazon EC2 service code.

Command:

```
aws pricing describe-services --service-code AmazonEC2 --format-version aws_v1 --max-items 1
```

Output:

```
{
  "Services": [
    {
      "ServiceCode": "AmazonEC2",
      "AttributeNames": [
```

```
"volumeType",
"maxIopsVolume",
"instance",
"instanceCapacity10xlarge",
"locationType",
"instanceFamily",
"operatingSystem",
"clockSpeed",
"LeaseContractLength",
"ecu",
"networkPerformance",
"instanceCapacity8xlarge",
"group",
"maxThroughputVolume",
"gpuMemory",
"ebsOptimized",
"elasticGpuType",
"maxVolumeSize",
"gpu",
"processorFeatures",
"intelAvxAvailable",
"instanceCapacity4xlarge",
"servicecode",
"groupDescription",
"processorArchitecture",
"physicalCores",
"productFamily",
"enhancedNetworkingSupported",
"intelTurboAvailable",
"memory",
"dedicatedEbsThroughput",
"vcpu",
"OfferingClass",
"instanceCapacityLarge",
"capacitystatus",
"termType",
"storage",
"intelAvx2Available",
"storageMedia",
"physicalProcessor",
"provisioned",
"servicename",
"PurchaseOption",
"instanceCapacity18xlarge",
```

```

        "instanceType",
        "tenancy",
        "usagetype",
        "normalizationSizeFactor",
        "instanceCapacity2xlarge",
        "instanceCapacity16xlarge",
        "maxIopsBurstPerformance",
        "instanceCapacity12xlarge",
        "instanceCapacity32xlarge",
        "instanceCapacityXlarge",
        "licenseModel",
        "currentGeneration",
        "preInstalledSw",
        "location",
        "instanceCapacity24xlarge",
        "instanceCapacity9xlarge",
        "instanceCapacityMedium",
        "operation"
    ]
}
],
"FormatVersion": "aws_v1"
}

```

- For API details, see [DescribeServices](#) in *AWS CLI Command Reference*.

get-attribute-values

The following code example shows how to use `get-attribute-values`.

AWS CLI

To retrieve a list of attribute values

The following `get-attribute-values` example retrieves a list of values available for the given attribute.

```

aws pricing get-attribute-values \
  --service-code AmazonEC2 \
  --attribute-name volumeType \
  --max-items 2

```

Output:

```
{
  "NextToken": "eyJ0ZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAyfQ==",
  "AttributeValues": [
    {
      "Value": "Cold HDD"
    },
    {
      "Value": "General Purpose"
    }
  ]
}
```

- For API details, see [GetAttributeValues](#) in *AWS CLI Command Reference*.

get-products

The following code example shows how to use get-products.

AWS CLI

To retrieve a list of products

This example retrieves a list of products that match the given criteria.

Command:

```
aws pricing get-products --filters file:///filters.json --format-version aws_v1 --
max-results 1 --service-code AmazonEC2
```

filters.json:

```
[
  {
    "Type": "TERM_MATCH",
    "Field": "ServiceCode",
    "Value": "AmazonEC2"
  },
  {
    "Type": "TERM_MATCH",
    "Field": "volumeType",
    "Value": "Provisioned IOPS"
  }
]
```

]

Output:

```
{
  "FormatVersion": "aws_v1",
  "NextToken": "WGDY7ko8fQXdlaUZVdasFQ==:RVSagyIFn770XQ0zdUIc09BY6ucBG9itXAZGZF/
zioUz0sUKh6PCcPWa0yPZRiMePb986TeoKYB9155fw/
CyoMq5ymnGmT1Vj39T1jbbAlhcqnVfTmPIilx8Uy5bdDaBYy/e/20fw9Edzsykbs8LTBUbNbiDQ
+BBds5yeI9AQkUepruKk3aEahFPxJ55kx/zk",
  "PriceList": [
    {
      "\product\":{\productFamily\":\Storage\", \attributes\":{\storageMedia\":
      \SSD-backed\", \maxThroughputvolume\":\320 MB/sec\", \volumeType\":\Provisioned
      IOPS\", \maxIopsvolume\":\20000\", \servicecode\":\AmazonEC2\", \usagetype
      \\":\APS1-EBS:VolumeUsage.piops\", \locationType\":\AWS Region\", \location\":
      \Asia Pacific (Singapore)\", \servicename\":\Amazon Elastic Compute Cloud\",
      \maxVolumeSize\":\16 TiB\", \operation\":\\", \sku\":\3MKHN58N7RDDVGKJ\",
      \serviceCode\":\AmazonEC2\", \terms\":{\OnDemand\":{\3MKHN58N7RDDVGKJ.JRTCKXETXF
      \\":{\priceDimensions\":{\3MKHN58N7RDDVGKJ.JRTCKXETXF.6YS6EN2CT7\":{\unit\":\GB-
      Mo\", \endRange\":\Inf\", \description\":\$0.138 per GB-month of Provisioned
      IOPS SSD (io1) provisioned storage - Asia Pacific (Singapore)\", \appliesTo
      \":[], \rateCode\":\3MKHN58N7RDDVGKJ.JRTCKXETXF.6YS6EN2CT7\", \beginRange\":
      \0\", \pricePerUnit\":{\USD\":\0.1380000000\"}}}, \sku\":\3MKHN58N7RDDVGKJ
      \\", \effectiveDate\":\2018-08-01T00:00:00Z\", \offerTermCode\":\JRTCKXETXF
      \\", \termAttributes\":{}}}}, \version\":\20180808005701\", \publicationDate\":
      \2018-08-08T00:57:01Z\"}
    ]
  }
}
```

- For API details, see [GetProducts](#) in *AWS CLI Command Reference*.

AWS Private CA examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Private CA.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-certificate-authority-audit-report

The following code example shows how to use `create-certificate-authority-audit-report`.

AWS CLI

To create a certificate authority audit report

The following `create-certificate-authority-audit-report` command creates an audit report for the private CA identified by the ARN.

```
aws acm-pca create-certificate-authority-audit-report --certificate-  
authority-arn arn:aws:acm-pca:us-east-1:accountid:certificate-  
authority/12345678-1234-1234-1234-123456789012 --s3-bucket-name your-bucket-name --  
audit-report-response-format JSON
```

- For API details, see [CreateCertificateAuthorityAuditReport](#) in *AWS CLI Command Reference*.

create-certificate-authority

The following code example shows how to use `create-certificate-authority`.

AWS CLI

To create a private certificate authority

The following `create-certificate-authority` command creates a private certificate authority in your AWS account.

```
aws acm-pca create-certificate-authority --certificate-authority-configuration  
file://C:\ca_config.txt --revocation-configuration file://C:\revoke_config.txt --  
certificate-authority-type "SUBORDINATE" --idempotency-token 98256344
```


- For API details, see [CreateCertificateAuthority](#) in *AWS CLI Command Reference*.

delete-certificate-authority

The following code example shows how to use `delete-certificate-authority`.

AWS CLI

To delete a private certificate authority

The following `delete-certificate-authority` command deletes the certificate authority identified by the ARN.

```
aws acm-pca delete-certificate-authority --certificate-
authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-
authority/12345678-1234-1234-1234-123456789012
```

- For API details, see [DeleteCertificateAuthority](#) in *AWS CLI Command Reference*.

describe-certificate-authority-audit-report

The following code example shows how to use `describe-certificate-authority-audit-report`.

AWS CLI

To describe an audit report for a certificate authority

The following `describe-certificate-authority-audit-report` command lists information about the specified audit report for the CA identified by the ARN.

```
aws acm-pca describe-certificate-authority-audit-report --certificate-
authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-
authority/99999999-8888-7777-6666-555555555555 --audit-report-id
11111111-2222-3333-4444-555555555555
```

- For API details, see [DescribeCertificateAuthorityAuditReport](#) in *AWS CLI Command Reference*.

describe-certificate-authority

The following code example shows how to use `describe-certificate-authority`.

AWS CLI

To describe a private certificate authority

The following `describe-certificate-authority` command lists information about the private CA identified by the ARN.

```
aws acm-pca describe-certificate-authority --certificate-  
authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-  
authority/12345678-1234-1234-1234-123456789012
```

- For API details, see [DescribeCertificateAuthority](#) in *AWS CLI Command Reference*.

get-certificate-authority-certificate

The following code example shows how to use `get-certificate-authority-certificate`.

AWS CLI

To retrieve a certificate authority (CA) certificate

The following `get-certificate-authority-certificate` command retrieves the certificate and certificate chain for the private CA specified by the ARN.

```
aws acm-pca get-certificate-authority-certificate --certificate-  
authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-  
authority/12345678-1234-1234-1234-123456789012 --output text
```

- For API details, see [GetCertificateAuthorityCertificate](#) in *AWS CLI Command Reference*.

get-certificate-authority-csr

The following code example shows how to use `get-certificate-authority-csr`.

AWS CLI

To retrieve the certificate signing request for a certificate authority

The following `get-certificate-authority-csr` command retrieves the CSR for the private CA specified by the ARN.

```
aws acm-pca get-certificate-authority-csr --certificate-
authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-
authority/12345678-1234-1234-1234-123456789012 --output text
```

- For API details, see [GetCertificateAuthorityCsr](#) in *AWS CLI Command Reference*.

get-certificate

The following code example shows how to use `get-certificate`.

AWS CLI

To retrieve an issued certificate

The following `get-certificate` example retrieves a certificate from the specified private CA.

```
aws acm-pca get-certificate \
  --certificate-authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-
authority/12345678-1234-1234-1234-123456789012 \
  --certificate-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-
authority/12345678-1234-1234-1234-123456789012/
certificate/6707447683a9b7f4055627ffd55cebcc \
  --output text
```

Output:

```
-----BEGIN CERTIFICATE-----
MIIEDzCCAvegAwIBAgIRAJuJ8f6ZVYL7gG/rS3qvrZMwDQYJKoZIhvcNAQELBQAw
cTElMAkGA1UEBhMCVVMxEzARBgNVBAgMC1dhc2hpbmd0b24xEDA0BgNVBACMB1Nl
...certificate body truncated for brevity...
tKCSglgZZrd4FdLw1EkGm+UVXnodwMtJEQyy3oTfZjURPIyyaqskTu/KSS7YDjK0
KQNY73D6Ltmd0EbAyyq10XiDxqY41lvKHJ1eZrPaBmYNABxU=
-----END CERTIFICATE----- -----BEGIN CERTIFICATE-----
MIIDrzCCApegAwIBAgIRA0skdzLvcj1eShkoyEE693AwDQYJKoZIhvcNAQELBQAw
cTElMAkGA1UEBhMCVVMxEzARBgNVBAgMC1dhc2hpbmd0b24xEDA0BgNVBACMB1Nl
...certificate body truncated for brevity...
kdRGB6P2hpxstD0UIwAoCbhoawWfA4ybJznf+j0QhAziNlRdKQRR8n0DWpKt7H9w
dJ5nxsTk/fniJz86Ddtp6n8s82wYdkN3cVffeK72A9aTCOU=
-----END CERTIFICATE-----
```

The first part of the output is the certificate itself. The second part is the certificate chain that chains to the root CA certificate. Note that when you use the `--output text` option, a TAB

character is inserted between the two certificate pieces (that is the cause of the indented text). If you intend to take this output and parse the certificates with other tools, you might need to remove the TAB character so it is processed correctly.

- For API details, see [GetCertificate](#) in *AWS CLI Command Reference*.

import-certificate-authority-certificate

The following code example shows how to use `import-certificate-authority-certificate`.

AWS CLI

To import your certificate authority certificate into ACM PCA

The following `import-certificate-authority-certificate` command imports the signed private CA certificate for the CA specified by the ARN into ACM PCA.

```
aws acm-pca import-certificate-authority-certificate --certificate-authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-authority/12345678-1234-1234-1234-123456789012 --certificate file://C:\ca_cert.pem --certificate-chain file://C:\ca_cert_chain.pem
```

- For API details, see [ImportCertificateAuthorityCertificate](#) in *AWS CLI Command Reference*.

issue-certificate

The following code example shows how to use `issue-certificate`.

AWS CLI

To issue a private certificate

The following `issue-certificate` command uses the private CA specified by the ARN to issue a private certificate.

```
aws acm-pca issue-certificate --certificate-authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-authority/12345678-1234-1234-1234-123456789012 --csr file://C:\cert_1.csr --signing-algorithm "SHA256WITHRSA" --validity Value=365,Type="DAYS" --idempotency-token 1234
```

- For API details, see [IssueCertificate](#) in *AWS CLI Command Reference*.

list-certificate-authorities

The following code example shows how to use `list-certificate-authorities`.

AWS CLI

To list your private certificate authorities

The following `list-certificate-authorities` command lists information about all of the private CAs in your account.

```
aws acm-pca list-certificate-authorities --max-results 10
```

- For API details, see [ListCertificateAuthorities](#) in *AWS CLI Command Reference*.

list-tags

The following code example shows how to use `list-tags`.

AWS CLI

To list the tags for your certificate authority

The following `list-tags` command lists the tags associated with the private CA specified by the ARN.

```
aws acm-pca list-tags --certificate-authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-authority/123455678-1234-1234-1234-123456789012 --max-results 10
```

- For API details, see [ListTags](#) in *AWS CLI Command Reference*.

revoke-certificate

The following code example shows how to use `revoke-certificate`.

AWS CLI

To revoke a private certificate

The following `revoke-certificate` command revokes a private certificate from the CA identified by the ARN.

```
aws acm-pca revoke-certificate --certificate-authority-arn arn:aws:acm-pca:us-west-2:1234567890:certificate-authority/12345678-1234-1234-1234-123456789012 --certificate-serial 67:07:44:76:83:a9:b7:f4:05:56:27:ff:d5:5c:eb:cc --revocation-reason "KEY_COMPROMISE"
```

- For API details, see [RevokeCertificate](#) in *AWS CLI Command Reference*.

tag-certificate-authority

The following code example shows how to use tag-certificate-authority.

AWS CLI

To attach tags to a private certificate authority

The following tag-certificate-authority command attaches one or more tags to your private CA.

```
aws acm-pca tag-certificate-authority --certificate-authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-authority/12345678-1234-1234-1234-123456789012 --tags Key=Admin,Value=Alice
```

- For API details, see [TagCertificateAuthority](#) in *AWS CLI Command Reference*.

untag-certificate-authority

The following code example shows how to use untag-certificate-authority.

AWS CLI

To remove one or more tags from your private certificate authority

The following untag-certificate-authority command removes tags from the private CA identified by the ARN.

```
aws acm-pca untag-certificate-authority --certificate-authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-authority/12345678-1234-1234-1234-123456789012 --tags Key=Purpose,Value=Website
```

- For API details, see [UntagCertificateAuthority](#) in *AWS CLI Command Reference*.

update-certificate-authority

The following code example shows how to use `update-certificate-authority`.

AWS CLI

To update the configuration of your private certificate authority

The following `update-certificate-authority` command updates the status and configuration of the private CA identified by the ARN.

```
aws acm-pca update-certificate-authority --certificate-  
authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-  
authority/12345678-1234-1234-1234-1232456789012 --revocation-configuration file://C:  
\revoke_config.txt --status "DISABLED"
```

- For API details, see [UpdateCertificateAuthority](#) in *AWS CLI Command Reference*.

AWS Proton examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Proton.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

cancel-service-instance-deployment

The following code example shows how to use `cancel-service-instance-deployment`.

AWS CLI

To cancel a service instance deployment

The following `cancel-service-instance-deployment` example cancels a service instance deployment.

```
aws proton cancel-service-instance-deployment \  
  --service-instance-name "instance-one" \  
  --service-name "simple-svc"
```

Output:

```
{  
  "serviceInstance": {  
    "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/service-  
instance/instance-one",  
    "createdAt": "2021-04-02T21:29:59.962000+00:00",  
    "deploymentStatus": "CANCELLING",  
    "environmentName": "simple-env",  
    "lastDeploymentAttemptedAt": "2021-04-02T21:45:15.406000+00:00",  
    "lastDeploymentSucceededAt": "2021-04-02T21:38:00.823000+00:00",  
    "name": "instance-one",  
    "serviceName": "simple-svc",  
    "spec": "proton: ServiceSpec\npipeline:\n  
my_sample_pipeline_optional_input: abc\n my_sample_pipeline_required_input:  
'123'\ninstances:\n- name: my-instance\n environment: MySimpleEnv  
\n spec:\n  my_sample_service_instance_optional_input: def\n  my_sample_service_instance_required_input: '456'\n- name: my-other-instance\n environment: MySimpleEnv\n spec:\n  my_sample_service_instance_required_input:  
'789'\n",  
    "templateMajorVersion": "1",  
    "templateMinorVersion": "1",  
    "templateName": "svc-simple"  
  }  
}
```

For more information, see [Update a service instance](#) in the *The AWS Proton Administrator Guide* or [Update a service instance](#) in the *The AWS Proton User Guide*.

- For API details, see [CancelServiceInstanceDeployment](#) in *AWS CLI Command Reference*.

cancel-service-pipeline-deployment

The following code example shows how to use `cancel-service-pipeline-deployment`.

AWS CLI

To cancel a service pipeline deployment

The following `cancel-service-pipeline-deployment` example cancels a service pipeline deployment.

```
aws proton cancel-service-pipeline-deployment \  
  --service-name "simple-svc"
```

Output:

```
{  
  "pipeline": {  
    "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/pipeline",  
    "createdAt": "2021-04-02T21:29:59.962000+00:00",  
    "deploymentStatus": "CANCELLING",  
    "lastDeploymentAttemptedAt": "2021-04-02T22:02:45.095000+00:00",  
    "lastDeploymentSucceededAt": "2021-04-02T21:39:28.991000+00:00",  
    "templateMajorVersion": "1",  
    "templateMinorVersion": "1",  
    "templateName": "svc-simple"  
  }  
}
```

For more information, see [Update a service pipeline](#) in the *The AWS Proton Administrator Guide* or [Update a service pipeline](#) in the *The AWS Proton User Guide*.

- For API details, see [CancelServicePipelineDeployment](#) in *AWS CLI Command Reference*.

create-service

The following code example shows how to use `create-service`.

AWS CLI

To create a service

The following `create-service` example creates a service with a service pipeline.

```
aws proton create-service \  
  --name "MySimpleService" \  
  --template-name "fargate-service" \  
  --template-major-version "1" \  
  --branch-name "mainline" \  
  --repository-connection-arn "arn:aws:codestar-connections:region-id:account-  
id:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111" \  
  --repository-id "myorg/myapp" \  
  --spec file://spec.yaml
```

Contents of `spec.yaml`:

```
proton: ServiceSpec  
  
pipeline:  
  my_sample_pipeline_required_input: "hello"  
  my_sample_pipeline_optional_input: "bye"  
  
instances:  
  - name: "acme-network-dev"  
    environment: "ENV_NAME"  
    spec:  
      my_sample_service_instance_required_input: "hi"  
      my_sample_service_instance_optional_input: "ho"
```

Output:

```
{  
  "service": {  
    "arn": "arn:aws:proton:region-id:123456789012:service/MySimpleService",  
    "createdAt": "2020-11-18T19:50:27.460000+00:00",  
    "lastModifiedAt": "2020-11-18T19:50:27.460000+00:00",  
    "name": "MySimpleService",  
    "repositoryConnectionArn": "arn:aws:codestar-connections:region-  
id:123456789012connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
    "repositoryId": "myorg/myapp",  
    "status": "CREATE_IN_PROGRESS",  
    "templateName": "fargate-service"  
  }  
}
```

For more information, see [Create a service](#) in the *The AWS Proton Administrator Guide* and [Create a service](#) in the *The AWS Proton User Guide*.

- For API details, see [CreateService](#) in *AWS CLI Command Reference*.

delete-service

The following code example shows how to use `delete-service`.

AWS CLI

To delete a service

The following `delete-service` example deletes a service.

```
aws proton delete-service \  
  --name "simple-svc"
```

Output:

```
{  
  "service": {  
    "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc",  
    "branchName": "mainline",  
    "createdAt": "2020-11-28T22:40:50.512000+00:00",  
    "description": "Edit by updating description",  
    "lastModifiedAt": "2020-11-29T00:30:39.248000+00:00",  
    "name": "simple-svc",  
    "repositoryConnectionArn": "arn:aws:codestar-connections:region-  
id:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
    "repositoryId": "myorg/myapp",  
    "status": "DELETE_IN_PROGRESS",  
    "templateName": "fargate-service"  
  }  
}
```

For more information, see [Delete a service](#) in the *The AWS Proton Administrator Guide*.

- For API details, see [DeleteService](#) in *AWS CLI Command Reference*.

get-service-instance

The following code example shows how to use `get-service-instance`.

AWS CLI

To get service instance details

The following `get-service-instance` example gets detail data for a service instance.

```
aws proton get-service-instance \  
  --name "instance-one" \  
  --service-name "simple-svc"
```

Output:

```
{  
  "serviceInstance": {  
    "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/service-  
instance/instance-one",  
    "createdAt": "2020-11-28T22:40:50.512000+00:00",  
    "deploymentStatus": "SUCCEEDED",  
    "environmentName": "simple-env",  
    "lastDeploymentAttemptedAt": "2020-11-28T22:40:50.512000+00:00",  
    "lastDeploymentSucceededAt": "2020-11-28T22:40:50.512000+00:00",  
    "name": "instance-one",  
    "serviceName": "simple-svc",  
    "spec": "proton: ServiceSpec\npipeline:\nmy_sample_pipeline_optional_input: hello world\nmy_sample_pipeline_required_input: pipeline up\ninstances:\n- name: instance-one\nenvironment: my-simple-env\n spec:\n  my_sample_service_instance_optional_input:  
Ola\n  my_sample_service_instance_required_input: Ciao",  
    "templateMajorVersion": "1",  
    "templateMinorVersion": "0",  
    "templateName": "svc-simple"  
  }  
}
```

For more information, see [View service data](#) in the *The AWS Proton Administrator Guide* or [View service data](#) in the *The AWS Proton User Guide*.

- For API details, see [GetServiceInstance](#) in *AWS CLI Command Reference*.

get-service

The following code example shows how to use `get-service`.

AWS CLI

To get service details

The following `get-service` example gets detail data for a service.

```
aws proton get-service \
  --name "simple-svc"
```

Output:

```
{
  "service": {
    "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc",
    "branchName": "mainline",
    "createdAt": "2020-11-28T22:40:50.512000+00:00",
    "lastModifiedAt": "2020-11-28T22:44:51.207000+00:00",
    "name": "simple-svc",
    "pipeline": {
      "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/
pipeline/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "createdAt": "2020-11-28T22:40:50.512000+00:00",
      "deploymentStatus": "SUCCEEDED",
      "lastDeploymentAttemptedAt": "2020-11-28T22:40:50.512000+00:00",
      "lastDeploymentSucceededAt": "2020-11-28T22:40:50.512000+00:00",
      "spec": "proton: ServiceSpec\npipeline:\n
my_sample_pipeline_required_input: hello\n my_sample_pipeline_optional_input:
bye\ninstances:\n- name: instance-svc-simple\n environment: my-simple-
env\n spec:\n my_sample_service_instance_required_input: hi\n
my_sample_service_instance_optional_input: ho\n",
      "templateMajorVersion": "1",
      "templateMinorVersion": "1",
      "templateName": "svc-simple"
    },
    "repositoryConnectionArn": "arn:aws:codestar-connections:region-
id:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "repositoryId": "myorg/myapp",
    "spec": "proton: ServiceSpec\npipeline:\n
my_sample_pipeline_required_input: hello\n my_sample_pipeline_optional_input:
bye\ninstances:\n- name: instance-svc-simple\n environment: my-simple-
env\n spec:\n my_sample_service_instance_required_input: hi\n
my_sample_service_instance_optional_input: ho\n",
    "status": "ACTIVE",
```

```
    "templateName": "svc-simple"
  }
}
```

For more information, see [View service data](#) in the *The AWS Proton Administrator Guide* or [View service data](#) in the *The AWS Proton User Guide*.

- For API details, see [GetService](#) in *AWS CLI Command Reference*.

list-service-instances

The following code example shows how to use `list-service-instances`.

AWS CLI

Example 1: To list all service instances

The following `list-service-instances` example lists service instances.

```
aws proton list-service-instances
```

Output:

```
{
  "serviceInstances": [
    {
      "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/
service-instance/instance-one",
      "createdAt": "2020-11-28T22:40:50.512000+00:00",
      "deploymentStatus": "SUCCEEDED",
      "environmentArn": "arn:aws:proton:region-id:123456789012:environment/
simple-env",
      "lastDeploymentAttemptedAt": "2020-11-28T22:40:50.512000+00:00",
      "lastDeploymentSucceededAt": "2020-11-28T22:40:50.512000+00:00",
      "name": "instance-one",
      "serviceName": "simple-svc",
      "templateMajorVersion": "1",
      "templateMinorVersion": "0",
      "templateName": "fargate-service"
    }
  ]
}
```

For more information, see [View service instance data](#) in the *The AWS Proton Administrator Guide* or [View service instance data](#) in the *The AWS Proton User Guide*.

Example 2: To list the specified service instance

The following `get-service-instance` example gets a service instance.

```
aws proton get-service-instance \  
  --name "instance-one" \  
  --service-name "simple-svc"
```

Output:

```
{  
  "serviceInstance": {  
    "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/service-  
instance/instance-one",  
    "createdAt": "2020-11-28T22:40:50.512000+00:00",  
    "deploymentStatus": "SUCCEEDED",  
    "environmentName": "simple-env",  
    "lastDeploymentAttemptedAt": "2020-11-28T22:40:50.512000+00:00",  
    "lastDeploymentSucceededAt": "2020-11-28T22:40:50.512000+00:00",  
    "name": "instance-one",  
    "serviceName": "simple-svc",  
    "spec": "proton: ServiceSpec\npipeline:\nmy_sample_pipeline_optional_input: hello world\nmy_sample_pipeline_required_input: pipeline up\ninstances:\n- name: instance-one\nenvironment: my-simple-env\nspec:\n  my_sample_service_instance_optional_input:  
Ola\n  my_sample_service_instance_required_input: Ciao\n",  
    "templateMajorVersion": "1",  
    "templateMinorVersion": "0",  
    "templateName": "svc-simple"  
  }  
}
```

For more information, see [View service instance data](#) in the *The AWS Proton Administrator Guide* or [View service instance data](#) in the *The AWS Proton User Guide*.

- For API details, see [ListServiceInstances](#) in *AWS CLI Command Reference*.

update-service-instance

The following code example shows how to use `update-service-instance`.

AWS CLI

To update a service instance to a new minor version

The following `update-service-instance` example updates a service instance to a new minor version of its service template that adds a new instance named "my-other-instance" with a new required input.

```
aws proton update-service-instance \
  --service-name "simple-svc" \
  --spec "file://service-spec.yaml " \
  --template-major-version "1" \
  --template-minor-version "1" \
  --deployment-type "MINOR_VERSION" \
  --name "instance-one"
```

Contents of `service-spec.yaml`:

```
proton: ServiceSpec
pipeline:
  my_sample_pipeline_optional_input: "abc"
  my_sample_pipeline_required_input: "123"
instances:
  - name: "instance-one"
    environment: "simple-env"
    spec:
      my_sample_service_instance_optional_input: "def"
      my_sample_service_instance_required_input: "456"
  - name: "my-other-instance"
    environment: "simple-env"
    spec:
      my_sample_service_instance_required_input: "789"
```

Output:

```
{
  "serviceInstance": {
    "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/service-instance/instance-one",
    "createdAt": "2021-04-02T21:29:59.962000+00:00",
    "deploymentStatus": "IN_PROGRESS",
    "environmentName": "arn:aws:proton:region-id:123456789012:environment/simple-env",
```



```
    "lastDeploymentAttemptedAt": "2021-04-02T21:38:00.823000+00:00",
    "lastDeploymentSucceededAt": "2021-04-02T21:29:59.962000+00:00",
    "name": "instance-one",
    "serviceName": "simple-svc",
    "templateMajorVersion": "1",
    "templateMinorVersion": "0",
    "templateName": "svc-simple"
  }
}
```

For more information, see [Update a service instance](#) in the *The AWS Proton Administrator Guide* or [Update a service instance](#) in the *The AWS Proton User Guide*.

- For API details, see [UpdateServiceInstance](#) in *AWS CLI Command Reference*.

update-service-pipeline

The following code example shows how to use `update-service-pipeline`.

AWS CLI

To update a service pipeline

The following `update-service-pipeline` example updates a service pipeline to a new minor version of its service template.

```
aws proton update-service-pipeline \
  --service-name "simple-svc" \
  --spec "file://service-spec.yaml" \
  --template-major-version "1" \
  --template-minor-version "1" \
  --deployment-type "MINOR_VERSION"
```

Output:

```
{
  "pipeline": {
    "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/pipeline/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "createdAt": "2021-04-02T21:29:59.962000+00:00",
    "deploymentStatus": "IN_PROGRESS",
    "lastDeploymentAttemptedAt": "2021-04-02T21:39:28.991000+00:00",
```

```

        "lastDeploymentSucceededAt": "2021-04-02T21:29:59.962000+00:00",
        "spec": "proton: ServiceSpec\n\npipeline:\n
my_sample_pipeline_optional_input: \"abc\"\n my_sample_pipeline_required_input:
\"123\"\n\ninstances:\n - name: \"my-instance\"\n environment: \"MySimpleEnv
\n\n spec:\n my_sample_service_instance_optional_input: \"def
\n\n my_sample_service_instance_required_input: \"456\"\n - name:
\"my-other-instance\"\n environment: \"MySimpleEnv\"\n spec:\n
my_sample_service_instance_required_input: \"789\"\n",
        "templateMajorVersion": "1",
        "templateMinorVersion": "0",
        "templateName": "svc-simple"
    }
}

```

For more information, see [Update a service pipeline](#) in the *The AWS Proton Administrator Guide* or [Update a service pipeline](#) in the *The AWS Proton User Guide*.

- For API details, see [UpdateServicePipeline](#) in *AWS CLI Command Reference*.

update-service

The following code example shows how to use `update-service`.

AWS CLI

To update a service

The following `update-service` example edits a service description.

```

aws proton update-service \
  --name "MySimpleService" \
  --description "Edit by updating description"

```

Output:

```

{
  "service": {
    "arn": "arn:aws:proton:region-id:123456789012:service/MySimpleService",
    "branchName": "mainline",
    "createdAt": "2021-03-12T22:39:42.318000+00:00",
    "description": "Edit by updating description",
    "lastModifiedAt": "2021-03-12T22:44:21.975000+00:00",
  }
}

```

```
    "name": "MySimpleService",
    "repositoryConnectionArn": "arn:aws:codestar-connections:region-
id:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "repositoryId": "myorg/myapp",
    "status": "ACTIVE",
    "templateName": "fargate-service"
  }
}
```

For more information, see [Edit a service](#) in the *The AWS Proton Administrator Guide* or [Edit a service](#) in the *The AWS Proton User Guide*.

- For API details, see [UpdateService](#) in *AWS CLI Command Reference*.

QLDB examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with QLDB.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

cancel-journal-kinesis-stream

The following code example shows how to use `cancel-journal-kinesis-stream`.

AWS CLI

To cancel a journal stream

The following `cancel-journal-kinesis-stream` example cancels the specified journal stream from a ledger.

```
aws qlldb cancel-journal-kinesis-stream \  
  --ledger-name myExampleLedger \  
  --stream-id 7ISCKqwe4y25YyHLzYUFAf
```

Output:

```
{  
  "StreamId": "7ISCKqwe4y25YyHLzYUFAf"  
}
```

For more information, see [Streaming journal data from Amazon QLDB](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [CancelJournalKinesisStream](#) in *AWS CLI Command Reference*.

create-ledger

The following code example shows how to use `create-ledger`.

AWS CLI

Example 1: To create a ledger with default properties

The following `create-ledger` example creates a ledger with the name `myExampleLedger` and the permissions mode `STANDARD`. The optional parameters for deletion protection and AWS KMS key are not specified, so they default to `true` and an AWS owned KMS key respectively.

```
aws qlldb create-ledger \  
  --name myExampleLedger \  
  --permissions-mode STANDARD
```

Output:

```
{  
  "State": "CREATING",  
  "Arn": "arn:aws:qlldb:us-west-2:123456789012:ledger/myExampleLedger",  
  "DeletionProtection": true,
```

```
"CreationDateTime": 1568839243.951,  
"Name": "myExampleLedger",  
"PermissionsMode": "STANDARD"  
}
```

Example 2: To create a ledger with deletion protection disabled, a customer managed KMS key, and specified tags

The following `create-ledger` example creates a ledger with the name `myExampleLedger2` and the permissions mode `STANDARD`. The deletion protection feature is disabled, the specified customer managed KMS key is used for encryption at rest, and the specified tags are attached to the resource.

```
aws qlldb create-ledger \  
  --name myExampleLedger2 \  
  --permissions-mode STANDARD \  
  --no-deletion-protection \  
  --kms-key arn:aws:kms:us-west-2:123456789012:key/a1b2c3d4-5678-90ab-cdef-  
EXAMPLE11111 \  
  --tags IsTest=true,Domain=Test
```

Output:

```
{  
  "Arn": "arn:aws:qlldb:us-west-2:123456789012:ledger/myExampleLedger2",  
  "DeletionProtection": false,  
  "CreationDateTime": 1568839543.557,  
  "State": "CREATING",  
  "Name": "myExampleLedger2",  
  "PermissionsMode": "STANDARD",  
  "KmsKeyArn": "arn:aws:kms:us-west-2:123456789012:key/a1b2c3d4-5678-90ab-cdef-  
EXAMPLE11111"  
}
```

For more information, see [Basic Operations for Amazon QLDB Ledgers](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [CreateLedger](#) in *AWS CLI Command Reference*.

delete-ledger

The following code example shows how to use `delete-ledger`.

AWS CLI

To delete a ledger

The following `delete-ledger` example deletes the specified ledger.

```
aws qlldb delete-ledger \  
  --name myExampleLedger
```

This command produces no output.

For more information, see [Basic Operations for Amazon QLDB Ledgers](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [DeleteLedger](#) in *AWS CLI Command Reference*.

describe-journal-kinesis-stream

The following code example shows how to use `describe-journal-kinesis-stream`.

AWS CLI

To describe a journal stream

The following `describe-journal-kinesis-stream` example displays the details for the specified journal stream from a ledger.

```
aws qlldb describe-journal-kinesis-stream \  
  --ledger-name myExampleLedger \  
  --stream-id 7ISckqwe4y25YyHLzYUFAf
```

Output:

```
{  
  "Stream": {  
    "LedgerName": "myExampleLedger",  
    "CreationTime": 1591221984.677,  
    "InclusiveStartTime": 1590710400.0,  
    "ExclusiveEndTime": 1590796799.0,  
    "RoleArn": "arn:aws:iam::123456789012:role/my-kinesis-stream-role",  
    "StreamId": "7ISckqwe4y25YyHLzYUFAf",
```

```

    "Arn": "arn:aws:qldb:us-east-1:123456789012:stream/
myExampleLedger/7ISCKqe4y25YyHLzYUFAf",
    "Status": "ACTIVE",
    "KinesisConfiguration": {
        "StreamArn": "arn:aws:kinesis:us-east-1:123456789012:stream/stream-for-
qldb",
        "AggregationEnabled": true
    },
    "StreamName": "myExampleLedger-stream"
}
}

```

For more information, see [Streaming journal data from Amazon QLDB](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [DescribeJournalKinesisStream](#) in *AWS CLI Command Reference*.

describe-journal-s3-export

The following code example shows how to use `describe-journal-s3-export`.

AWS CLI

To describe a journal export job

The following `describe-journal-s3-export` example displays the details for the specified export job from a ledger.

```

aws qldb describe-journal-s3-export \
  --name myExampleLedger \
  --export-id ADR20NPKN5LINYGb4dp7yZ

```

Output:

```

{
  "ExportDescription": {
    "S3ExportConfiguration": {
      "Bucket": "awsExampleBucket",
      "Prefix": "ledgerexport1/",
      "EncryptionConfiguration": {
        "ObjectEncryptionType": "SSE_S3"
      }
    }
  }
}

```

```
    },
    "RoleArn": "arn:aws:iam::123456789012:role/my-s3-export-role",
    "Status": "COMPLETED",
    "ExportCreationTime": 1568847801.418,
    "InclusiveStartTime": 1568764800.0,
    "ExclusiveEndTime": 1568847599.0,
    "LedgerName": "myExampleLedger",
    "ExportId": "ADR20NPKN5LINYGb4dp7yZ"
  }
}
```

For more information, see [Exporting Your Journal in Amazon QLDB](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [DescribeJournalS3Export](#) in *AWS CLI Command Reference*.

describe-ledger

The following code example shows how to use `describe-ledger`.

AWS CLI

To describe a ledger

The following `describe-ledger` example displays the details for the specified ledger.

```
aws qldb describe-ledger \
  --name myExampleLedger
```

Output:

```
{
  "CreationDateTime": 1568839243.951,
  "Arn": "arn:aws:qldb:us-west-2:123456789012:ledger/myExampleLedger",
  "State": "ACTIVE",
  "Name": "myExampleLedger",
  "DeletionProtection": true,
  "PermissionsMode": "STANDARD",
  "EncryptionDescription": {
    "KmsKeyArn": "arn:aws:kms:us-west-2:123456789012:key/a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111",
    "EncryptionStatus": "ENABLED"
  }
}
```



```
}
```

For more information, see [Basic Operations for Amazon QLDB Ledgers](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [DescribeLedger](#) in *AWS CLI Command Reference*.

export-journal-to-s3

The following code example shows how to use `export-journal-to-s3`.

AWS CLI

To export journal blocks to S3

The following `export-journal-to-s3` example creates an export job for journal blocks within a specified date and time range from a ledger with the name `myExampleLedger`. The export job writes the blocks into a specified Amazon S3 bucket.

```
aws qldb export-journal-to-s3 \  
  --name myExampleLedger \  
  --inclusive-start-time 2019-09-18T00:00:00Z \  
  --exclusive-end-time 2019-09-18T22:59:59Z \  
  --role-arn arn:aws:iam::123456789012:role/my-s3-export-role \  
  --s3-export-configuration file://my-s3-export-config.json
```

Contents of `my-s3-export-config.json`:

```
{  
  "Bucket": "awsExampleBucket",  
  "Prefix": "ledgerexport1/",  
  "EncryptionConfiguration": {  
    "ObjectEncryptionType": "SSE_S3"  
  }  
}
```

Output:

```
{  
  "ExportId": "ADR2ONPKN5LINYGb4dp7yZ"  
}
```

For more information, see [Exporting Your Journal in Amazon QLDB](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [ExportJournalToS3](#) in *AWS CLI Command Reference*.

get-block

The following code example shows how to use `get-block`.

AWS CLI

Example 1: To get a journal block and proof for verification using input files

The following `get-block` example requests a block data object and a proof from the specified ledger. The request is for a specified digest tip address and block address.

```
aws qlldb get-block \  
  --name vehicle-registration \  
  --block-address file://myblockaddress.json \  
  --digest-tip-address file://mydigesttipaddress.json
```

Contents of `myblockaddress.json`:

```
{  
  "IonText": "{strandId:\\"KmA3ZZca7vAIiJAK9S5Iw1\\",sequenceNo:100}"  
}
```

Contents of `mydigesttipaddress.json`:

```
{  
  "IonText": "{strandId:\\"KmA3ZZca7vAIiJAK9S5Iw1\\",sequenceNo:123}"  
}
```

Output:

```
{  
  "Block": {  
    "IonText": "{blockAddress:{strandId:\\"KmA3ZZca7vAIiJAK9S5Iw1\  
\\",sequenceNo:100},transactionId:\\"FnQeJBAicTX0Ah32ZnVtSX\  
\\",blockTimestamp:2019-09-16T19:37:05.360Z,blockHash:
```

```

{{NoChM92yKRuJAb/jeLd1VnYn4DHiWIf071ACfic9uHc=}}, entriesHash:
{{105L0siKV14SdbuaYnH7uwXzUvqzIwUiRLXGbTyj/nY=}}, previousBlockHash:
{{7kewBXhpdBc1cZKxhVmpoMHPUG0JtWQD0iY2LPfZkYA=}}, entriesHashList:
[{{eRSwnmAM7WWANWdD5iG0yK+T4tDXyzUq6HZ/0fgLHos=}}, {{mHVex/
yJHAWjFPpwhBuH2GKXmKJjK2FBa9faquUVNtg=}}],
{{y5cCB7p0AIUfsVQ1j0TqtE97b4b4oo1R0vnYyE5wWM=}}, {{TvTXygML1bMe6NvEZtGkX
+KR+W/EJl4qD1mmV77KZQg=}}], transactionInfo: {statements: [{statement:
\FROM VehicleRegistration AS r \\\nWHERE r.VIN = '1N4AL11D75C109151'\
\nINSERT INTO r.Owners.SecondaryOwners\\\n    VALUE { 'PersonId' :
'CMVdR77XP8zAg1mmFDGTvt' }\}], startTime:2019-09-16T19:37:05.302Z, statementDigest:
{{jcgPX2vs0J0waum4qmDYtn1pCAT9xKNIzA+2k4R+mxA=}}}], documents:
{JUJgkIcNbhS2goq8RqLuZ4: {tableName:\\"VehicleRegistration\\", tableId:
\\"BFJKdXgzT9oF4wjMbuxy4G\\", statements: [0]}}}, revisions: [{blockAddress:
{strandId:\\"KmA3ZZca7vAIiJAK9S5Iw1\\", sequenceNo:100}, hash:
{{mHVex/yJHAWjFPpwhBuH2GKXmKJjK2FBa9faquUVNtg=}}, data: {VIN:
\\"1N4AL11D75C109151\\", LicensePlateNumber:\\"LEWISR261LL\\", State:\\"WA
\\", PendingPenaltyTicketAmount:90.25, ValidFromDate:2017-08-21, ValidToDate:2020-05-11, Owners:
{PrimaryOwner: {PersonId:\\"BFJKdXhnLRT27sXBnojNGW\\"}, SecondaryOwners:
[{{PersonId:\\"CMVdR77XP8zAg1mmFDGTvt\\"}]}], City:\\"Everett\\"}, metadata: {id:
\\"JUJgkIcNbhS2goq8RqLuZ4\\"}, version:3, txTime:2019-09-16T19:37:05.344Z, txId:
\\"FnQeJBAicTX0Ah32ZnVtSX\\"}}}]
    },
    "Proof": {
        "IonText": "[{{13+EXs69K1+rehlqyWLkt+oHDlw4Zi9pCLW/t/mgTPM=}},
{{48CXG3ehPqsxCYd34EEa8Fso00RpWwA08010RJKf3Do=}}, {{9UnwnKSQT0i3ge1JMVa
+tMIqCEDaOPTkwxmyHSn8UPQ=}}, {{3nW6Vryghk+7pd6wFCtLufgPM6qXHyTNeCb1sCwcDaI=}},
{{Irb5fNhBrNEQ1VPhz1nGT/ZQPadSmgfdtMYcwkN0xoI=}}, {{+3CwpYG/ytf/
vq9GidpzSx6JJiLXt1hMQWnNq0y3jfY=}}, {{NPx6cRhwsiy5m9UEWS5JTJrZoUd02jB0AA0myZAT
+qE=}}]"
    }
}

```

For more information, see [Data Verification in Amazon QLDB](#) in the *Amazon QLDB Developer Guide*.

Example 2: To get a journal block and proof for verification using shorthand syntax

The following `get-block` example requests a block data object and a proof from the specified ledger using shorthand syntax. The request is for a specified digest tip address and block address.

```

aws qlldb get-block \
  --name vehicle-registration \

```

```
--block-address 'IonText="{strandId:\\"KmA3ZZca7vAIiJAK9S5Iw1\\",sequenceNo:100}"'
\
--digest-tip-address 'IonText="{strandId:\\"KmA3ZZca7vAIiJAK9S5Iw1
\\",sequenceNo:123}"'
```

Output:

```
{
  "Block": {
    "IonText": "{blockAddress:{strandId:\\"KmA3ZZca7vAIiJAK9S5Iw1
\\",sequenceNo:100},transactionId:\\"FnQeJBAicTX0Ah32ZnVtSX
\\",blockTimestamp:2019-09-16T19:37:05.360Z,blockHash:
{{NoChM92yKRuJAb/jeLd1VnYn4DHiWIf071ACfic9uHc=}},entriesHash:
{{105L0siKV14SDbuaYnH7uwXzUvqzIwUiRLXGbTyj/nY=}},previousBlockHash:
{{7kewBXhpdBc1cZKxhVmpoMHPUG0JtWQD0iY2LPfZkYA=}},entriesHashList:
[{{eRSwnmAM7WWANWdD5iG0yK+T4tDXyzUq6HZ/0fgLHos=}},{{mHVex/
yjHAWjFPPwhBuH2GKXmKjK2FBa9faqoUVNtg=}},
{{y5cCB7p0AIUfsVQ1j0TqtE97b4b4oo1R0vnYyE5wWM=}},{{TvTXygML1bMe6NvEZtGkX
+KR+W/EJl4qd1mmV77KZQg=}}],transactionInfo:{statements:[{statement:
\\"FROM VehicleRegistration AS r \\nWHERE r.VIN = '1N4AL11D75C109151'\\n
\\nINSERT INTO r.Owners.SecondaryOwners\\n    VALUE { 'PersonId' :
'CMVdR77XP8zAglmmFDGTvt' }\\n",startTime:2019-09-16T19:37:05.302Z,statementDigest:
{{jcgPX2vs0J0waum4qmDYtn1pCAT9xKNIzA+2k4R+mxA=}}}],documents:
{JUJgkIcNbhS2goq8RqLuZ4:{tableName:\\"VehicleRegistration\\",tableId:
\\"BFJKdXgzT9oF4wjMbuxy4G\\",statements:[0]}}],revisions:[{blockAddress:
{strandId:\\"KmA3ZZca7vAIiJAK9S5Iw1\\",sequenceNo:100},hash:
{{mHVex/yjHAWjFPPwhBuH2GKXmKjK2FBa9faqoUVNtg=}},data:{VIN:
\\"1N4AL11D75C109151\\",LicensePlateNumber:\\"LEWISR261LL\\",State:\\"WA
\\",PendingPenaltyTicketAmount:90.25,ValidFromDate:2017-08-21,ValidToDate:2020-05-11,Owners:
{PrimaryOwner:{PersonId:\\"BFJKdXhnLRT27sXBnojNGW\\"},SecondaryOwners:
[{{PersonId:\\"CMVdR77XP8zAglmmFDGTvt\\"}]}],City:\\"Everett\\"},metadata:{id:
\\"JUJgkIcNbhS2goq8RqLuZ4\\",version:3,txTime:2019-09-16T19:37:05.344Z,txId:
\\"FnQeJBAicTX0Ah32ZnVtSX\\"}}]}]"},
  },
  "Proof": {
    "IonText": "[{{13+EXs69K1+rehlqyWLkt+oHDlw4Zi9pCLW/t/mgTPM=}},
{{48CXG3ehPqsxCYd34EEa8Fso00RpWwA08010RJkF3Do=}},{{9UnwnKSQT0i3ge1JMVa
+tMIqCEDaOPTkwxmyHSn8UPQ=}},{{3nW6Vryghk+7pd6wFctLufgPM6qXHyTNeCb1sCwcDaI=}},
{{Irb5fNhBrNEQ1VPhzlnGT/ZQPadSmgfdtMYcwkN0xoI=}},{{+3CwpYG/ytf/
vq9GidpzSx6JJiLXt1hMQWnNq0y3jfY=}},{{NPx6cRhwsiy5m9UEWS5JTJrZoUd02jB0AA0myZAT
+qE=}}]"
  }
}
```

For more information, see [Data Verification in Amazon QLDB](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [GetBlock](#) in *AWS CLI Command Reference*.

get-digest

The following code example shows how to use `get-digest`.

AWS CLI

To get a digest for a ledger

The following `get-digest` example requests a digest from the specified ledger at the latest committed block in the journal.

```
aws qlldb get-digest \  
  --name vehicle-registration
```

Output:

```
{  
  "Digest": "6m6BMXobbJKpMhahwVthAEsN6awgnHK62Qq5McGP1Gk=",  
  "DigestTipAddress": {  
    "IonText": "{strandId:\\"KmA3ZZca7vAIiJAK9S5Iw1\\",sequenceNo:123}"  
  }  
}
```

For more information, see [Data Verification in Amazon QLDB](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [GetDigest](#) in *AWS CLI Command Reference*.

get-revision

The following code example shows how to use `get-revision`.

AWS CLI

Example 1: To get a document revision and proof for verification using input files

The following `get-revision` example requests a revision data object and a proof from the specified ledger. The request is for a specified digest tip address, document ID, and block address of the revision.

```
aws qlldb get-revision \
  --name vehicle-registration \
  --block-address file://myblockaddress.json \
  --document-id JUJgkIcNbhS2goq8RqLuZ4 \
  --digest-tip-address file://mydigesttipaddress.json
```

Contents of `myblockaddress.json`:

```
{
  "IonText": "{strandId:\\"KmA3ZZca7vAIiJAK9S5Iw1\\",sequenceNo:100}"
}
```

Contents of `mydigesttipaddress.json`:

```
{
  "IonText": "{strandId:\\"KmA3ZZca7vAIiJAK9S5Iw1\\",sequenceNo:123}"
}
```

Output:

```
{
  "Revision": {
    "IonText": "{blockAddress:{strandId:\\"KmA3ZZca7vAIiJAK9S5Iw1\\",sequenceNo:100},hash:{{mHVex/yjHAWjFPpwhBuH2GKXmKJjK2FBa9faqoUVNtg=}},data:
    {VIN:\\"1N4AL11D75C109151\\",LicensePlateNumber:\\"LEWISR261LL\\",State:\\"WA\\",PendingPenaltyTicketAmount:90.25,ValidFromDate:2017-08-21,ValidToDate:2020-05-11,Owners:
    {PrimaryOwner:{PersonId:\\"BFJKdXhnLRT27sXBnojNGW\\"},SecondaryOwners:
    [{PersonId:\\"CMVdR77XP8zAg1mmFDGTvt\\"}]},City:\\"Everett\\"},metadata:{id:
    \\"JUJgkIcNbhS2goq8RqLuZ4\\",version:3,txTime:2019-09-16T19:37:05.344Z,txId:
    \\"FnQeJBAicTX0Ah32ZnVtSX\\"}}}"
  },
  "Proof": {
    "IonText": "[{{eRSwnmAM7WWANWDD5iG0yK+T4tDXyzUq6HZ/0fgLHos=}},{{VV1rdaNuf
    +yJZVGlmsM6gr2T52QvB08Lg+KgpjcnWAU=}},
    {{7kewBXhpdbClcZKxhVmpoMHPUGOJtWQD0iY2LPfZkYA=}},{{13+EXs69K1+rehlqyWLkt
    +oHD1w4Zi9pCLW/t/mgTPM=}},{{48CXG3ehPqsxCYd34EEa8Fso00RpWAA08010RJkf3Do=}},
    {{9UnwnKSQT0i3ge1JMVa+tMIqCEDaOPTkwxmyHSn8UPQ=}},{{3nW6Vryghk
    +7pd6wFctLufgPM6qxHyTNeCb1sCwcDaI=}},{{Irb5fNhBrNEQ1VPhzlnGT/
```

```
ZQPadSmgfdtMYcwkN0xoI=}}, {{+3CWpYG/ytf/vq9GidpzSx6JJiLXt1hMQWNnq0y3jfY=}},
{{NPx6cRhwsiy5m9UEWS5JTJrZoUd02jB0AA0myZAT+qE=}}]"
  }
}
```

For more information, see [Data Verification in Amazon QLDB](#) in the *Amazon QLDB Developer Guide*.

Example 2: To get a document revision and proof for verification using shorthand syntax

The following `get-revision` example requests a revision data object and a proof from the specified ledger using shorthand syntax. The request is for a specified digest tip address, document ID, and block address of the revision.

```
aws qldb get-revision \
  --name vehicle-registration \
  --block-address 'IonText="{strandId:\\"KmA3ZZca7vAIiJAK9S5Iw1\\",sequenceNo:100}"'
  \
  --document-id JUJgkIcNbhS2goq8RqLuZ4 \
  --digest-tip-address 'IonText="{strandId:\\"KmA3ZZca7vAIiJAK9S5Iw1
  \\",sequenceNo:123}"'
```

Output:

```
{
  "Revision": {
    "IonText": "{blockAddress:{strandId:\\"KmA3ZZca7vAIiJAK9S5Iw1
  \\",sequenceNo:100},hash:{{mHVex/yjHAWjFPpwhBuH2GKXmKJjK2FBa9faqoUVNtg=}},data:
  {VIN:\\"1N4AL11D75C109151\\",LicensePlateNumber:\\"LEWISR261LL\\",State:\\"WA
  \\",PendingPenaltyTicketAmount:90.25,ValidFromDate:2017-08-21,ValidToDate:2020-05-11,Owners:
  {PrimaryOwner:{PersonId:\\"BFJKdXhnLRT27sXBnojNGW\\"},SecondaryOwners:
  [{PersonId:\\"CMVdR77XP8zAg1mmFDGTvt\\"}]},City:\\"Everett\\"},metadata:{id:
  \\"JUJgkIcNbhS2goq8RqLuZ4\\",version:3,txTime:2019-09-16T19:37:05.344Z,txId:
  \\"FnQeJBAicTX0Ah32ZnVtSX\\"}}}"
  },
  "Proof": {
    "IonText": "[{{eRSwnmAM7WWANWdd5iG0yK+T4tDXyzUq6HZ/0fgLHos=}},{{VV1rdaNuf
  +yJZVGlmsM6gr2T52QvB08Lg+KgpjcnWAU=}},
  {{7kewBXhpdBc1cZKxhVmpoMhpUGOJtWQD0iY2LPfZkYA=}},{{13+EXs69K1+rehlqyWLkt
  +oHD1w4Zi9pCLW/t/mgTPM=}},{{48CXG3ehPqsxCYd34EEa8Fso00RpWwA08010RJkf3Do=}},
  {{9UnwnKSQT0i3ge1JMva+tMIqCEDaOPTkwxmyHSn8UPQ=}},{{3nW6Vryghk
  +7pd6wFctLufgPM6qxHyTNeCb1sCwcDaI=}},{{Irb5fNhBrNEQ1VPhzlnGT/
```

```
ZQPadSmgfdtMYcwkN0xoI=}}, {{+3CWpYG/ytf/vq9GidpzSx6JJiLXt1hMQWNnq0y3jfY=}},
{{NPx6cRhwsiy5m9UEWS5JTJrZoUd02jB0AA0myZAT+qE=}}]"
  }
}
```

For more information, see [Data Verification in Amazon QLDB](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [GetRevision](#) in *AWS CLI Command Reference*.

list-journal-kinesis-streams-for-ledger

The following code example shows how to use `list-journal-kinesis-streams-for-ledger`.

AWS CLI

To list journal streams for a ledger

The following `list-journal-kinesis-streams-for-ledger` example lists journal streams for the specified ledger.

```
aws qlldb list-journal-kinesis-streams-for-ledger \
  --ledger-name myExampleLedger
```

Output:

```
{
  "Streams": [
    {
      "LedgerName": "myExampleLedger",
      "CreationTime": 1591221984.677,
      "InclusiveStartTime": 1590710400.0,
      "ExclusiveEndTime": 1590796799.0,
      "RoleArn": "arn:aws:iam::123456789012:role/my-kinesis-stream-role",
      "StreamId": "7ISCKqwe4y25YyHLzYUFaf",
      "Arn": "arn:aws:qlldb:us-east-1:123456789012:stream/
myExampleLedger/7ISCKqwe4y25YyHLzYUFaf",
      "Status": "ACTIVE",
      "KinesisConfiguration": {
        "StreamArn": "arn:aws:kinesis:us-east-1:123456789012:stream/stream-
for-qlldb",
        "AggregationEnabled": true
      },
    },
  ],
}
```



```

        "StreamName": "myExampleLedger-stream"
      }
    ]
  }

```

For more information, see [Streaming journal data from Amazon QLDB](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [ListJournalKinesisStreamsForLedger](#) in *AWS CLI Command Reference*.

list-journal-s3-exports-for-ledger

The following code example shows how to use `list-journal-s3-exports-for-ledger`.

AWS CLI

To list journal export jobs for a ledger

The following `list-journal-s3-exports-for-ledger` example lists journal export jobs for the specified ledger.

```

aws qlldb list-journal-s3-exports-for-ledger \
  --name myExampleLedger

```

Output:

```

{
  "JournalS3Exports": [
    {
      "LedgerName": "myExampleLedger",
      "ExclusiveEndTime": 1568847599.0,
      "ExportCreationTime": 1568847801.418,
      "S3ExportConfiguration": {
        "Bucket": "awsExampleBucket",
        "Prefix": "ledgerexport1/",
        "EncryptionConfiguration": {
          "ObjectEncryptionType": "SSE_S3"
        }
      },
      "ExportId": "ADR20NPKN5LINYGb4dp7yZ",
      "RoleArn": "arn:aws:iam::123456789012:role/qlldb-s3-export",
      "InclusiveStartTime": 1568764800.0,
      "Status": "IN_PROGRESS"
    }
  ]
}

```

```
    }  
  ]  
}
```

For more information, see [Exporting Your Journal in Amazon QLDB](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [ListJournalS3ExportsForLedger](#) in *AWS CLI Command Reference*.

list-journal-s3-exports

The following code example shows how to use `list-journal-s3-exports`.

AWS CLI

To list journal export jobs

The following `list-journal-s3-exports` example lists journal export jobs for all ledgers that are associated with the current AWS account and Region.

```
aws qlldb list-journal-s3-exports
```

Output:

```
{  
  "JournalS3Exports": [  
    {  
      "Status": "IN_PROGRESS",  
      "LedgerName": "myExampleLedger",  
      "S3ExportConfiguration": {  
        "EncryptionConfiguration": {  
          "ObjectEncryptionType": "SSE_S3"  
        },  
        "Bucket": "awsExampleBucket",  
        "Prefix": "ledgerexport1/"  
      },  
      "RoleArn": "arn:aws:iam::123456789012:role/my-s3-export-role",  
      "ExportCreationTime": 1568847801.418,  
      "ExportId": "ADR20NPKN5LINYGb4dp7yZ",  
      "InclusiveStartTime": 1568764800.0,  
      "ExclusiveEndTime": 1568847599.0  
    },  
    {
```

```
"Status": "COMPLETED",
"LedgerName": "myExampleLedger2",
"S3ExportConfiguration": {
  "EncryptionConfiguration": {
    "ObjectEncryptionType": "SSE_S3"
  },
  "Bucket": "awsExampleBucket",
  "Prefix": "ledgerexport1/"
},
"RoleArn": "arn:aws:iam::123456789012:role/my-s3-export-role",
"ExportCreationTime": 1568846847.638,
"ExportId": "2pdvW8UQrjBAiYTMehEJDI",
"InclusiveStartTime": 1568592000.0,
"ExclusiveEndTime": 1568764800.0
}
]
}
```

For more information, see [Exporting Your Journal in Amazon QLDB](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [ListJournalS3Exports](#) in *AWS CLI Command Reference*.

list-ledgers

The following code example shows how to use `list-ledgers`.

AWS CLI

To list your available ledgers

The following `list-ledgers` example lists all ledgers that are associated with the current AWS account and Region.

```
aws qldb list-ledgers
```

Output:

```
{
  "Ledgers": [
    {
      "State": "ACTIVE",
      "CreationDateTime": 1568839243.951,
```

```
        "Name": "myExampleLedger"
      },
      {
        "State": "ACTIVE",
        "CreationDateTime": 1568839543.557,
        "Name": "myExampleLedger2"
      }
    ]
  }
}
```

For more information, see [Basic Operations for Amazon QLDB Ledgers](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [ListLedgers](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags attached to a ledger

The following `list-tags-for-resource` example lists all tags attached to the specified ledger.

```
aws qldb list-tags-for-resource \
  --resource-arn arn:aws:qldb:us-west-2:123456789012:ledger/myExampleLedger
```

Output:

```
{
  "Tags": {
    "IsTest": "true",
    "Domain": "Test"
  }
}
```

For more information, see [Tagging Amazon QLDB Resources](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

stream-journal-to-kinesis

The following code example shows how to use `stream-journal-to-kinesis`.

AWS CLI

Example 1: To stream journal data to Kinesis Data Streams using input files

The following `stream-journal-to-kinesis` example creates a stream of journal data within a specified date and time range from a ledger with the name `myExampleLedger`. The stream sends the data to a specified Amazon Kinesis data stream.

```
aws qlldb stream-journal-to-kinesis \  
  --ledger-name myExampleLedger \  
  --inclusive-start-time 2020-05-29T00:00:00Z \  
  --exclusive-end-time 2020-05-29T23:59:59Z \  
  --role-arn arn:aws:iam::123456789012:role/my-kinesis-stream-role \  
  --kinesis-configuration file://my-kinesis-config.json \  
  --stream-name myExampleLedger-stream
```

Contents of `my-kinesis-config.json`:

```
{  
  "StreamArn": "arn:aws:kinesis:us-east-1:123456789012:stream/stream-for-qlldb",  
  "AggregationEnabled": true  
}
```

Output:

```
{  
  "StreamId": "7ISckqwe4y25YyHLzYUFAf"  
}
```

For more information, see [Streaming journal data from Amazon QLDB](#) in the *Amazon QLDB Developer Guide*.

Example 2: To stream journal data to Kinesis Data Streams using shorthand syntax

The following `stream-journal-to-kinesis` example creates a stream of journal data within a specified date and time range from a ledger with the name `myExampleLedger`. The stream sends the data to a specified Amazon Kinesis data stream.

```
aws qlldb stream-journal-to-kinesis \  
  --ledger-name myExampleLedger \  
  --inclusive-start-time 2020-05-29T00:00:00Z \  
  --exclusive-end-time 2020-05-29T23:59:59Z \  
  --role-arn arn:aws:iam::123456789012:role/my-kinesis-stream-role \  
  --stream-name myExampleLedger-stream \  
  --kinesis-configuration StreamArn=arn:aws:kinesis:us-east-1:123456789012:stream/  
stream-for-qlldb,AggregationEnabled=true
```

Output:

```
{  
  "StreamId": "7ISCKqwe4y25YyHLzYUFAf"  
}
```

For more information, see [Streaming journal data from Amazon QLDB](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [StreamJournalToKinesis](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To tag a ledger

The following `tag-resource` example adds a set of tags to a specified ledger.

```
aws qlldb tag-resource \  
  --resource-arn arn:aws:qlldb:us-west-2:123456789012:ledger/myExampleLedger \  
  --tags IsTest=true,Domain=Test
```

This command produces no output.

For more information, see [Tagging Amazon QLDB Resources](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from a resource

The following `untag-resource` example removes tags with the specified tag keys from a specified ledger.

```
aws qlldb untag-resource \  
  --resource-arn arn:aws:qlldb:us-west-2:123456789012:ledger/myExampleLedger \  
  --tag-keys IsTest Domain
```

This command produces no output.

For more information, see [Tagging Amazon QLDB Resources](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-ledger-permissions-mode

The following code example shows how to use `update-ledger-permissions-mode`.

AWS CLI

Example 1: To update the permissions mode of a ledger to STANDARD

The following `update-ledger-permissions-mode` example assigns the STANDARD permissions mode to the specified ledger.

```
aws qlldb update-ledger-permissions-mode \  
  --name myExampleLedger \  
  --permissions-mode STANDARD
```

Output:

```
{  
  "Name": "myExampleLedger",  
  "Arn": "arn:aws:qlldb:us-west-2:123456789012:ledger/myExampleLedger",  
  "PermissionsMode": "STANDARD"
```

```
}
```

Example 2: To update the permissions mode of a ledger to ALLOW_ALL

The following `update-ledger-permissions-mode` example assigns the `ALLOW_ALL` permissions mode to the specified ledger.

```
aws qlldb update-ledger-permissions-mode \  
  --name myExampleLedger \  
  --permissions-mode ALLOW_ALL
```

Output:

```
{  
  "Name": "myExampleLedger",  
  "Arn": "arn:aws:qlldb:us-west-2:123456789012:ledger/myExampleLedger",  
  "PermissionsMode": "ALLOW_ALL"  
}
```

For more information, see [Basic Operations for Amazon QLDB Ledgers](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [UpdateLedgerPermissionsMode](#) in *AWS CLI Command Reference*.

update-ledger

The following code example shows how to use `update-ledger`.

AWS CLI

Example 1: To update the deletion protection property of a ledger

The following `update-ledger` example updates the specified ledger to disable the deletion protection feature.

```
aws qlldb update-ledger \  
  --name myExampleLedger \  
  --no-deletion-protection
```

Output:

```
{
```



```
"CreationDateTime": 1568839243.951,  
"Arn": "arn:aws:qldb:us-west-2:123456789012:ledger/myExampleLedger",  
"DeletionProtection": false,  
"Name": "myExampleLedger",  
"State": "ACTIVE"  
}
```

Example 2: To update the AWS KMS key of a ledger to a customer managed key

The following `update-ledger` example updates the specified ledger to use a customer managed KMS key for encryption at rest.

```
aws qldb update-ledger \  
  --name myExampleLedger \  
  --kms-key arn:aws:kms:us-west-2:123456789012:key/a1b2c3d4-5678-90ab-cdef-  
EXAMPLE11111
```

Output:

```
{  
  "CreationDateTime": 1568839243.951,  
  "Arn": "arn:aws:qldb:us-west-2:123456789012:ledger/myExampleLedger",  
  "DeletionProtection": false,  
  "Name": "myExampleLedger",  
  "State": "ACTIVE",  
  "EncryptionDescription": {  
    "KmsKeyArn": "arn:aws:kms:us-west-2:123456789012:key/a1b2c3d4-5678-90ab-  
cdef-EXAMPLE11111",  
    "EncryptionStatus": "UPDATING"  
  }  
}
```

Example 3: To update the AWS KMS key of a ledger to an AWS owned key

The following `update-ledger` example updates the specified ledger to use an AWS owned KMS key for encryption at rest.

```
aws qldb update-ledger \  
  --name myExampleLedger \  
  --kms-key AWS_OWNED_KMS_KEY
```

Output:

```
{
  "CreationDateTime": 1568839243.951,
  "Arn": "arn:aws:qldb:us-west-2:123456789012:ledger/myExampleLedger",
  "DeletionProtection": false,
  "Name": "myExampleLedger",
  "State": "ACTIVE",
  "EncryptionDescription": {
    "KmsKeyArn": "AWS_OWNED_KMS_KEY",
    "EncryptionStatus": "UPDATING"
  }
}
```

For more information, see [Basic Operations for Amazon QLDB Ledgers](#) in the *Amazon QLDB Developer Guide*.

- For API details, see [UpdateLedger](#) in *AWS CLI Command Reference*.

Amazon RDS examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon RDS.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-option-to-option-group

The following code example shows how to use `add-option-to-option-group`.

AWS CLI

To add an option to an option group

The following add-option-to-option-group example adds an option to the specified option group.

```
aws rds add-option-to-option-group \  
  --option-group-name myoptiongroup \  
  --options OptionName=OEM,Port=5500,DBSecurityGroupMemberships=default \  
  --apply-immediately
```

Output:

```
{  
  "OptionGroup": {  
    "OptionGroupName": "myoptiongroup",  
    "OptionGroupDescription": "Test Option Group",  
    "EngineName": "oracle-ee",  
    "MajorEngineVersion": "12.1",  
    "Options": [  
      {  
        "OptionName": "Timezone",  
        "OptionDescription": "Change time zone",  
        "Persistent": true,  
        "Permanent": false,  
        "OptionSettings": [  
          {  
            "Name": "TIME_ZONE",  
            "Value": "Australia/Sydney",  
            "DefaultValue": "UTC",  
            "Description": "Specifies the timezone the user wants to  
change the system time to",  
            "ApplyType": "DYNAMIC",  
            "DataType": "STRING",  
            "AllowedValues": "Africa/Cairo,Africa/Casablanca,Africa/  
Harare,Africa/Lagos,Africa/Luanda,Africa/Monrovia,Africa/Nairobi,Africa/  
Tripoli,Africa/Windhoek,America/Araguaina,America/Argentina/Buenos_Aires,America/  
Asuncion,America/Bogota,America/Caracas,America/Chicago,America/Chihuahua,America/  
Cuiaba,America/Denver,America/Detroit,America/Fortaleza,America/Godthab,America/  
Guatemala,America/Halifax,America/Lima,America/Los_Angeles,America/Manaus,America/  
Matamoros,America/Mexico_City,America/Monterrey,America/Montevideo,America/  
New_York,America/Phoenix,America/Santiago,America/Sao_Paulo,America/Tijuana,America/
```

```

Toronto,Asia/Amman,Asia/Ashgabat,Asia/Baghdad,Asia/Baku,Asia/Bangkok,Asia/
Beirut,Asia/Calcutta,Asia/Damascus,Asia/Dhaka,Asia/Hong_Kong,Asia/Irkutsk,Asia/
Jakarta,Asia/Jerusalem,Asia/Kabul,Asia/Karachi,Asia/Kathmandu,Asia/Kolkata,Asia/
Krasnoyarsk,Asia/Magadan,Asia/Manila,Asia/Muscat,Asia/Novosibirsk,Asia/Rangoon,Asia/
Riyadh,Asia/Seoul,Asia/Shanghai,Asia/Singapore,Asia/Taipei,Asia/Tehran,Asia/
Tokyo,Asia/Ulaanbaatar,Asia/Vladivostok,Asia/Yakutsk,Asia/Yerevan,Atlantic/
Azores,Atlantic/Cape_Verde,Australia/Adelaide,Australia/Brisbane,Australia/
Darwin,Australia/Eucla,Australia/Hobart,Australia/Lord_Howe,Australia/
Perth,Australia/Sydney,Brazil/DeNoronha,Brazil/East,Canada/Newfoundland,Canada/
Saskatchewan,Etc/GMT-3,Europe/Amsterdam,Europe/Athens,Europe/Berlin,Europe/
Dublin,Europe/Helsinki,Europe/Kaliningrad,Europe/London,Europe/Madrid,Europe/
Moscow,Europe/Paris,Europe/Prague,Europe/Rome,Europe/Sarajevo,Pacific/Apia,Pacific/
Auckland,Pacific/Chatham,Pacific/Fiji,Pacific/Guam,Pacific/Honolulu,Pacific/
Kiritimati,Pacific/Marquesas,Pacific/Samoa,Pacific/Tongatapu,Pacific/Wake,US/
Alaska,US/Central,US/East-Indiana,US/Eastern,US/Pacific,UTC",
        "IsModifiable": true,
        "IsCollection": false
    }
],
"DBSecurityGroupMemberships": [],
"VpcSecurityGroupMemberships": []
},
{
    "OptionName": "OEM",
    "OptionDescription": "Oracle 12c EM Express",
    "Persistent": false,
    "Permanent": false,
    "Port": 5500,
    "OptionSettings": [],
    "DBSecurityGroupMemberships": [
        {
            "DBSecurityGroupName": "default",
            "Status": "authorized"
        }
    ],
    "VpcSecurityGroupMemberships": []
}
],
"AllowsVpcAndNonVpcInstanceMemberships": false,
"OptionGroupArn": "arn:aws:rds:us-east-1:123456789012:og:myoptiongroup"
}
}

```

For more information, see [Adding an Option to an Option Group](#) in the *Amazon RDS User Guide*.

- For API details, see [AddOptionToOptionGroup](#) in *AWS CLI Command Reference*.

add-role-to-db-cluster

The following code example shows how to use `add-role-to-db-cluster`.

AWS CLI

To associate an AWS Identity and Access Management (IAM) role with a DB cluster

The following `add-role-to-db-cluster` example associates a role with a DB cluster.

```
aws rds add-role-to-db-cluster \  
  --db-cluster-identifier mydbcluster \  
  --role-arn arn:aws:iam::123456789012:role/RDSLoadFromS3
```

This command produces no output.

For more information, see [Associating an IAM role with an Amazon Aurora MySQL DB cluster](#) in the *Amazon Aurora User Guide*.

- For API details, see [AddRoleToDbCluster](#) in *AWS CLI Command Reference*.

add-role-to-db-instance

The following code example shows how to use `add-role-to-db-instance`.

AWS CLI

To associate an AWS Identity and Access Management (IAM) role with a DB instance

The following `add-role-to-db-instance` example adds the role to an Oracle DB instance named `test-instance`.

```
aws rds add-role-to-db-instance \  
  --db-instance-identifier test-instance \  
  --feature-name S3_INTEGRATION \  
  --role-arn arn:aws:iam::111122223333:role/rds-s3-integration-role
```

This command produces no output.

For more information, see [Prerequisites for Amazon RDS Oracle Integration with Amazon S3](#) in the *Amazon RDS User Guide*.

- For API details, see [AddRoleToDbInstance](#) in *AWS CLI Command Reference*.

add-source-identifier-to-subscription

The following code example shows how to use `add-source-identifier-to-subscription`.

AWS CLI

To add a source identifier to a subscription

The following `add-source-identifier` example adds another source identifier to an existing subscription.

```
aws rds add-source-identifier-to-subscription \  
  --subscription-name my-instance-events \  
  --source-identifier test-instance-repl
```

Output:

```
{  
  "EventSubscription": {  
    "SubscriptionCreationTime": "Tue Jul 31 23:22:01 UTC 2018",  
    "CustSubscriptionId": "my-instance-events",  
    "EventSubscriptionArn": "arn:aws:rds:us-east-1:123456789012:es:my-instance-  
events",  
    "Enabled": false,  
    "Status": "modifying",  
    "EventCategoriesList": [  
      "backup",  
      "recovery"  
    ],  
    "CustomerAwsId": "123456789012",  
    "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:interesting-events",  
    "SourceType": "db-instance",  
    "SourceIdsList": [  
      "test-instance",  
      "test-instance-repl"  
    ]  
  }  
}
```

```
}
```

- For API details, see [AddSourceIdentifierToSubscription](#) in *AWS CLI Command Reference*.

add-tags-to-resource

The following code example shows how to use `add-tags-to-resource`.

AWS CLI

To add tags to a resource

The following `add-tags-to-resource` example add tags to an RDS database.

```
aws rds add-tags-to-resource \  
  --resource-name arn:aws:rds:us-east-1:123456789012:db:database-mysql \  
  --tags "[{\"Key\": \"Name\", \"Value\": \"MyDatabase\"}, {\"Key\": \"Environment\", \"Value\": \"test\"}]\"
```

This command produces no output.

For more information, see [Tagging Amazon RDS Resources](#) in the *Amazon RDS User Guide*.

- For API details, see [AddTagsToResource](#) in *AWS CLI Command Reference*.

apply-pending-maintenance-action

The following code example shows how to use `apply-pending-maintenance-action`.

AWS CLI

To apply pending maintenance actions

The following `apply-pending-maintenance-action` example applies the pending maintenance actions for a DB cluster.

```
aws rds apply-pending-maintenance-action \  
  --resource-identifier arn:aws:rds:us-east-1:123456789012:cluster:my-db-cluster \  
  --apply-action system-update \  
  --opt-in-type immediate
```

Output:

```
{
  "ResourcePendingMaintenanceActions": {
    "ResourceIdentifier": "arn:aws:rds:us-east-1:123456789012:cluster:my-db-cluster",
    "PendingMaintenanceActionDetails": [
      {
        "Action": "system-update",
        "OptInStatus": "immediate",
        "CurrentApplyDate": "2021-01-23T01:07:36.100Z",
        "Description": "Upgrade to Aurora PostgreSQL 3.3.2"
      }
    ]
  }
}
```

For more information, see [Maintaining a DB instance](#) in the *Amazon RDS User Guide* and [Maintaining an Amazon Aurora DB cluster](#) in the *Amazon Aurora User Guide*.

- For API details, see [ApplyPendingMaintenanceAction](#) in *AWS CLI Command Reference*.

authorize-db-security-group-ingress

The following code example shows how to use `authorize-db-security-group-ingress`.

AWS CLI

To associate an AWS Identity and Access Management (IAM) role with a DB instance

The following `authorize-db-security-group-ingress` example configures the default security group with an ingress rule for the CIDR IP range 192.0.2.0/24.

```
aws rds authorize-db-security-group-ingress \
  --db-security-group-name default \
  --cidrip 192.0.2.0/24
```

Output:

```
{
  "DBSecurityGroup": {
    "OwnerId": "123456789012",
    "DBSecurityGroupName": "default",
```



```

    "DBSecurityGroupDescription": "default",
    "EC2SecurityGroups": [],
    "IPRanges": [
      {
        "Status": "authorizing",
        "CIDRIP": "192.0.2.0/24"
      }
    ],
    "DBSecurityGroupArn": "arn:aws:rds:us-east-1:111122223333:secgrp:default"
  }
}

```

For more information, see [Authorizing Network Access to a DB Security Group from an IP Range](#) in the *Amazon RDS User Guide*.

- For API details, see [AuthorizeDbSecurityGroupIngress](#) in *AWS CLI Command Reference*.

backtrack-db-cluster

The following code example shows how to use `backtrack-db-cluster`.

AWS CLI

To backtrack an Aurora DB cluster

The following `backtrack-db-cluster` example backtracks the specified DB cluster `sample-cluster` to March 19, 2018, at 10 a.m.

```
aws rds backtrack-db-cluster --db-cluster-identifier sample-cluster --backtrack-to
2018-03-19T10:00:00+00:00
```

This command outputs a JSON block that acknowledges the change to the RDS resource.

- For API details, see [BacktrackDbCluster](#) in *AWS CLI Command Reference*.

cancel-export-task

The following code example shows how to use `cancel-export-task`.

AWS CLI

To cancel a snapshot export to Amazon S3

The following `cancel-export-task` example cancels an export task in progress that is exporting a snapshot to Amazon S3.

```
aws rds cancel-export-task \  
  --export-task-identifier my-s3-export-1
```

Output:

```
{  
  "ExportTaskIdentifier": "my-s3-export-1",  
  "SourceArn": "arn:aws:rds:us-east-1:123456789012:snapshot:publisher-final-  
snapshot",  
  "SnapshotTime": "2019-03-24T20:01:09.815Z",  
  "S3Bucket": "mybucket",  
  "S3Prefix": "",  
  "IamRoleArn": "arn:aws:iam::123456789012:role/service-role/export-snap-S3-role",  
  "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/abcd0000-7bfd-4594-af38-  
aabbccddeeff",  
  "Status": "CANCELING",  
  "PercentProgress": 0,  
  "TotalExtractedDataInGB": 0  
}
```

For more information, see [Canceling a snapshot export task](#) in the *Amazon RDS User Guide* or [Canceling a snapshot export task](#) in the *Amazon Aurora User Guide*.

- For API details, see [CancelExportTask](#) in *AWS CLI Command Reference*.

copy-db-cluster-parameter-group

The following code example shows how to use `copy-db-cluster-parameter-group`.

AWS CLI

To copy a DB cluster parameter group

The following `copy-db-cluster-parameter-group` example makes a copy of a DB cluster parameter group.

```
aws rds copy-db-cluster-parameter-group \  
  --source-db-cluster-parameter-group-identifier mydbclusterpg \  
  --target-db-cluster-parameter-group-identifier mydbclusterpg
```

```
--target-db-cluster-parameter-group-identifier mydbclusterpgcopy \  
--target-db-cluster-parameter-group-description "Copy of mydbclusterpg parameter  
group"
```

Output:

```
{  
  "DBClusterParameterGroup": {  
    "DBClusterParameterGroupName": "mydbclusterpgcopy",  
    "DBClusterParameterGroupArn": "arn:aws:rds:us-east-1:123456789012:cluster-  
pg:mydbclusterpgcopy",  
    "DBParameterGroupFamily": "aurora-mysql5.7",  
    "Description": "Copy of mydbclusterpg parameter group"  
  }  
}
```

For more information, see [Copying a DB Cluster Parameter Group](#) in the *Amazon Aurora Users Guide*.

- For API details, see [CopyDbClusterParameterGroup](#) in *AWS CLI Command Reference*.

copy-db-cluster-snapshot

The following code example shows how to use `copy-db-cluster-snapshot`.

AWS CLI

To copy a DB cluster snapshot

The following `copy-db-cluster-snapshot` example creates a copy of a DB cluster snapshot, including its tags.

```
aws rds copy-db-cluster-snapshot \  
  --source-db-cluster-snapshot-identifier arn:aws:rds:us-  
east-1:123456789012:cluster-snapshot:rds:myaurora-2019-06-04-09-16  
  --target-db-cluster-snapshot-identifier myclustersnapshotcopy \  
  --copy-tags
```

Output:

```
{
```

```

"DBClusterSnapshot": {
  "AvailabilityZones": [
    "us-east-1a",
    "us-east-1b",
    "us-east-1e"
  ],
  "DBClusterSnapshotIdentifier": "myclustersnapshotcopy",
  "DBClusterIdentifier": "myaurora",
  "SnapshotCreateTime": "2019-06-04T09:16:42.649Z",
  "Engine": "aurora-mysql",
  "AllocatedStorage": 0,
  "Status": "available",
  "Port": 0,
  "VpcId": "vpc-6594f31c",
  "ClusterCreateTime": "2019-04-15T14:18:42.785Z",
  "MasterUsername": "myadmin",
  "EngineVersion": "5.7.mysql_aurora.2.04.2",
  "LicenseModel": "aurora-mysql",
  "SnapshotType": "manual",
  "PercentProgress": 100,
  "StorageEncrypted": true,
  "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/AKIAIOSFODNN7EXAMPLE",
  "DBClusterSnapshotArn": "arn:aws:rds:us-east-1:123456789012:cluster-
snapshot:myclustersnapshotcopy",
  "IAMDatabaseAuthenticationEnabled": false
}
}

```

For more information, see [Copying a Snapshot](#) in the *Amazon Aurora User Guide*.

- For API details, see [CopyDbClusterSnapshot](#) in *AWS CLI Command Reference*.

copy-db-parameter-group

The following code example shows how to use copy-db-parameter-group.

AWS CLI

To copy a DB cluster parameter group

The following copy-db-parameter-group example makes a copy of a DB parameter group.

```
aws rds copy-db-parameter-group \
```

```
--source-db-parameter-group-identifier mydbpg \  
--target-db-parameter-group-identifier mydbpgcopy \  
--target-db-parameter-group-description "Copy of mydbpg parameter group"
```

Output:

```
{  
  "DBParameterGroup": {  
    "DBParameterGroupName": "mydbpgcopy",  
    "DBParameterGroupArn": "arn:aws:rds:us-east-1:814387698303:pg:mydbpgcopy",  
    "DBParameterGroupFamily": "mysql5.7",  
    "Description": "Copy of mydbpg parameter group"  
  }  
}
```

For more information, see [Copying a DB Parameter Group](#) in the *Amazon RDS Users Guide*.

- For API details, see [CopyDbParameterGroup](#) in *AWS CLI Command Reference*.

copy-db-snapshot

The following code example shows how to use copy-db-snapshot.

AWS CLI

To copy a DB snapshot

The following copy-db-snapshot example creates a copy of a DB snapshot.

```
aws rds copy-db-snapshot \  
  --source-db-snapshot-identifier rds:database-mysql-2019-06-06-08-38  
  --target-db-snapshot-identifier mydbsnapshotcopy
```

Output:

```
{  
  "DBSnapshot": {  
    "VpcId": "vpc-6594f31c",  
    "Status": "creating",  
    "Encrypted": true,  
    "SourceDBSnapshotIdentifier": "arn:aws:rds:us-east-1:123456789012:snapshot:rds:database-mysql-2019-06-06-08-38",  
  }  
}
```

```

    "MasterUsername": "admin",
    "Iops": 1000,
    "Port": 3306,
    "LicenseModel": "general-public-license",
    "DBSnapshotArn": "arn:aws:rds:us-
east-1:123456789012:snapshot:mysnapshotcopy",
    "EngineVersion": "5.6.40",
    "OptionGroupName": "default:mysql-5-6",
    "ProcessorFeatures": [],
    "Engine": "mysql",
    "StorageType": "io1",
    "DbiResourceId": "db-ZI7UJ5BLKMBYFGX7FDENCKADC4",
    "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/AKIAIOSFODNN7EXAMPLE",
    "SnapshotType": "manual",
    "IAMDatabaseAuthenticationEnabled": false,
    "SourceRegion": "us-east-1",
    "DBInstanceIdentifier": "database-mysql",
    "InstanceCreateTime": "2019-04-30T15:45:53.663Z",
    "AvailabilityZone": "us-east-1f",
    "PercentProgress": 0,
    "AllocatedStorage": 100,
    "DBSnapshotIdentifier": "mysnapshotcopy"
  }
}

```

For more information, see [Copying a Snapshot](#) in the *Amazon RDS User Guide*.

- For API details, see [CopyDbSnapshot](#) in *AWS CLI Command Reference*.

copy-option-group

The following code example shows how to use `copy-option-group`.

AWS CLI

To copy an option group

The following `copy-option-group` example makes a copy of an option group.

```

aws rds copy-option-group \
  --source-option-group-identifier myoptiongroup \
  --target-option-group-identifier new-option-group \
  --target-option-group-description "My option group copy"

```

Output:

```
{
  "OptionGroup": {
    "Options": [],
    "OptionGroupName": "new-option-group",
    "MajorEngineVersion": "11.2",
    "OptionGroupDescription": "My option group copy",
    "AllowsVpcAndNonVpcInstanceMemberships": true,
    "EngineName": "oracle-ee",
    "OptionGroupArn": "arn:aws:rds:us-east-1:123456789012:og:new-option-group"
  }
}
```

For more information, see [Making a Copy of an Option Group](#) in the *Amazon RDS User Guide*.

- For API details, see [CopyOptionGroup](#) in *AWS CLI Command Reference*.

create-blue-green-deployment

The following code example shows how to use `create-blue-green-deployment`.

AWS CLI**Example 1: To create a blue/green deployment for an RDS for MySQL DB instance**

The following `create-blue-green-deployment` example creates a blue/green deployment for a MySQL DB instance.

```
aws rds create-blue-green-deployment \
  --blue-green-deployment-name bgd-cli-test-instance \
  --source arn:aws:rds:us-east-1:123456789012:db:my-db-instance \
  --target-engine-version 8.0 \
  --target-db-parameter-group-name mysql-80-group
```

Output:

```
{
  "BlueGreenDeployment": {
    "BlueGreenDeploymentIdentifier": "bgd-v53303651eexfake",
    "BlueGreenDeploymentName": "bgd-cli-test-instance",
    "Source": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance",
  }
}
```

```

    "SwitchoverDetails": [
      {
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance"
      },
      {
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-1"
      },
      {
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-2"
      },
      {
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-3"
      }
    ],
    "Tasks": [
      {
        "Name": "CREATING_READ_REPLICA_OF_SOURCE",
        "Status": "PENDING"
      },
      {
        "Name": "DB_ENGINE_VERSION_UPGRADE",
        "Status": "PENDING"
      },
      {
        "Name": "CONFIGURE_BACKUPS",
        "Status": "PENDING"
      },
      {
        "Name": "CREATING_TOPOLOGY_OF_SOURCE",
        "Status": "PENDING"
      }
    ],
    "Status": "PROVISIONING",
    "CreateTime": "2022-02-25T21:18:51.183000+00:00"
  }
}

```

For more information, see [Creating a blue/green deployment](#) in the *Amazon RDS User Guide*.

Example 2: To create a blue/green deployment for an Aurora MySQL DB cluster

The following `create-blue-green-deployment` example creates a blue/green deployment for an Aurora MySQL DB cluster.

```
aws rds create-blue-green-deployment \  
  --blue-green-deployment-name my-blue-green-deployment \  
  --source arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-cluster \  
  --target-engine-version 8.0 \  
  --target-db-cluster-parameter-group-name ams-80-binlog-enabled \  
  --target-db-parameter-group-name mysql-80-cluster-group
```

Output:

```
{  
  "BlueGreenDeployment": {  
    "BlueGreenDeploymentIdentifier": "bgd-wi89nwzglccsfake",  
    "BlueGreenDeploymentName": "my-blue-green-deployment",  
    "Source": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-  
cluster",  
    "SwitchoverDetails": [  
      {  
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-  
mysql-cluster",  
        "Status": "PROVISIONING"  
      },  
      {  
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-mysql-  
cluster-1",  
        "Status": "PROVISIONING"  
      },  
      {  
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-mysql-  
cluster-2",  
        "Status": "PROVISIONING"  
      },  
      {  
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-mysql-  
cluster-3",  
        "Status": "PROVISIONING"  
      },  
      {  
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-endpoint:my-  
excluded-member-endpoint",  
        "Status": "PROVISIONING"  
      }  
    ]  
  }  
}
```

```
    },
    {
      "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-endpoint:my-
reader-endpoint",
      "Status": "PROVISIONING"
    }
  ],
  "Tasks": [
    {
      "Name": "CREATING_READ_REPLICA_OF_SOURCE",
      "Status": "PENDING"
    },
    {
      "Name": "DB_ENGINE_VERSION_UPGRADE",
      "Status": "PENDING"
    },
    {
      "Name": "CREATE_DB_INSTANCES_FOR_CLUSTER",
      "Status": "PENDING"
    },
    {
      "Name": "CREATE_CUSTOM_ENDPOINTS",
      "Status": "PENDING"
    }
  ],
  "Status": "PROVISIONING",
  "CreateTime": "2022-02-25T21:12:00.288000+00:00"
}
```

For more information, see [Creating a blue/green deployment](#) in the *Amazon Aurora User Guide*.

- For API details, see [CreateBlueGreenDeployment](#) in *AWS CLI Command Reference*.

create-db-cluster-endpoint

The following code example shows how to use `create-db-cluster-endpoint`.

AWS CLI

To create a custom DB cluster endpoint

The following `create-db-cluster-endpoint` example creates a custom DB cluster endpoint and associate it with the specified Aurora DB cluster.

```
aws rds create-db-cluster-endpoint \  
  --db-cluster-endpoint-identifier mycustomendpoint \  
  --endpoint-type reader \  
  --db-cluster-identifier mydbcluster \  
  --static-members dbinstance1 dbinstance2
```

Output:

```
{  
  "DBClusterEndpointIdentifier": "mycustomendpoint",  
  "DBClusterIdentifier": "mydbcluster",  
  "DBClusterEndpointResourceIdentifier": "cluster-endpoint-ANPAJ4AE5446DAEXAMPLE",  
  "Endpoint": "mycustomendpoint.cluster-custom-cnpxample.us-  
east-1.rds.amazonaws.com",  
  "Status": "creating",  
  "EndpointType": "CUSTOM",  
  "CustomEndpointType": "READER",  
  "StaticMembers": [  
    "dbinstance1",  
    "dbinstance2"  
  ],  
  "ExcludedMembers": [],  
  "DBClusterEndpointArn": "arn:aws:rds:us-east-1:123456789012:cluster-  
endpoint:mycustomendpoint"  
}
```

For more information, see [Amazon Aurora Connection Management](#) in the *Amazon Aurora User Guide*.

- For API details, see [CreateDbClusterEndpoint](#) in *AWS CLI Command Reference*.

create-db-cluster-parameter-group

The following code example shows how to use `create-db-cluster-parameter-group`.

AWS CLI

To create a DB cluster parameter group

The following `create-db-cluster-parameter-group` example creates a DB cluster parameter group.

```
aws rds create-db-cluster-parameter-group \  
  --db-cluster-parameter-group-name mydbclusterparametergroup \  
  --db-parameter-group-family aurora5.6 \  
  --description "My new cluster parameter group"
```

Output:

```
{  
  "DBClusterParameterGroup": {  
    "DBClusterParameterGroupName": "mydbclusterparametergroup",  
    "DBParameterGroupFamily": "aurora5.6",  
    "Description": "My new cluster parameter group",  
    "DBClusterParameterGroupArn": "arn:aws:rds:us-east-1:123456789012:cluster-  
pg:mydbclusterparametergroup"  
  }  
}
```

For more information, see [Creating a DB Cluster Parameter Group](#) in the *Amazon Aurora User Guide*.

- For API details, see [CreateDbClusterParameterGroup](#) in *AWS CLI Command Reference*.

create-db-cluster-snapshot

The following code example shows how to use `create-db-cluster-snapshot`.

AWS CLI

To create a DB cluster snapshot

The following `create-db-cluster-snapshot` example creates a DB cluster snapshot.

```
aws rds create-db-cluster-snapshot \  
  --db-cluster-identifier mydbcluster \  
  --db-cluster-snapshot-identifier mydbclustersnapshot
```

Output:

```
{
```

```

"DBClusterSnapshot": {
  "AvailabilityZones": [
    "us-east-1a",
    "us-east-1b",
    "us-east-1e"
  ],
  "DBClusterSnapshotIdentifier": "mydbclustersnapshot",
  "DBClusterIdentifier": "mydbcluster",
  "SnapshotCreateTime": "2019-06-18T21:21:00.469Z",
  "Engine": "aurora-mysql",
  "AllocatedStorage": 1,
  "Status": "creating",
  "Port": 0,
  "VpcId": "vpc-6594f31c",
  "ClusterCreateTime": "2019-04-15T14:18:42.785Z",
  "MasterUsername": "myadmin",
  "EngineVersion": "5.7.mysql_aurora.2.04.2",
  "LicenseModel": "aurora-mysql",
  "SnapshotType": "manual",
  "PercentProgress": 0,
  "StorageEncrypted": true,
  "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/AKIAIOSFODNN7EXAMPLE",
  "DBClusterSnapshotArn": "arn:aws:rds:us-east-1:123456789012:cluster-
snapshot:mydbclustersnapshot",
  "IAMDatabaseAuthenticationEnabled": false
}
}

```

For more information, see [Creating a DB Cluster Snapshot](#) in the *Amazon Aurora User Guide*.

- For API details, see [CreateDbClusterSnapshot](#) in *AWS CLI Command Reference*.

create-db-cluster

The following code example shows how to use `create-db-cluster`.

AWS CLI

Example 1: To create a MySQL 5.7--compatible DB cluster

The following `create-db-cluster` example creates a MySQL 5.7-compatible DB cluster using the default engine version. Replace the sample password `secret99` with a secure password. When you use the console to create a DB cluster, Amazon RDS automatically creates the writer

DB instance for your DB cluster. However, when you use the AWS CLI to create a DB cluster, you must explicitly create the writer DB instance for your DB cluster using the `create-db-instance` AWS CLI command.

```
aws rds create-db-cluster \  
  --db-cluster-identifier sample-cluster \  
  --engine aurora-mysql \  
  --engine-version 5.7 \  
  --master-username admin \  
  --master-user-password secret99 \  
  --db-subnet-group-name default \  
  --vpc-security-group-ids sg-0b9130572daf3dc16
```

Output:

```
{  
  "DBCluster": {  
    "DBSubnetGroup": "default",  
    "VpcSecurityGroups": [  
      {  
        "VpcSecurityGroupId": "sg-0b9130572daf3dc16",  
        "Status": "active"  
      }  
    ],  
    "AllocatedStorage": 1,  
    "AssociatedRoles": [],  
    "PreferredBackupWindow": "09:12-09:42",  
    "ClusterCreateTime": "2023-02-27T23:21:33.048Z",  
    "DeletionProtection": false,  
    "IAMDatabaseAuthenticationEnabled": false,  
    "ReadReplicaIdentifiers": [],  
    "EngineMode": "provisioned",  
    "Engine": "aurora-mysql",  
    "StorageEncrypted": false,  
    "MultiAZ": false,  
    "PreferredMaintenanceWindow": "mon:04:31-mon:05:01",  
    "HttpEndpointEnabled": false,  
    "BackupRetentionPeriod": 1,  
    "DbClusterResourceId": "cluster-ANPAJ4AE5446DAEXAMPLE",  
    "DBClusterIdentifier": "sample-cluster",  
    "AvailabilityZones": [  
      "us-east-1a",  
      "us-east-1b",
```

```

        "us-east-1e"
    ],
    "MasterUsername": "master",
    "EngineVersion": "5.7.mysql_aurora.2.11.1",
    "DBClusterArn": "arn:aws:rds:us-east-1:123456789012:cluster:sample-cluster",
    "DBClusterMembers": [],
    "Port": 3306,
    "Status": "creating",
    "Endpoint": "sample-cluster.cluster-cnpxexample.us-east-1.rds.amazonaws.com",
    "DBClusterParameterGroup": "default.aurora-mysql5.7",
    "HostedZoneId": "Z2R2ITUGPM61AM",
    "ReaderEndpoint": "sample-cluster.cluster-ro-cnpxexample.us-
east-1.rds.amazonaws.com",
    "CopyTagsToSnapshot": false
}
}

```

Example 2: To create a PostgreSQL--compatible DB cluster

The following `create-db-cluster` example creates a PostgreSQL-compatible DB cluster using the default engine version. Replace the example password `secret99` with a secure password. When you use the console to create a DB cluster, Amazon RDS automatically creates the writer DB instance for your DB cluster. However, when you use the AWS CLI to create a DB cluster, you must explicitly create the writer DB instance for your DB cluster using the `create-db-instance` AWS CLI command.

```

aws rds create-db-cluster \
  --db-cluster-identifier sample-pg-cluster \
  --engine aurora-postgresql \
  --master-username master \
  --master-user-password secret99 \
  --db-subnet-group-name default \
  --vpc-security-group-ids sg-0b9130572daf3dc16

```

Output:

```

{
  "DBCluster": {
    "Endpoint": "sample-pg-cluster.cluster-cnpxexample.us-
east-1.rds.amazonaws.com",
    "HttpEndpointEnabled": false,
    "DBClusterMembers": [],

```

```
    "EngineMode": "provisioned",
    "CopyTagsToSnapshot": false,
    "HostedZoneId": "Z2R2ITUGPM61AM",
    "IAMDatabaseAuthenticationEnabled": false,
    "AllocatedStorage": 1,
    "VpcSecurityGroups": [
      {
        "VpcSecurityGroupId": "sg-0b9130572daf3dc16",
        "Status": "active"
      }
    ],
    "DeletionProtection": false,
    "StorageEncrypted": false,
    "BackupRetentionPeriod": 1,
    "PreferredBackupWindow": "09:56-10:26",
    "ClusterCreateTime": "2023-02-27T23:26:08.371Z",
    "DBClusterParameterGroup": "default.aurora-postgresql13",
    "EngineVersion": "13.7",
    "Engine": "aurora-postgresql",
    "Status": "creating",
    "DBClusterIdentifier": "sample-pg-cluster",
    "MultiAZ": false,
    "Port": 5432,
    "DBClusterArn": "arn:aws:rds:us-east-1:123456789012:cluster:sample-pg-
cluster",
    "AssociatedRoles": [],
    "DbClusterResourceId": "cluster-ANPAJ4AE5446DAEXAMPLE",
    "PreferredMaintenanceWindow": "wed:03:33-wed:04:03",
    "ReaderEndpoint": "sample-pg-cluster.cluster-ro-cnpxexample.us-
east-1.rds.amazonaws.com",
    "MasterUsername": "master",
    "AvailabilityZones": [
      "us-east-1a",
      "us-east-1b",
      "us-east-1c"
    ],
    "ReadReplicaIdentifiers": [],
    "DBSubnetGroup": "default"
  }
}
```

For more information, see [Creating an Amazon Aurora DB cluster](#) in the *Amazon Aurora User Guide*.

- For API details, see [CreateDbCluster](#) in *AWS CLI Command Reference*.

create-db-instance-read-replica

The following code example shows how to use `create-db-instance-read-replica`.

AWS CLI

To create a DB instance read replica

This example creates a read replica of an existing DB instance named `test-instance`. The read replica is named `test-instance-repl`.

```
aws rds create-db-instance-read-replica \  
  --db-instance-identifier test-instance-repl \  
  --source-db-instance-identifier test-instance
```

Output:

```
{  
  "DBInstance": {  
    "IAMDatabaseAuthenticationEnabled": false,  
    "MonitoringInterval": 0,  
    "DBInstanceArn": "arn:aws:rds:us-east-1:123456789012:db:test-instance-repl",  
    "ReadReplicaSourceDBInstanceIdentifier": "test-instance",  
    "DBInstanceIdentifier": "test-instance-repl",  
    ...some output truncated...  
  }  
}
```

- For API details, see [CreateDbInstanceReadReplica](#) in *AWS CLI Command Reference*.

create-db-instance

The following code example shows how to use `create-db-instance`.

AWS CLI

To create a DB instance

The following `create-db-instance` example uses the required options to launch a new DB instance.

```
aws rds create-db-instance \  
  --db-instance-identifier test-mysql-instance \  
  --db-instance-class db.t3.micro \  
  --engine mysql \  
  --master-username admin \  
  --master-user-password secret99 \  
  --allocated-storage 20
```

Output:

```
{  
  "DBInstance": {  
    "DBInstanceIdentifier": "test-mysql-instance",  
    "DBInstanceClass": "db.t3.micro",  
    "Engine": "mysql",  
    "DBInstanceStatus": "creating",  
    "MasterUsername": "admin",  
    "AllocatedStorage": 20,  
    "PreferredBackupWindow": "12:55-13:25",  
    "BackupRetentionPeriod": 1,  
    "DBSecurityGroups": [],  
    "VpcSecurityGroups": [  
      {  
        "VpcSecurityGroupId": "sg-12345abc",  
        "Status": "active"  
      }  
    ],  
    "DBParameterGroups": [  
      {  
        "DBParameterGroupName": "default.mysql5.7",  
        "ParameterApplyStatus": "in-sync"  
      }  
    ],  
    "DBSubnetGroup": {  
      "DBSubnetGroupName": "default",  
      "DBSubnetGroupDescription": "default",  
      "VpcId": "vpc-2ff2ff2f",  
      "SubnetGroupStatus": "Complete",  
      "Subnets": [  
        {  
          "SubnetIdentifier": "subnet-#####",  
          "SubnetAvailabilityZone": {  
            "Name": "us-west-2c"  
          }  
        }  
      ]  
    }  
  }  
}
```

```
    },
    "SubnetStatus": "Active"
  },
  {
    "SubnetIdentifier": "subnet-#####",
    "SubnetAvailabilityZone": {
      "Name": "us-west-2d"
    },
    "SubnetStatus": "Active"
  },
  {
    "SubnetIdentifier": "subnet-#####",
    "SubnetAvailabilityZone": {
      "Name": "us-west-2a"
    },
    "SubnetStatus": "Active"
  },
  {
    "SubnetIdentifier": "subnet-#####",
    "SubnetAvailabilityZone": {
      "Name": "us-west-2b"
    },
    "SubnetStatus": "Active"
  }
]
},
"PreferredMaintenanceWindow": "sun:08:07-sun:08:37",
"PendingModifiedValues": {
  "MasterUserPassword": "*****"
},
"MultiAZ": false,
"EngineVersion": "5.7.22",
"AutoMinorVersionUpgrade": true,
"ReadReplicaDBInstanceIdentifiers": [],
"LicenseModel": "general-public-license",
"OptionGroupMemberships": [
  {
    "OptionGroupName": "default:mysql-5-7",
    "Status": "in-sync"
  }
],
"PubliclyAccessible": true,
"StorageType": "gp2",
"DbInstancePort": 0,
```

```
"StorageEncrypted": false,
"DbiResourceId": "db-5555EXAMPLE44444444EXAMPLE",
"CACertificateIdentifier": "rds-ca-2019",
"DomainMemberships": [],
"CopyTagsToSnapshot": false,
"MonitoringInterval": 0,
"DBInstanceArn": "arn:aws:rds:us-west-2:123456789012:db:test-mysql-
instance",
"IAMDatabaseAuthenticationEnabled": false,
"PerformanceInsightsEnabled": false,
"DeletionProtection": false,
"AssociatedRoles": []
}
}
```

For more information, see [Creating an Amazon RDS DB Instance](#) in the *Amazon RDS User Guide*.

- For API details, see [CreateDBInstance](#) in *AWS CLI Command Reference*.

create-db-parameter-group

The following code example shows how to use `create-db-parameter-group`.

AWS CLI

To create a DB parameter group

The following `create-db-parameter-group` example creates a DB parameter group.

```
aws rds create-db-parameter-group \
  --db-parameter-group-name mydbparametergroup \
  --db-parameter-group-family MySQL5.6 \
  --description "My new parameter group"
```

Output:

```
{
  "DBParameterGroup": {
    "DBParameterGroupName": "mydbparametergroup",
    "DBParameterGroupFamily": "mysql5.6",
    "Description": "My new parameter group",
    "DBParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:pg:mydbparametergroup"
  }
}
```

```
}  
}
```

For more information, see [Creating a DB Parameter Group](#) in the *Amazon RDS User Guide*.

- For API details, see [CreateDBParameterGroup](#) in *AWS CLI Command Reference*.

create-db-proxy-endpoint

The following code example shows how to use `create-db-proxy-endpoint`.

AWS CLI

To create a DB proxy endpoint for an RDS database

The following `create-db-proxy-endpoint` example creates a DB proxy endpoint.

```
aws rds create-db-proxy-endpoint \  
  --db-proxy-name proxyExample \  
  --db-proxy-endpoint-name "proxyep1" \  
  --vpc-subnet-ids subnetgroup1 subnetgroup2
```

Output:

```
{  
  "DBProxyEndpoint": {  
    "DBProxyEndpointName": "proxyep1",  
    "DBProxyEndpointArn": "arn:aws:rds:us-east-1:123456789012:db-proxy-  
endpoint:prx-endpoint-0123a01b12345c0ab",  
    "DBProxyName": "proxyExample",  
    "Status": "creating",  
    "VpcId": "vpc-1234567",  
    "VpcSecurityGroupIds": [  
      "sg-1234",  
      "sg-5678"  
    ],  
    "VpcSubnetIds": [  
      "subnetgroup1",  
      "subnetgroup2"  
    ],  
    "Endpoint": "proxyep1.endpoint.proxy-ab0cd1efghij.us-  
east-1.rds.amazonaws.com",  
    "CreatedDate": "2023-04-05T16:09:33.452000+00:00",
```

```

        "TargetRole": "READ_WRITE",
        "IsDefault": false
    }
}

```

For more information, see [Creating a proxy endpoint](#) in the *Amazon RDS User Guide* and [Creating a proxy endpoint](#) in the *Amazon Aurora User Guide*.

- For API details, see [CreateDbProxyEndpoint](#) in *AWS CLI Command Reference*.

create-db-proxy

The following code example shows how to use create-db-proxy.

AWS CLI

To create a DB proxy for an RDS database

The following create-db-proxy example creates a DB proxy.

```

aws rds create-db-proxy \
  --db-proxy-name proxyExample \
  --engine-family MYSQL \
  --auth
  Description="proxydescription1",AuthScheme="SECRETS",SecretArn="arn:aws:secretsmanager:us-
west-2:123456789123:secret:secretName-1234f",IAMAuth="DISABLED",ClientPasswordAuthType="MYSQ
  \
  --role-arn arn:aws:iam::123456789123:role/ProxyRole \
  --vpc-subnet-ids subnetgroup1 subnetgroup2

```

Output:

```

{
  "DBProxy": {
    "DBProxyName": "proxyExample",
    "DBProxyArn": "arn:aws:rds:us-east-1:123456789012:db-
proxy:prx-0123a01b12345c0ab",
    "EngineFamily": "MYSQL",
    "VpcId": "vpc-1234567",
    "VpcSecuritytGroupIds": [
      "sg-1234",
      "sg-5678",
      "sg-9101"
    ]
  }
}

```

```

    ],
    "VpcSubnetIds": [
        "subnetgroup1",
        "subnetgroup2"
    ],
    "Auth": "[
        {
            "Description": "proxydescription1",
            "AuthScheme": "SECRETS",
            "SecretArn": "arn:aws:secretsmanager:us-
west-2:123456789123:secret:proxysecret1-Abcd1e",
            "IAMAuth": "DISABLED"
        }
    ]",
    "RoleArn": "arn:aws:iam::12345678912:role/ProxyRole",
    "Endpoint": "proxyExample.proxy-ab0cd1efghij.us-east-1.rds.amazonaws.com",
    "RequireTLS": false,
    "IdleClientTimeout": 1800,
    "DebuggingLogging": false,
    "CreateDate": "2023-04-05T16:09:33.452000+00:00",
    "UpdateDate": "2023-04-13T01:49:38.568000+00:00"
}
}

```

For more information, see [Creating an RDS Proxy](#) in the *Amazon RDS User Guide* and [Creating an RDS Proxy](#) in the *Amazon Aurora User Guide*.

- For API details, see [CreateDbProxy](#) in *AWS CLI Command Reference*.

create-db-security-group

The following code example shows how to use create-db-security-group.

AWS CLI

To create an Amazon RDS DB security group

The following create-db-security-group command creates a new Amazon RDS DB security group:

```
aws rds create-db-security-group --db-security-group-name mysecgroup --db-security-
group-description "My Test Security Group"
```

In the example, the new DB security group is named `mysecgroup` and has a description.

Output:

```
{
  "DBSecurityGroup": {
    "OwnerId": "123456789012",
    "DBSecurityGroupName": "mysecgroup",
    "DBSecurityGroupDescription": "My Test Security Group",
    "VpcId": "vpc-a1b2c3d4",
    "EC2SecurityGroups": [],
    "IPRanges": [],
    "DBSecurityGroupArn": "arn:aws:rds:us-west-2:123456789012:secgrp:mysecgroup"
  }
}
```

- For API details, see [CreateDbSecurityGroup](#) in *AWS CLI Command Reference*.

create-db-shard-group

The following code example shows how to use `create-db-shard-group`.

AWS CLI

Example 1: To create an Aurora PostgreSQL primary DB cluster

The following `create-db-cluster` example creates an Aurora PostgreSQL SQL primary DB cluster that's compatible with Aurora Serverless v2 and Aurora Limitless Database.

```
aws rds create-db-cluster \  
  --db-cluster-identifier my-sv2-cluster \  
  --engine aurora-postgresql \  
  --engine-version 15.2-limitless \  
  --storage-type aurora-iopt1 \  
  --serverless-v2-scaling-configuration MinCapacity=2,MaxCapacity=16 \  
  --enable-limitless-database \  
  --master-username myuser \  
  --master-user-password mypassword \  
  --enable-cloudwatch-logs-exports postgresql
```

Output:


```
{
  "DBCluster": {
    "AllocatedStorage": 1,
    "AvailabilityZones": [
      "us-east-2b",
      "us-east-2c",
      "us-east-2a"
    ],
    "BackupRetentionPeriod": 1,
    "DBClusterIdentifier": "my-sv2-cluster",
    "DBClusterParameterGroup": "default.aurora-postgresql15",
    "DBSubnetGroup": "default",
    "Status": "creating",
    "Endpoint": "my-sv2-cluster.cluster-cekyexample.us-
east-2.rds.amazonaws.com",
    "ReaderEndpoint": "my-sv2-cluster.cluster-ro-cekyexample.us-
east-2.rds.amazonaws.com",
    "MultiAZ": false,
    "Engine": "aurora-postgresql",
    "EngineVersion": "15.2-limitless",
    "Port": 5432,
    "MasterUsername": "myuser",
    "PreferredBackupWindow": "06:05-06:35",
    "PreferredMaintenanceWindow": "mon:08:25-mon:08:55",
    "ReadReplicaIdentifiers": [],
    "DBClusterMembers": [],
    "VpcSecurityGroups": [
      {
        "VpcSecurityGroupId": "sg-#####",
        "Status": "active"
      }
    ],
    "HostedZoneId": "Z2XHWR1EXAMPLE",
    "StorageEncrypted": false,
    "DbClusterResourceId": "cluster-XYEDT6ML6FHIXH4Q2J1EXAMPLE",
    "DBClusterArn": "arn:aws:rds:us-east-2:123456789012:cluster:my-sv2-cluster",
    "AssociatedRoles": [],
    "IAMDatabaseAuthenticationEnabled": false,
    "ClusterCreateTime": "2024-02-19T16:24:07.771000+00:00",
    "EnabledCloudwatchLogsExports": [
      "postgresql"
    ],
    "EngineMode": "provisioned",
```

```

    "DeletionProtection": false,
    "HttpEndpointEnabled": false,
    "CopyTagsToSnapshot": false,
    "CrossAccountClone": false,
    "DomainMemberships": [],
    "TagList": [],
    "StorageType": "aurora-iopt1",
    "AutoMinorVersionUpgrade": true,
    "ServerlessV2ScalingConfiguration": {
      "MinCapacity": 2.0,
      "MaxCapacity": 16.0
    },
    "NetworkType": "IPV4",
    "IOOptimizedNextAllowedModificationTime":
"2024-03-21T16:24:07.781000+00:00",
    "LimitlessDatabase": {
      "Status": "not-in-use",
      "MinRequiredACU": 96.0
    }
  }
}

```

Example 2: To create the primary (writer) DB instance

The following `create-db-instance` example creates an Aurora Serverless v2 primary (writer) DB instance. When you use the console to create a DB cluster, Amazon RDS automatically creates the writer DB instance for your DB cluster. However, when you use the AWS CLI to create a DB cluster, you must explicitly create the writer DB instance for your DB cluster using the `create-db-instance` AWS CLI command.

```

aws rds create-db-instance \
  --db-instance-identifier my-sv2-instance \
  --db-cluster-identifier my-sv2-cluster \
  --engine aurora-postgresql \
  --db-instance-class db.serverless

```

Output:

```

{
  "DBInstance": {
    "DBInstanceIdentifier": "my-sv2-instance",
    "DBInstanceClass": "db.serverless",

```

```
"Engine": "aurora-postgresql",
"DBInstanceStatus": "creating",
"MasterUsername": "myuser",
"AllocatedStorage": 1,
"PreferredBackupWindow": "06:05-06:35",
"BackupRetentionPeriod": 1,
"DBSecurityGroups": [],
"VpcSecurityGroups": [
  {
    "VpcSecurityGroupId": "sg-#####",
    "Status": "active"
  }
],
"DBParameterGroups": [
  {
    "DBParameterGroupName": "default.aurora-postgresql15",
    "ParameterApplyStatus": "in-sync"
  }
],
"DBSubnetGroup": {
  "DBSubnetGroupName": "default",
  "DBSubnetGroupDescription": "default",
  "VpcId": "vpc-#####",
  "SubnetGroupStatus": "Complete",
  "Subnets": [
    {
      "SubnetIdentifier": "subnet-#####",
      "SubnetAvailabilityZone": {
        "Name": "us-east-2c"
      },
      "SubnetOutpost": {},
      "SubnetStatus": "Active"
    },
    {
      "SubnetIdentifier": "subnet-#####",
      "SubnetAvailabilityZone": {
        "Name": "us-east-2a"
      },
      "SubnetOutpost": {},
      "SubnetStatus": "Active"
    },
    {
      "SubnetIdentifier": "subnet-#####",
      "SubnetAvailabilityZone": {
```

```
        "Name": "us-east-2b"
      },
      "SubnetOutpost": {},
      "SubnetStatus": "Active"
    }
  ]
},
"PreferredMaintenanceWindow": "fri:09:01-fri:09:31",
"PendingModifiedValues": {
  "PendingCloudwatchLogsExports": {
    "LogTypesToEnable": [
      "postgresql"
    ]
  }
},
"MultiAZ": false,
"EngineVersion": "15.2-limitless",
"AutoMinorVersionUpgrade": true,
"ReadReplicaDBInstanceIdentifiers": [],
"LicenseModel": "postgresql-license",
"OptionGroupMemberships": [
  {
    "OptionGroupName": "default:aurora-postgresql-15",
    "Status": "in-sync"
  }
],
"PubliclyAccessible": false,
"StorageType": "aurora-iopt1",
"DbInstancePort": 0,
"DBClusterIdentifier": "my-sv2-cluster",
"StorageEncrypted": false,
"DbiResourceId": "db-BIQTE3B3K3RM7M74SK5EXAMPLE",
"CACertificateIdentifier": "rds-ca-rsa2048-g1",
"DomainMemberships": [],
"CopyTagsToSnapshot": false,
"MonitoringInterval": 0,
"PromotionTier": 1,
"DBInstanceArn": "arn:aws:rds:us-east-2:123456789012:db:my-sv2-instance",
"IAMDatabaseAuthenticationEnabled": false,
"PerformanceInsightsEnabled": false,
"DeletionProtection": false,
"AssociatedRoles": [],
"TagList": [],
"CustomerOwnedIpEnabled": false,
```

```

    "BackupTarget": "region",
    "NetworkType": "IPV4",
    "StorageThroughput": 0,
    "CertificateDetails": {
      "CAIdentifier": "rds-ca-rsa2048-g1"
    },
    "DedicatedLogVolume": false
  }
}

```

Example 3: To create the DB shard group

The following `create-db-shard-group` example creates a DB shard group in your Aurora PostgreSQL primary DB cluster.

```

aws rds create-db-shard-group \
  --db-shard-group-identifier my-db-shard-group \
  --db-cluster-identifier my-sv2-cluster \
  --max-acu 768

```

Output:

```

{
  "DBShardGroupResourceId": "shardgroup-a6e3a0226aa243e2ac6c7a1234567890",
  "DBShardGroupIdentifier": "my-db-shard-group",
  "DBClusterIdentifier": "my-sv2-cluster",
  "MaxACU": 768.0,
  "ComputeRedundancy": 0,
  "Status": "creating",
  "PubliclyAccessible": false,
  "Endpoint": "my-sv2-cluster.limitless-cekyceexample.us-east-2.rds.amazonaws.com"
}

```

For more information, see [Using Aurora Serverless v2](#) in the *Amazon Aurora User Guide*.

- For API details, see [CreateDbShardGroup](#) in *AWS CLI Command Reference*.

create-db-snapshot

The following code example shows how to use `create-db-snapshot`.

AWS CLI

To create a DB snapshot

The following `create-db-snapshot` example creates a DB snapshot.

```
aws rds create-db-snapshot \  
  --db-instance-identifier database-mysql \  
  --db-snapshot-identifier mydbsnapshot
```

Output:

```
{  
  "DBSnapshot": {  
    "DBSnapshotIdentifier": "mydbsnapshot",  
    "DBInstanceIdentifier": "database-mysql",  
    "Engine": "mysql",  
    "AllocatedStorage": 100,  
    "Status": "creating",  
    "Port": 3306,  
    "AvailabilityZone": "us-east-1b",  
    "VpcId": "vpc-6594f31c",  
    "InstanceCreateTime": "2019-04-30T15:45:53.663Z",  
    "MasterUsername": "admin",  
    "EngineVersion": "5.6.40",  
    "LicenseModel": "general-public-license",  
    "SnapshotType": "manual",  
    "Iops": 1000,  
    "OptionGroupName": "default:mysql-5-6",  
    "PercentProgress": 0,  
    "StorageType": "io1",  
    "Encrypted": true,  
    "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/AKIAIOSFODNN7EXAMPLE",  
    "DBSnapshotArn": "arn:aws:rds:us-east-1:123456789012:snapshot:mydbsnapshot",  
    "IAMDatabaseAuthenticationEnabled": false,  
    "ProcessorFeatures": [],  
    "DbiResourceId": "db-AKIAIOSFODNN7EXAMPLE"  
  }  
}
```

For more information, see [Creating a DB Snapshot](#) in the *Amazon RDS User Guide*.

- For API details, see [CreateDBSnapshot](#) in *AWS CLI Command Reference*.

create-db-subnet-group

The following code example shows how to use `create-db-subnet-group`.

AWS CLI

To create a DB subnet group

The following `create-db-subnet-group` example creates a DB subnet group called `mysubnetgroup` using existing subnets.

```
aws rds create-db-subnet-group \  
  --db-subnet-group-name mysubnetgroup \  
  --db-subnet-group-description "test DB subnet group" \  
  --subnet-ids  
  '["subnet-0a1dc4e1a6f123456","subnet-070dd7ecb3aaaaaaaa","subnet-00f5b198bc0abcdef"]'
```

Output:

```
{  
  "DBSubnetGroup": {  
    "DBSubnetGroupName": "mysubnetgroup",  
    "DBSubnetGroupDescription": "test DB subnet group",  
    "VpcId": "vpc-0f08e7610a1b2c3d4",  
    "SubnetGroupStatus": "Complete",  
    "Subnets": [  
      {  
        "SubnetIdentifier": "subnet-070dd7ecb3aaaaaaaa",  
        "SubnetAvailabilityZone": {  
          "Name": "us-west-2b"  
        },  
        "SubnetStatus": "Active"  
      },  
      {  
        "SubnetIdentifier": "subnet-00f5b198bc0abcdef",  
        "SubnetAvailabilityZone": {  
          "Name": "us-west-2d"  
        },  
        "SubnetStatus": "Active"  
      },  
      {  
        "SubnetIdentifier": "subnet-0a1dc4e1a6f123456",  
        "SubnetAvailabilityZone": {
```

```

        "Name": "us-west-2b"
      },
      "SubnetStatus": "Active"
    }
  ],
  "DBSubnetGroupArn": "arn:aws:rds:us-
west-2:0123456789012:subgrp:mysubnetgroup"
}
}

```

For more information, see [Creating a DB Instance in a VPC](#) in the *Amazon RDS User Guide*.

- For API details, see [CreateDbSubnetGroup](#) in *AWS CLI Command Reference*.

create-event-subscription

The following code example shows how to use `create-event-subscription`.

AWS CLI

To create an event subscription

The following `create-event-subscription` example creates a subscription for backup and recovery events for DB instances in the current AWS account. Notifications are sent to an Amazon Simple Notification Service topic, specified by `--sns-topic-arn`.

```

aws rds create-event-subscription \
  --subscription-name my-instance-events \
  --source-type db-instance \
  --event-categories ["backup","recovery"] \
  --sns-topic-arn arn:aws:sns:us-east-1:123456789012:interesting-events

```

Output:

```

{
  "EventSubscription": {
    "Status": "creating",
    "CustSubscriptionId": "my-instance-events",
    "SubscriptionCreationTime": "Tue Jul 31 23:22:01 UTC 2018",
    "EventCategoriesList": [
      "backup",
      "recovery"
    ]
  },

```



```
    "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:interesting-events",
    "CustomerAwsId": "123456789012",
    "EventSubscriptionArn": "arn:aws:rds:us-east-1:123456789012:es:my-instance-
events",
    "SourceType": "db-instance",
    "Enabled": true
  }
}
```

- For API details, see [CreateEventSubscription](#) in *AWS CLI Command Reference*.

create-global-cluster

The following code example shows how to use `create-global-cluster`.

AWS CLI

To create a global DB cluster

The following `create-global-cluster` example creates a new Aurora MySQL-compatible global DB cluster.

```
aws rds create-global-cluster \  
  --global-cluster-identifier myglobalcluster \  
  --engine aurora-mysql
```

Output:

```
{
  "GlobalCluster": {
    "GlobalClusterIdentifier": "myglobalcluster",
    "GlobalClusterResourceId": "cluster-f0e523bfe07aabb",
    "GlobalClusterArn": "arn:aws:rds::123456789012:global-
cluster:myglobalcluster",
    "Status": "available",
    "Engine": "aurora-mysql",
    "EngineVersion": "5.7.mysql_aurora.2.07.2",
    "StorageEncrypted": false,
    "DeletionProtection": false,
    "GlobalClusterMembers": []
  }
}
```

For more information, see [Creating an Aurora global database](#) in the *Amazon Aurora User Guide*.

- For API details, see [CreateGlobalCluster](#) in *AWS CLI Command Reference*.

create-option-group

The following code example shows how to use create-option-group.

AWS CLI

To Create an Amazon RDS option group

The following create-option-group command creates a new Amazon RDS option group for Oracle Enterprise Edition version 11.2, is named `MyOptionGroup` and includes a description.

```
aws rds create-option-group \  
  --option-group-name MyOptionGroup \  
  --engine-name oracle-ee \  
  --major-engine-version 11.2 \  
  --option-group-description "Oracle Database Manager Database Control"
```

Output:

```
{  
  "OptionGroup": {  
    "OptionGroupName": "myoptiongroup",  
    "OptionGroupDescription": "Oracle Database Manager Database Control",  
    "EngineName": "oracle-ee",  
    "MajorEngineVersion": "11.2",  
    "Options": [],  
    "AllowsVpcAndNonVpcInstanceMemberships": true,  
    "OptionGroupArn": "arn:aws:rds:us-west-2:123456789012:og:myoptiongroup"  
  }  
}
```

- For API details, see [CreateOptionGroup](#) in *AWS CLI Command Reference*.

delete-blue-green-deployment

The following code example shows how to use delete-blue-green-deployment.

AWS CLI

Example 1: To delete resources in green environment for an RDS for MySQL DB instance

The following `delete-blue-green-deployment` example deletes the resources in a green environment for an RDS for MySQL DB instance.

```
aws rds delete-blue-green-deployment \  
  --blue-green-deployment-identifier bgd-v53303651eexfake \  
  --delete-target
```

Output:

```
{  
  "BlueGreenDeployment": {  
    "BlueGreenDeploymentIdentifier": "bgd-v53303651eexfake",  
    "BlueGreenDeploymentName": "bgd-cli-test-instance",  
    "Source": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance",  
    "Target": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance-green-rkfbpe",  
    "SwitchoverDetails": [  
      {  
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance",  
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance-green-rkfbpe",  
        "Status": "AVAILABLE"  
      },  
      {  
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance-replica-1",  
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance-replica-1-green-j382ha",  
        "Status": "AVAILABLE"  
      },  
      {  
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance-replica-2",  
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance-replica-2-green-ejv4ao",  
        "Status": "AVAILABLE"  
      },  
      {
```

```

        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-3",
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-3-green-vlpz3t",
        "Status": "AVAILABLE"
    }
],
"Tasks": [
    {
        "Name": "CREATING_READ_REPLICA_OF_SOURCE",
        "Status": "COMPLETED"
    },
    {
        "Name": "DB_ENGINE_VERSION_UPGRADE",
        "Status": "COMPLETED"
    },
    {
        "Name": "CONFIGURE_BACKUPS",
        "Status": "COMPLETED"
    },
    {
        "Name": "CREATING_TOPOLOGY_OF_SOURCE",
        "Status": "COMPLETED"
    }
],
"Status": "DELETING",
"CreateTime": "2022-02-25T21:18:51.183000+00:00",
>DeleteTime": "2022-02-25T22:25:31.331000+00:00"
}
}

```

For more information, see [Deleting a blue/green deployment](#) in the *Amazon RDS User Guide*.

Example 2: To delete resources in green environment for an Aurora MySQL DB cluster

The following delete-blue-green-deployment example deletes the resources in a green environment for an Aurora MySQL DB cluster.

```

aws rds delete-blue-green-deployment \
  --blue-green-deployment-identifier bgd-wi89nwzglccsfake \
  --delete-target

```

Output:

```

{
  "BlueGreenDeployment": {
    "BlueGreenDeploymentIdentifier": "bgd-wi89nwzglccsfake",
    "BlueGreenDeploymentName": "my-blue-green-deployment",
    "Source": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-
cluster",
    "Target": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-
cluster-green-3rnukl",
    "SwitchoverDetails": [
      {
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-
aurora-mysql-cluster",
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-
aurora-mysql-cluster-green-3rnukl",
        "Status": "AVAILABLE"
      },
      {
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-
mysql-cluster-1",
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-
mysql-cluster-1-green-gpmaxf",
        "Status": "AVAILABLE"
      },
      {
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-
mysql-cluster-2",
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-
mysql-cluster-2-green-j2oajq",
        "Status": "AVAILABLE"
      },
      {
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-
mysql-cluster-3",
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-
mysql-cluster-3-green-mkxies",
        "Status": "AVAILABLE"
      },
      {
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint",
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint-green-4sqjrq",
        "Status": "AVAILABLE"
      }
    ]
  }
}

```

```

    },
    {
      "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint",
      "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint-green-gwwzlg",
      "Status": "AVAILABLE"
    }
  ],
  "Tasks": [
    {
      "Name": "CREATING_READ_REPLICA_OF_SOURCE",
      "Status": "COMPLETED"
    },
    {
      "Name": "DB_ENGINE_VERSION_UPGRADE",
      "Status": "COMPLETED"
    },
    {
      "Name": "CREATE_DB_INSTANCES_FOR_CLUSTER",
      "Status": "COMPLETED"
    },
    {
      "Name": "CREATE_CUSTOM_ENDPOINTS",
      "Status": "COMPLETED"
    }
  ],
  "Status": "DELETING",
  "CreateTime": "2022-02-25T21:12:00.288000+00:00",
  "DeleteTime": "2022-02-25T22:29:11.336000+00:00"
}
}

```

For more information, see [Deleting a blue/green deployment](#) in the *Amazon Aurora User Guide*.

- For API details, see [DeleteBlueGreenDeployment](#) in *AWS CLI Command Reference*.

delete-db-cluster-endpoint

The following code example shows how to use `delete-db-cluster-endpoint`.

AWS CLI

To delete a custom DB cluster endpoint

The following `delete-db-cluster-endpoint` example deletes the specified custom DB cluster endpoint.

```
aws rds delete-db-cluster-endpoint \  
  --db-cluster-endpoint-identifier mycustomendpoint
```

Output:

```
{  
  "DBClusterEndpointIdentifier": "mycustomendpoint",  
  "DBClusterIdentifier": "mydbcluster",  
  "DBClusterEndpointResourceIdentifier": "cluster-endpoint-ANPAJ4AE5446DAEXAMPLE",  
  "Endpoint": "mycustomendpoint.cluster-custom-cnpeexample.us-  
east-1.rds.amazonaws.com",  
  "Status": "deleting",  
  "EndpointType": "CUSTOM",  
  "CustomEndpointType": "READER",  
  "StaticMembers": [  
    "dbinstance1",  
    "dbinstance2",  
    "dbinstance3"  
  ],  
  "ExcludedMembers": [],  
  "DBClusterEndpointArn": "arn:aws:rds:us-east-1:123456789012:cluster-  
endpoint:mycustomendpoint"  
}
```

For more information, see [Amazon Aurora Connection Management](#) in the *Amazon Aurora User Guide*.

- For API details, see [DeleteDbClusterEndpoint](#) in *AWS CLI Command Reference*.

delete-db-cluster-parameter-group

The following code example shows how to use `delete-db-cluster-parameter-group`.

AWS CLI

To delete a DB cluster parameter group

The following `delete-db-cluster-parameter-group` example deletes the specified DB cluster parameter group.

```
aws rds delete-db-cluster-parameter-group \  
  --db-cluster-parameter-group-name mydbclusterparametergroup
```

This command produces no output.

For more information, see [Working with DB Parameter Groups and DB Cluster Parameter Groups](#) in the *Amazon Aurora User Guide*.

- For API details, see [DeleteDbClusterParameterGroup](#) in *AWS CLI Command Reference*.

delete-db-cluster-snapshot

The following code example shows how to use `delete-db-cluster-snapshot`.

AWS CLI

To delete a DB cluster snapshot

The following `delete-db-cluster-snapshot` example deletes the specified DB cluster snapshot.

```
aws rds delete-db-cluster-snapshot \  
  --db-cluster-snapshot-identifier mydbclustersnapshot
```

Output:

```
{  
  "DBClusterSnapshot": {  
    "AvailabilityZones": [  
      "us-east-1a",  
      "us-east-1b",  
      "us-east-1e"  
    ],  
    "DBClusterSnapshotIdentifier": "mydbclustersnapshot",  
    "DBClusterIdentifier": "mydbcluster",  
    "SnapshotCreateTime": "2019-06-18T21:21:00.469Z",  
    "Engine": "aurora-mysql",  
    "AllocatedStorage": 0,  
    "Status": "available",  
    "Port": 0,  
    "VpcId": "vpc-6594f31c",
```



```
"ClusterCreateTime": "2019-04-15T14:18:42.785Z",
"MasterUsername": "myadmin",
"EngineVersion": "5.7.mysql_aurora.2.04.2",
"LicenseModel": "aurora-mysql",
"SnapshotType": "manual",
"PercentProgress": 100,
"StorageEncrypted": true,
"KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/AKIAIOSFODNN7EXAMPLE",
"DBClusterSnapshotArn": "arn:aws:rds:us-east-1:123456789012:cluster-
snapshot:mydbclustersnapshot",
"IAMDatabaseAuthenticationEnabled": false
}
}
```

For more information, see [Deleting a Snapshot](#) in the *Amazon Aurora User Guide*.

- For API details, see [DeleteDbClusterSnapshot](#) in *AWS CLI Command Reference*.

delete-db-cluster

The following code example shows how to use `delete-db-cluster`.

AWS CLI

Example 1: To delete a DB instance in a DB cluster

The following `delete-db-instance` example deletes the final DB instance in a DB cluster. You can't delete a DB cluster if it contains DB instances that aren't in the **deleting** state. You can't take a final snapshot when deleting a DB instance in a DB cluster.

```
aws rds delete-db-instance \
  --db-instance-identifier database-3
```

Output:

```
{
  "DBInstance": {
    "DBInstanceIdentifier": "database-3",
    "DBInstanceClass": "db.r4.large",
    "Engine": "aurora-postgresql",
    "DBInstanceStatus": "deleting",
```

```
...output omitted...  
}  
}
```

For more information, see [Deleting a DB Instance in an Aurora DB Cluster](#) in the *Amazon Aurora User Guide*.

Example 2: To delete a DB cluster

The following `delete-db-cluster` example deletes the DB cluster named `mycluster` and takes a final snapshot named `mycluster-final-snapshot`. The status of the DB cluster is **available** while the snapshot is being taken. To follow the progress of the deletion, use the `describe-db-clusters` CLI command.

```
aws rds delete-db-cluster \  
  --db-cluster-identifier mycluster \  
  --no-skip-final-snapshot \  
  --final-db-snapshot-identifier mycluster-final-snapshot
```

Output:

```
{  
  "DBCluster": {  
    "AllocatedStorage": 20,  
    "AvailabilityZones": [  
      "eu-central-1b",  
      "eu-central-1c",  
      "eu-central-1a"  
    ],  
    "BackupRetentionPeriod": 7,  
    "DBClusterIdentifier": "mycluster",  
    "DBClusterParameterGroup": "default.aurora-postgresql10",  
    "DBSubnetGroup": "default-vpc-aa11bb22",  
    "Status": "available",  
  
    ...output omitted...  
  }  
}
```

For more information, see [Aurora Clusters with a Single DB Instance](#) in the *Amazon Aurora User Guide*.

- For API details, see [DeleteDbCluster](#) in *AWS CLI Command Reference*.

delete-db-instance-automated-backup

The following code example shows how to use `delete-db-instance-automated-backup`.

AWS CLI

To delete a replicated automated backup from a Region

The following `delete-db-instance-automated-backup` example deletes the automated backup with the specified Amazon Resource Name (ARN).

```
aws rds delete-db-instance-automated-backup \
  --db-instance-automated-backups-arn "arn:aws:rds:us-west-2:123456789012:auto-
  backup:ab-jkib2gfgq5rv7replzadausbrktni2bn4example"
```

Output:

```
{
  "DBInstanceAutomatedBackup": {
    "DBInstanceArn": "arn:aws:rds:us-east-1:123456789012:db:new-orcl-db",
    "DbiResourceId": "db-JKIB2GFQ5RV7REPLZA4EXAMPLE",
    "Region": "us-east-1",
    "DBInstanceIdentifier": "new-orcl-db",
    "RestoreWindow": {},
    "AllocatedStorage": 20,
    "Status": "deleting",
    "Port": 1521,
    "AvailabilityZone": "us-east-1b",
    "VpcId": "vpc-#####",
    "InstanceCreateTime": "2020-12-04T15:28:31Z",
    "MasterUsername": "admin",
    "Engine": "oracle-se2",
    "EngineVersion": "12.1.0.2.v21",
    "LicenseModel": "bring-your-own-license",
    "OptionGroupName": "default:oracle-se2-12-1",
    "Encrypted": false,
    "StorageType": "gp2",
```

```
    "IAMDatabaseAuthenticationEnabled": false,  
    "BackupRetentionPeriod": 7,  
    "DBInstanceAutomatedBackupsArn": "arn:aws:rds:us-west-2:123456789012:auto-  
backup:ab-jkib2gfq5rv7replzadausbrktni2bn4example"  
  }  
}
```

For more information, see [Deleting replicated backups](#) in the *Amazon RDS User Guide*.

- For API details, see [DeleteDbInstanceAutomatedBackup](#) in *AWS CLI Command Reference*.

delete-db-instance

The following code example shows how to use `delete-db-instance`.

AWS CLI

To delete a DB instance

The following `delete-db-instance` example deletes the specified DB instance after creating a final DB snapshot named `test-instance-final-snap`.

```
aws rds delete-db-instance \  
  --db-instance-identifier test-instance \  
  --final-db-snapshot-identifier test-instance-final-snap
```

Output:

```
{  
  "DBInstance": {  
    "DBInstanceIdentifier": "test-instance",  
    "DBInstanceStatus": "deleting",  
    ...some output truncated...  
  }  
}
```

- For API details, see [DeleteDBInstance](#) in *AWS CLI Command Reference*.

delete-db-parameter-group

The following code example shows how to use `delete-db-parameter-group`.

AWS CLI

To delete a DB parameter group

The following command example deletes a DB parameter group.

```
aws rds delete-db-parameter-group \  
  --db-parameter-group-name mydbparametergroup
```

This command produces no output.

For more information, see [Working with DB Parameter Groups](#) in the *Amazon RDS User Guide*.

- For API details, see [DeleteDBParameterGroup](#) in *AWS CLI Command Reference*.

delete-db-proxy-endpoint

The following code example shows how to use delete-db-proxy-endpoint.

AWS CLI

To delete a DB proxy endpoint for an RDS database

The following delete-db-proxy-endpoint example deletes a DB proxy endpoint for the target database.

```
aws rds delete-db-proxy-endpoint \  
  --db-proxy-endpoint-name proxyEP1
```

Output:

```
{  
  "DBProxyEndpoint":  
    {  
      "DBProxyEndpointName": "proxyEP1",  
      "DBProxyEndpointArn": "arn:aws:rds:us-east-1:123456789012:db-proxy-  
endpoint:prx-endpoint-0123a01b12345c0ab",  
      "DBProxyName": "proxyExample",  
      "Status": "deleting",  
      "VpcId": "vpc-1234567",  
      "VpcSecurityGroupIds": [  
        "sg-1234",
```

```
        "sg-5678"
      ],
      "VpcSubnetIds": [
        "subnetgroup1",
        "subnetgroup2"
      ],
      "Endpoint": "proxyEP1.endpoint.proxy-ab0cd1efghij.us-
east-1.rds.amazonaws.com",
      "CreateDate": "2023-04-13T01:49:38.568000+00:00",
      "TargetRole": "READ_ONLY",
      "IsDefault": false
    }
  }
}
```

For more information, see [Deleting a proxy endpoint](#) in the *Amazon RDS User Guide* and [Deleting a proxy endpoint](#) in the *Amazon Aurora User Guide*.

- For API details, see [DeleteDbProxyEndpoint](#) in *AWS CLI Command Reference*.

delete-db-proxy

The following code example shows how to use `delete-db-proxy`.

AWS CLI

To delete a DB proxy for an RDS database

The following `delete-db-proxy` example deletes a DB proxy.

```
aws rds delete-db-proxy \
  --db-proxy-name proxyExample
```

Output:

```
{
  "DBProxy": {
    "DBProxyName": "proxyExample",
    "DBProxyArn": "arn:aws:rds:us-east-1:123456789012:db-
proxy:prx-0123a01b12345c0ab",
    "Status": "deleting",
    "EngineFamily": "PostgreSQL",
```

```

    "VpcId": "vpc-1234567",
    "VpcSecurityGroupIds": [
        "sg-1234",
        "sg-5678"
    ],
    "VpcSubnetIds": [
        "subnetgroup1",
        "subnetgroup2"
    ],
    "Auth": "[
        {
            "Description": "proxydescription`"
            "AuthScheme": "SECRETS",
            "SecretArn": "arn:aws:secretsmanager:us-
west-2:123456789123:secret:proxysecret1-Abcd1e",
            "IAMAuth": "DISABLED"
        } ],
    "RoleArn": "arn:aws:iam::12345678912:role/ProxyPostgreSQLRole",
    "Endpoint": "proxyExample.proxy-ab0cd1efghij.us-
east-1.rds.amazonaws.com",
    "RequireTLS": false,
    "IdleClientTimeout": 1800,
    "DebuggingLogging": false,
    "CreateDate": "2023-04-05T16:09:33.452000+00:00",
    "UpdateDate": "2023-04-13T01:49:38.568000+00:00"
}
}

```

For more information, see [Deleting an RDS Proxy](#) in the *Amazon RDS User Guide* and [Deleting an RDS Proxy](#) in the *Amazon Aurora User Guide*.

- For API details, see [DeleteDbProxy](#) in *AWS CLI Command Reference*.

delete-db-security-group

The following code example shows how to use delete-db-security-group.

AWS CLI

To delete a DB security group

The following delete-db-security-group example deletes a DB security group named mysecuritygroup.

```
aws rds delete-db-security-group \  
  --db-security-group-name mysecuritygroup
```

This command produces no output.

For more information, see [Working with DB security groups \(EC2-Classic platform\)](#) in the *Amazon RDS User Guide*.

- For API details, see [DeleteDbSecurityGroup](#) in *AWS CLI Command Reference*.

delete-db-shard-group

The following code example shows how to use `delete-db-shard-group`.

AWS CLI

Example 1: To delete a DB shard group unsuccessfully

The following `delete-db-shard-group` example shows the error that occurs when you try to delete a DB shard group before deleting all of your databases and schemas.

```
aws rds delete-db-shard-group \  
  --db-shard-group-identifier limitless-test-shard-grp
```

Output:

```
An error occurred (InvalidDBShardGroupState) when calling the DeleteDBShardGroup  
operation: Unable to delete the DB shard group limitless-test-db-shard-group.  
Delete all of your Limitless Database databases and schemas, then try again.
```

Example 2: To delete a DB shard group successfully

The following `delete-db-shard-group` example deletes a DB shard group after you've deleted all of your databases and schemas, including the `public` schema.

```
aws rds delete-db-shard-group \  
  --db-shard-group-identifier limitless-test-shard-grp
```

Output:

```
{
```



```
"DBShardGroupResourceId": "shardgroup-7bb446329da94788b3f957746example",
"DBShardGroupIdentifier": "limitless-test-shard-grp",
"DBClusterIdentifier": "limitless-test-cluster",
"MaxACU": 768.0,
"ComputeRedundancy": 0,
"Status": "deleting",
"PubliclyAccessible": true,
"Endpoint": "limitless-test-cluster.limitless-cekyceexample.us-east-2.rds.amazonaws.com"
}
```

For more information, see [Deleting Aurora DB clusters and DB instances](#) in the *Amazon Aurora User Guide*.

- For API details, see [DeleteDbShardGroup](#) in *AWS CLI Command Reference*.

delete-db-snapshot

The following code example shows how to use `delete-db-snapshot`.

AWS CLI

To delete a DB snapshot

The following `delete-db-snapshot` example deletes the specified DB snapshot.

```
aws rds delete-db-snapshot \
  --db-snapshot-identifier mydbsnapshot
```

Output:

```
{
  "DBSnapshot": {
    "DBSnapshotIdentifier": "mydbsnapshot",
    "DBInstanceIdentifier": "database-mysql",
    "SnapshotCreateTime": "2019-06-18T22:08:40.702Z",
    "Engine": "mysql",
    "AllocatedStorage": 100,
    "Status": "deleted",
    "Port": 3306,
    "AvailabilityZone": "us-east-1b",
    "VpcId": "vpc-6594f31c",
    "InstanceCreateTime": "2019-04-30T15:45:53.663Z",
```

```
"MasterUsername": "admin",
"EngineVersion": "5.6.40",
"LicenseModel": "general-public-license",
"SnapshotType": "manual",
"Iops": 1000,
"OptionGroupName": "default:mysql-5-6",
"PercentProgress": 100,
"StorageType": "io1",
"Encrypted": true,
"KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/AKIAIOSFODNN7EXAMPLE",
"DBSnapshotArn": "arn:aws:rds:us-east-1:123456789012:snapshot:mydbsnapshot",
"IAMDatabaseAuthenticationEnabled": false,
"ProcessorFeatures": [],
"DbiResourceId": "db-AKIAIOSFODNN7EXAMPLE"
}
}
```

For more information, see [Deleting a Snapshot](#) in the *Amazon RDS User Guide*.

- For API details, see [DeleteDbSnapshot](#) in *AWS CLI Command Reference*.

delete-db-subnet-group

The following code example shows how to use `delete-db-subnet-group`.

AWS CLI

To delete a DB subnet group

The following `delete-db-subnet-group` example deletes the DB subnet group called `mysubnetgroup`.

```
aws rds delete-db-subnet-group --db-subnet-group-name mysubnetgroup
```

This command produces no output.

For more information, see [Working with a DB Instance in a VPC](#) in the *Amazon RDS User Guide*.

- For API details, see [DeleteDbSubnetGroup](#) in *AWS CLI Command Reference*.

delete-event-subscription

The following code example shows how to use `delete-event-subscription`.

AWS CLI

To delete an event subscription

The following `delete-event-subscription` example deletes the specified event subscription.

```
aws rds delete-event-subscription --subscription-name my-instance-events
```

Output:

```
{
  "EventSubscription": {
    "EventSubscriptionArn": "arn:aws:rds:us-east-1:123456789012:es:my-instance-
events",
    "CustomerAwsId": "123456789012",
    "Enabled": false,
    "SourceIdsList": [
      "test-instance"
    ],
    "SourceType": "db-instance",
    "EventCategoriesList": [
      "backup",
      "recovery"
    ],
    "SubscriptionCreationTime": "2018-07-31 23:22:01.893",
    "CustSubscriptionId": "my-instance-events",
    "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:interesting-events",
    "Status": "deleting"
  }
}
```

- For API details, see [DeleteEventSubscription](#) in *AWS CLI Command Reference*.

`delete-global-cluster`

The following code example shows how to use `delete-global-cluster`.

AWS CLI

To delete a global DB cluster

The following `delete-global-cluster` example deletes an Aurora MySQL-compatible global DB cluster. The output shows the cluster that you're deleting, but subsequent `describe-global-clusters` commands don't list that DB cluster.

```
aws rds delete-global-cluster \  
  --global-cluster-identifier myglobalcluster
```

Output:

```
{  
  "GlobalCluster": {  
    "GlobalClusterIdentifier": "myglobalcluster",  
    "GlobalClusterResourceId": "cluster-f0e523bfe07aabb",  
    "GlobalClusterArn": "arn:aws:rds::123456789012:global-  
cluster:myglobalcluster",  
    "Status": "available",  
    "Engine": "aurora-mysql",  
    "EngineVersion": "5.7.mysql_aurora.2.07.2",  
    "StorageEncrypted": false,  
    "DeletionProtection": false,  
    "GlobalClusterMembers": []  
  }  
}
```

For more information, see [Deleting an Aurora global database](#) in the *Amazon Aurora User Guide*.

- For API details, see [DeleteGlobalCluster](#) in *AWS CLI Command Reference*.

delete-option-group

The following code example shows how to use `delete-option-group`.

AWS CLI

To delete an option group

The following `delete-option-group` example deletes the specified option group.

```
aws rds delete-option-group \  
  --option-group-name myoptiongroup
```

This command produces no output.

For more information, see [Deleting an Option Group](#) in the *Amazon RDS User Guide*.

- For API details, see [DeleteOptionGroup](#) in *AWS CLI Command Reference*.

deregister-db-proxy-targets

The following code example shows how to use `deregister-db-proxy-targets`.

AWS CLI

To deregister a DB proxy target from database target group

The following `deregister-db-proxy-targets` example removes the association between the proxy `proxyExample` and its target.

```
aws rds deregister-db-proxy-targets \  
  --db-proxy-name proxyExample \  
  --db-instance-identifiers database-1
```

This command produces no output.

For more information, see [Deleting an RDS Proxy](#) in the *Amazon RDS User Guide* and [Deleting an RDS Proxy](#) in the *Amazon Aurora User Guide*.

- For API details, see [DeregisterDbProxyTargets](#) in *AWS CLI Command Reference*.

describe-account-attributes

The following code example shows how to use `describe-account-attributes`.

AWS CLI

To describe account attributes

The following `describe-account-attributes` example retrieves the attributes for the current AWS account.

```
aws rds describe-account-attributes
```

Output:

```
{
  "AccountQuotas": [
    {
      "Max": 40,
      "Used": 4,
      "AccountQuotaName": "DBInstances"
    },
    {
      "Max": 40,
      "Used": 0,
      "AccountQuotaName": "ReservedDBInstances"
    },
    {
      "Max": 100000,
      "Used": 40,
      "AccountQuotaName": "AllocatedStorage"
    },
    {
      "Max": 25,
      "Used": 0,
      "AccountQuotaName": "DBSecurityGroups"
    },
    {
      "Max": 20,
      "Used": 0,
      "AccountQuotaName": "AuthorizationsPerDBSecurityGroup"
    },
    {
      "Max": 50,
      "Used": 1,
      "AccountQuotaName": "DBParameterGroups"
    },
    {
      "Max": 100,
      "Used": 3,
      "AccountQuotaName": "ManualSnapshots"
    },
    {
      "Max": 20,
      "Used": 0,
      "AccountQuotaName": "EventSubscriptions"
    },
  ],
}
```

```
{
  "Max": 50,
  "Used": 1,
  "AccountQuotaName": "DBSubnetGroups"
},
{
  "Max": 20,
  "Used": 1,
  "AccountQuotaName": "OptionGroups"
},
{
  "Max": 20,
  "Used": 6,
  "AccountQuotaName": "SubnetsPerDBSubnetGroup"
},
{
  "Max": 5,
  "Used": 0,
  "AccountQuotaName": "ReadReplicasPerMaster"
},
{
  "Max": 40,
  "Used": 1,
  "AccountQuotaName": "DBClusters"
},
{
  "Max": 50,
  "Used": 0,
  "AccountQuotaName": "DBClusterParameterGroups"
},
{
  "Max": 5,
  "Used": 0,
  "AccountQuotaName": "DBClusterRoles"
}
]
```

- For API details, see [DescribeAccountAttributes](#) in *AWS CLI Command Reference*.

describe-blue-green-deployments

The following code example shows how to use `describe-blue-green-deployments`.

AWS CLI

Example 1: To describe a blue/green deployment of an RDS DB instance after creation completes

The following `describe-blue-green-deployment` example retrieves the details of a blue/green deployment after creation completes.

```
aws rds describe-blue-green-deployments \  
  --blue-green-deployment-identifier bgd-v53303651eexfake
```

Output:

```
{  
  "BlueGreenDeployments": [  
    {  
      "BlueGreenDeploymentIdentifier": "bgd-v53303651eexfake",  
      "BlueGreenDeploymentName": "bgd-cli-test-instance",  
      "Source": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance",  
      "Target": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance-green-  
rkfbpe",  
      "SwitchoverDetails": [  
        {  
          "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-  
instance",  
          "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-  
instance-green-rkfbpe",  
          "Status": "AVAILABLE"  
        },  
        {  
          "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-  
instance-replica-1",  
          "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-  
instance-replica-1-green-j382ha",  
          "Status": "AVAILABLE"  
        },  
        {  
          "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-  
instance-replica-2",  
          "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-  
instance-replica-2-green-ejv4ao",  
          "Status": "AVAILABLE"  
        }  
      ]  
    }  
  ]  
}
```



```

        {
            "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-3",
            "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-3-green-vlpz3t",
            "Status": "AVAILABLE"
        }
    ],
    "Tasks": [
        {
            "Name": "CREATING_READ_REPLICA_OF_SOURCE",
            "Status": "COMPLETED"
        },
        {
            "Name": "DB_ENGINE_VERSION_UPGRADE",
            "Status": "COMPLETED"
        },
        {
            "Name": "CONFIGURE_BACKUPS",
            "Status": "COMPLETED"
        },
        {
            "Name": "CREATING_TOPOLOGY_OF_SOURCE",
            "Status": "COMPLETED"
        }
    ],
    "Status": "AVAILABLE",
    "CreateTime": "2022-02-25T21:18:51.183000+00:00"
}
]
}

```

For more information, see [Viewing a blue/green deployment](#) in the *Amazon RDS User Guide*.

Example 2: To describe a blue/green deployment for an Aurora MySQL DB cluster

The following describe-blue-green-deployment example retrieves the details of a blue/green deployment.

```
aws rds describe-blue-green-deployments \
  --blue-green-deployment-identifier bgd-wi89nwzglccsfake
```

Output:

```
{
  "BlueGreenDeployments": [
    {
      "BlueGreenDeploymentIdentifier": "bgd-wi89nwzglccsfake",
      "BlueGreenDeploymentName": "my-blue-green-deployment",
      "Source": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-
cluster",
      "Target": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-
cluster-green-3rnukl",
      "SwitchoverDetails": [
        {
          "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-
aurora-mysql-cluster",
          "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-
aurora-mysql-cluster-green-3rnukl",
          "Status": "AVAILABLE"
        },
        {
          "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-1",
          "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-1-green-gpmaxf",
          "Status": "AVAILABLE"
        },
        {
          "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-2",
          "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-2-green-j2oajq",
          "Status": "AVAILABLE"
        },
        {
          "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-3",
          "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-
aurora-mysql-cluster-3-green-mkxies",
          "Status": "AVAILABLE"
        },
        {
          "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint",
          "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint-green-4sqjrq",

```

```

        "Status": "AVAILABLE"
      },
      {
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint",
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint-green-gwwzlg",
        "Status": "AVAILABLE"
      }
    ],
    "Tasks": [
      {
        "Name": "CREATING_READ_REPLICA_OF_SOURCE",
        "Status": "COMPLETED"
      },
      {
        "Name": "DB_ENGINE_VERSION_UPGRADE",
        "Status": "COMPLETED"
      },
      {
        "Name": "CREATE_DB_INSTANCES_FOR_CLUSTER",
        "Status": "COMPLETED"
      },
      {
        "Name": "CREATE_CUSTOM_ENDPOINTS",
        "Status": "COMPLETED"
      }
    ],
    "Status": "AVAILABLE",
    "CreateTime": "2022-02-25T21:12:00.288000+00:00"
  }
]
}

```

For more information, see [Viewing a blue/green deployment](#) in the *Amazon Aurora User Guide*.

Example 3: To describe a blue/green deployment for an Aurora MySQL cluster after switchover

The following describe-blue-green-deployment example retrieves the details about a blue/green deployment after the green environment is promoted to be the production environment.

```
aws rds describe-blue-green-deployments \  
  --blue-green-deployment-identifier bgd-wi89nwzglccsfake
```

Output:

```
{  
  "BlueGreenDeployments": [  
    {  
      "BlueGreenDeploymentIdentifier": "bgd-wi89nwzglccsfake",  
      "BlueGreenDeploymentName": "my-blue-green-deployment",  
      "Source": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-  
cluster-old1",  
      "Target": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-  
cluster",  
      "SwitchoverDetails": [  
        {  
          "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-  
aurora-mysql-cluster-old1",  
          "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-  
aurora-mysql-cluster",  
          "Status": "SWITCHOVER_COMPLETED"  
        },  
        {  
          "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-  
aurora-mysql-cluster-1-old1",  
          "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-  
aurora-mysql-cluster-1",  
          "Status": "SWITCHOVER_COMPLETED"  
        },  
        {  
          "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-  
aurora-mysql-cluster-2-old1",  
          "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-  
aurora-mysql-cluster-2",  
          "Status": "SWITCHOVER_COMPLETED"  
        },  
        {  
          "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-  
aurora-mysql-cluster-3-old1",  
          "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-  
aurora-mysql-cluster-3",  
          "Status": "SWITCHOVER_COMPLETED"  
        }  
      ]  
    }  
  ]  
}
```

```

        {
            "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint-old1",
            "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint",
            "Status": "SWITCHOVER_COMPLETED"
        },
        {
            "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint-old1",
            "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint",
            "Status": "SWITCHOVER_COMPLETED"
        }
    ],
    "Tasks": [
        {
            "Name": "CREATING_READ_REPLICA_OF_SOURCE",
            "Status": "COMPLETED"
        },
        {
            "Name": "DB_ENGINE_VERSION_UPGRADE",
            "Status": "COMPLETED"
        },
        {
            "Name": "CREATE_DB_INSTANCES_FOR_CLUSTER",
            "Status": "COMPLETED"
        },
        {
            "Name": "CREATE_CUSTOM_ENDPOINTS",
            "Status": "COMPLETED"
        }
    ],
    "Status": "SWITCHOVER_COMPLETED",
    "CreateTime": "2022-02-25T22:38:49.522000+00:00"
}
]
}

```

For more information, see [Viewing a blue/green deployment](#) in the *Amazon Aurora User Guide*.

Example 4: To describe a combined blue/green deployment

The following `describe-blue-green-deployments` example retrieves the details of a combined blue/green deployment.

```
aws rds describe-blue-green-deployments
```

Output:

```
{
  "BlueGreenDeployments": [
    {
      "BlueGreenDeploymentIdentifier": "bgd-wi89nwzgfakelccs",
      "BlueGreenDeploymentName": "my-blue-green-deployment",
      "Source": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-cluster",
      "Target": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-cluster-green-3rnukl",
      "SwitchoverDetails": [
        {
          "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-cluster",
          "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-cluster-green-3rnukl",
          "Status": "AVAILABLE"
        },
        {
          "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-mysql-cluster-1",
          "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-mysql-cluster-1-green-gpmaxf",
          "Status": "AVAILABLE"
        },
        {
          "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-mysql-cluster-2",
          "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-mysql-cluster-2-green-j2oajq",
          "Status": "AVAILABLE"
        },
        {
          "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-mysql-cluster-3",
          "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-mysql-cluster-3-green-mkxies",

```

```

        "Status": "AVAILABLE"
      },
      {
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint",
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint-green-4sqjrq",
        "Status": "AVAILABLE"
      },
      {
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint",
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint-green-gwzlg",
        "Status": "AVAILABLE"
      }
    ],
    "Tasks": [
      {
        "Name": "CREATING_READ_REPLICA_OF_SOURCE",
        "Status": "COMPLETED"
      },
      {
        "Name": "DB_ENGINE_VERSION_UPGRADE",
        "Status": "COMPLETED"
      },
      {
        "Name": "CREATE_DB_INSTANCES_FOR_CLUSTER",
        "Status": "COMPLETED"
      },
      {
        "Name": "CREATE_CUSTOM_ENDPOINTS",
        "Status": "COMPLETED"
      }
    ],
    "Status": "AVAILABLE",
    "CreateTime": "2022-02-25T21:12:00.288000+00:00"
  },
  {
    "BlueGreenDeploymentIdentifier": "bgd-v5330365fake1eex",
    "BlueGreenDeploymentName": "bgd-cli-test-instance",
    "Source": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance-old1",
    "Target": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance",
    "SwitchoverDetails": [

```

```
    {
      "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-old1",
      "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance",
      "Status": "SWITCHOVER_COMPLETED"
    },
    {
      "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-1-old1",
      "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-1",
      "Status": "SWITCHOVER_COMPLETED"
    },
    {
      "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-2-old1",
      "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-2",
      "Status": "SWITCHOVER_COMPLETED"
    },
    {
      "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-3-old1",
      "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-3",
      "Status": "SWITCHOVER_COMPLETED"
    }
  ],
  "Tasks": [
    {
      "Name": "CREATING_READ_REPLICA_OF_SOURCE",
      "Status": "COMPLETED"
    },
    {
      "Name": "DB_ENGINE_VERSION_UPGRADE",
      "Status": "COMPLETED"
    },
    {
      "Name": "CONFIGURE_BACKUPS",
      "Status": "COMPLETED"
    },
    {
      "Name": "CREATING_TOPOLOGY_OF_SOURCE",
```



```

        "Status": "COMPLETED"
      }
    ],
    "Status": "SWITCHOVER_COMPLETED",
    "CreateTime": "2022-02-25T22:33:22.225000+00:00"
  }
]
}

```

For more information, see [Viewing a blue/green deployment](#) in the *Amazon RDS User Guide* and [Viewing a blue/green deployment](#) in the *Amazon Aurora User Guide*.

- For API details, see [DescribeBlueGreenDeployments](#) in *AWS CLI Command Reference*.

describe-certificates

The following code example shows how to use `describe-certificates`.

AWS CLI

To describe certificates

The following `describe-certificates` example retrieves the details of the certificate associated with the user's default region.

```
aws rds describe-certificates
```

Output:

```

{
  "Certificates": [
    {
      "CertificateIdentifier": "rds-ca-ecc384-g1",
      "CertificateType": "CA",
      "Thumbprint": "2ee3dcc06e50192559b13929e73484354f23387d",
      "ValidFrom": "2021-05-24T22:06:59+00:00",
      "ValidTill": "2121-05-24T23:06:59+00:00",
      "CertificateArn": "arn:aws:rds:us-west-2::cert:rds-ca-ecc384-g1",
      "CustomerOverride": false
    },
    {
      "CertificateIdentifier": "rds-ca-rsa4096-g1",

```

```

    "CertificateType": "CA",
    "Thumbprint": "19da4f2af579a8ae1f6a0fa77aa5befd874b4cab",
    "ValidFrom": "2021-05-24T22:03:20+00:00",
    "ValidTill": "2121-05-24T23:03:20+00:00",
    "CertificateArn": "arn:aws:rds:us-west-2::cert:rds-ca-rsa4096-g1",
    "CustomerOverride": false
  },
  {
    "CertificateIdentifier": "rds-ca-rsa2048-g1",
    "CertificateType": "CA",
    "Thumbprint": "7c40cb42714b6fdb2b296f9bbd0e8bb364436a76",
    "ValidFrom": "2021-05-24T21:59:00+00:00",
    "ValidTill": "2061-05-24T22:59:00+00:00",
    "CertificateArn": "arn:aws:rds:us-west-2::cert:rds-ca-rsa2048-g1",
    "CustomerOverride": true,
    "CustomerOverrideValidTill": "2061-05-24T22:59:00+00:00"
  },
  {
    "CertificateIdentifier": "rds-ca-2019",
    "CertificateType": "CA",
    "Thumbprint": "d40ddb29e3750dffa671c3140bbf5f478d1c8096",
    "ValidFrom": "2019-08-22T17:08:50+00:00",
    "ValidTill": "2024-08-22T17:08:50+00:00",
    "CertificateArn": "arn:aws:rds:us-west-2::cert:rds-ca-2019",
    "CustomerOverride": false
  }
],
"DefaultCertificateForNewLaunches": "rds-ca-rsa2048-g1"
}

```

For more information, see [Using SSL/TLS to encrypt a connection to a DB instance](#) in the *Amazon RDS User Guide* and [Using SSL/TLS to encrypt a connection to a DB cluster](#) in the *Amazon Aurora User Guide*.

- For API details, see [DescribeCertificates](#) in *AWS CLI Command Reference*.

describe-db-cluster-backtracks

The following code example shows how to use `describe-db-cluster-backtracks`.

AWS CLI

To describe backtracks for a DB cluster

The following `describe-db-cluster-backtracks` example retrieves details about the specified DB cluster.

```
aws rds describe-db-cluster-backtracks \
  --db-cluster-identifier mydbcluster
```

Output:

```
{
  "DBClusterBacktracks": [
    {
      "DBClusterIdentifier": "mydbcluster",
      "BacktrackIdentifier": "2f5f5294-0dd2-44c9-9f50-EXAMPLE",
      "BacktrackTo": "2021-02-12T04:59:22Z",
      "BacktrackedFrom": "2021-02-12T14:37:31.640Z",
      "BacktrackRequestCreationTime": "2021-02-12T14:36:18.819Z",
      "Status": "COMPLETED"
    },
    {
      "DBClusterIdentifier": "mydbcluster",
      "BacktrackIdentifier": "3c7a6421-af2a-4ea3-ae95-EXAMPLE",
      "BacktrackTo": "2021-02-11T22:53:46Z",
      "BacktrackedFrom": "2021-02-12T00:09:27.006Z",
      "BacktrackRequestCreationTime": "2021-02-12T00:07:53.487Z",
      "Status": "COMPLETED"
    }
  ]
}
```

For more information, see [Backtracking an Aurora DB cluster](#) in the *Amazon Aurora User Guide*.

- For API details, see [DescribeDbClusterBacktracks](#) in *AWS CLI Command Reference*.

describe-db-cluster-endpoints

The following code example shows how to use `describe-db-cluster-endpoints`.

AWS CLI

Example 1: To describe DB cluster endpoints

The following `describe-db-cluster-endpoints` example retrieves details for your DB cluster endpoints. The most common kinds of Aurora clusters have two endpoints. One endpoint has type `WRITER`. You can use this endpoint for all SQL statements. The other endpoint has type `READER`. You can use this endpoint only for `SELECT` and other read-only SQL statements.

```
aws rds describe-db-cluster-endpoints
```

Output:

```
{
  "DBClusterEndpoints": [
    {
      "DBClusterIdentifier": "my-database-1",
      "Endpoint": "my-database-1.cluster-cnpxample.us-east-1.rds.amazonaws.com",
      "Status": "creating",
      "EndpointType": "WRITER"
    },
    {
      "DBClusterIdentifier": "my-database-1",
      "Endpoint": "my-database-1.cluster-ro-cnpxample.us-east-1.rds.amazonaws.com",
      "Status": "creating",
      "EndpointType": "READER"
    },
    {
      "DBClusterIdentifier": "mydbcluster",
      "Endpoint": "mydbcluster.cluster-cnpxample.us-east-1.rds.amazonaws.com",
      "Status": "available",
      "EndpointType": "WRITER"
    },
    {
      "DBClusterIdentifier": "mydbcluster",
      "Endpoint": "mydbcluster.cluster-ro-cnpxample.us-east-1.rds.amazonaws.com",
      "Status": "available",
      "EndpointType": "READER"
    }
  ]
}
```

Example 2: To describe DB cluster endpoints of a single DB cluster

The following `describe-db-cluster-endpoints` example retrieves details for the DB cluster endpoints of a single specified DB cluster. Aurora Serverless clusters have only a single endpoint with a type of `WRITER`.

```
aws rds describe-db-cluster-endpoints \  
  --db-cluster-identifier serverless-cluster
```

Output:

```
{  
  "DBClusterEndpoints": [  
    {  
      "Status": "available",  
      "Endpoint": "serverless-cluster.cluster-cnpxample.us-  
east-1.rds.amazonaws.com",  
      "DBClusterIdentifier": "serverless-cluster",  
      "EndpointType": "WRITER"  
    }  
  ]  
}
```

For more information, see [Amazon Aurora Connection Management](#) in the *Amazon Aurora User Guide*.

- For API details, see [DescribeDbClusterEndpoints](#) in *AWS CLI Command Reference*.

`describe-db-cluster-parameter-groups`

The following code example shows how to use `describe-db-cluster-parameter-groups`.

AWS CLI

To describe DB cluster parameter groups

The following `describe-db-cluster-parameter-groups` example retrieves details for your DB cluster parameter groups.

```
aws rds describe-db-cluster-parameter-groups
```

Output:

```
{
  "DBClusterParameterGroups": [
    {
      "DBClusterParameterGroupName": "default.aurora-mysql5.7",
      "DBParameterGroupFamily": "aurora-mysql5.7",
      "Description": "Default cluster parameter group for aurora-mysql5.7",
      "DBClusterParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:cluster-pg:default.aurora-mysql5.7"
    },
    {
      "DBClusterParameterGroupName": "default.aurora-postgresql9.6",
      "DBParameterGroupFamily": "aurora-postgresql9.6",
      "Description": "Default cluster parameter group for aurora-
postgresql9.6",
      "DBClusterParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:cluster-pg:default.aurora-postgresql9.6"
    },
    {
      "DBClusterParameterGroupName": "default.aurora5.6",
      "DBParameterGroupFamily": "aurora5.6",
      "Description": "Default cluster parameter group for aurora5.6",
      "DBClusterParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:cluster-pg:default.aurora5.6"
    },
    {
      "DBClusterParameterGroupName": "mydbclusterpg",
      "DBParameterGroupFamily": "aurora-mysql5.7",
      "Description": "My DB cluster parameter group",
      "DBClusterParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:cluster-pg:mydbclusterpg"
    },
    {
      "DBClusterParameterGroupName": "mydbclusterpgcopy",
      "DBParameterGroupFamily": "aurora-mysql5.7",
      "Description": "Copy of mydbclusterpg parameter group",
      "DBClusterParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:cluster-pg:mydbclusterpgcopy"
    }
  ]
}
```

For more information, see [Working with DB Parameter Groups and DB Cluster Parameter Groups](#) in the *Amazon Aurora User Guide*.

- For API details, see [DescribeDbClusterParameterGroups](#) in *AWS CLI Command Reference*.

describe-db-cluster-parameters

The following code example shows how to use `describe-db-cluster-parameters`.

AWS CLI

Example 1: To describe the parameters in a DB cluster parameter group

The following `describe-db-cluster-parameters` example retrieves details about the parameters in a DB cluster parameter group.

```
aws rds describe-db-cluster-parameters \  
  --db-cluster-parameter-group-name mydbclusterpg
```

Output:

```
{  
  "Parameters": [  
    {  
      "ParameterName": "allow-suspicious-udfs",  
      "Description": "Controls whether user-defined functions that have only  
an xxx symbol for the main function can be loaded",  
      "Source": "engine-default",  
      "ApplyType": "static",  
      "DataType": "boolean",  
      "AllowedValues": "0,1",  
      "IsModifiable": false,  
      "ApplyMethod": "pending-reboot",  
      "SupportedEngineModes": [  
        "provisioned"  
      ]  
    },  
    {  
      "ParameterName": "aurora_lab_mode",  
      "ParameterValue": "0",  
      "Description": "Enables new features in the Aurora engine.",  
      "Source": "engine-default",  
      "ApplyType": "static",  
      "DataType": "boolean",  
      "AllowedValues": "0,1",  
      "IsModifiable": true,  
    }  
  ]  
}
```

```

        "ApplyMethod": "pending-reboot",
        "SupportedEngineModes": [
            "provisioned"
        ]
    },
    ...some output truncated...
]
}

```

Example 2: To list only the parameter names in a DB cluster parameter group

The following `describe-db-cluster-parameters` example retrieves only the names of the parameters in a DB cluster parameter group.

```

aws rds describe-db-cluster-parameters \
  --db-cluster-parameter-group-name default.aurora-mysql5.7 \
  --query 'Parameters[].{ParameterName:ParameterName}'

```

Output:

```

[
  {
    "ParameterName": "allow-suspicious-udfs"
  },
  {
    "ParameterName": "aurora_binlog_read_buffer_size"
  },
  {
    "ParameterName": "aurora_binlog_replication_max_yield_seconds"
  },
  {
    "ParameterName": "aurora_binlog_use_large_read_buffer"
  },
  {
    "ParameterName": "aurora_lab_mode"
  },
  ...some output truncated...
]

```

Example 3: To describe only the modifiable parameters in a DB cluster parameter group

The following `describe-db-cluster-parameters` example retrieves the names of only the parameters that you can modify in a DB cluster parameter group.

```
aws rds describe-db-cluster-parameters \  
  --db-cluster-parameter-group-name default.aurora-mysql5.7 \  
  --query 'Parameters[].{ParameterName:ParameterName,IsModifiable:IsModifiable} |  
  [?IsModifiable == `true`]'
```

Output:

```
[  
  {  
    "ParameterName": "aurora_binlog_read_buffer_size",  
    "IsModifiable": true  
  },  
  {  
    "ParameterName": "aurora_binlog_replication_max_yield_seconds",  
    "IsModifiable": true  
  },  
  {  
    "ParameterName": "aurora_binlog_use_large_read_buffer",  
    "IsModifiable": true  
  },  
  {  
    "ParameterName": "aurora_lab_mode",  
    "IsModifiable": true  
  },  
  ...some output truncated...  
]
```

Example 4: To describe only the modifiable Boolean parameters in a DB cluster parameter group

The following `describe-db-cluster-parameters` example retrieves the names of only the parameters that you can modify in a DB cluster parameter group and that have a Boolean data type.

```
aws rds describe-db-cluster-parameters \  
  --db-cluster-parameter-group-name default.aurora-mysql5.7 \  
  --query 'Parameters[].{ParameterName:ParameterName,IsModifiable:IsModifiable} |  
  [?IsModifiable == `true` && ParameterType == `boolean`]'
```

```
--query 'Parameters[]'.
{ParameterName:ParameterName,DataType:DataType,IsModifiable:IsModifiable} | [?
DataType == `boolean`] | [?IsModifiable == `true`]'
```

Output:

```
[
  {
    "DataType": "boolean",
    "ParameterName": "aurora_binlog_use_large_read_buffer",
    "IsModifiable": true
  },
  {
    "DataType": "boolean",
    "ParameterName": "aurora_lab_mode",
    "IsModifiable": true
  },
  {
    "DataType": "boolean",
    "ParameterName": "autocommit",
    "IsModifiable": true
  },
  {
    "DataType": "boolean",
    "ParameterName": "automatic_sp_privileges",
    "IsModifiable": true
  },
  ...some output truncated...
]
```

For more information, see [Working with DB Parameter Groups and DB Cluster Parameter Groups](#) in the *Amazon Aurora User Guide*.

- For API details, see [DescribeDbClusterParameters](#) in *AWS CLI Command Reference*.

describe-db-cluster-snapshot-attributes

The following code example shows how to use `describe-db-cluster-snapshot-attributes`.

AWS CLI

To describe the attribute names and values for a DB cluster snapshot

The following `describe-db-cluster-snapshot-attributes` example retrieves details of the attribute names and values for the specified DB cluster snapshot.

```
aws rds describe-db-cluster-snapshot-attributes \  
  --db-cluster-snapshot-identifier myclustersnapshot
```

Output:

```
{  
  "DBClusterSnapshotAttributesResult": {  
    "DBClusterSnapshotIdentifier": "myclustersnapshot",  
    "DBClusterSnapshotAttributes": [  
      {  
        "AttributeName": "restore",  
        "AttributeValues": [  
          "123456789012"  
        ]  
      }  
    ]  
  }  
}
```

For more information, see [Sharing a DB Cluster Snapshot](#) in the *Amazon Aurora User Guide*.

- For API details, see [DescribeDbClusterSnapshotAttributes](#) in *AWS CLI Command Reference*.

describe-db-cluster-snapshots

The following code example shows how to use `describe-db-cluster-snapshots`.

AWS CLI

To describe a DB cluster snapshot for a DB cluster

The following `describe-db-cluster-snapshots` example retrieves the details for the DB cluster snapshots for the specified DB cluster.

```
aws rds describe-db-cluster-snapshots \  
  --db-cluster-identifier mydbcluster
```

Output:

```

{
  "DBClusterSnapshots": [
    {
      "AvailabilityZones": [
        "us-east-1a",
        "us-east-1b",
        "us-east-1e"
      ],
      "DBClusterSnapshotIdentifier": "myclustersnapshotcopy",
      "DBClusterIdentifier": "mydbcluster",
      "SnapshotCreateTime": "2019-06-04T09:16:42.649Z",
      "Engine": "aurora-mysql",
      "AllocatedStorage": 0,
      "Status": "available",
      "Port": 0,
      "VpcId": "vpc-6594f31c",
      "ClusterCreateTime": "2019-04-15T14:18:42.785Z",
      "MasterUsername": "myadmin",
      "EngineVersion": "5.7.mysql_aurora.2.04.2",
      "LicenseModel": "aurora-mysql",
      "SnapshotType": "manual",
      "PercentProgress": 100,
      "StorageEncrypted": true,
      "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/
AKIAIOSFODNN7EXAMPLE",
      "DBClusterSnapshotArn": "arn:aws:rds:us-east-1:814387698303:cluster-
snapshot:myclustersnapshotcopy",
      "IAMDatabaseAuthenticationEnabled": false
    },
    {
      "AvailabilityZones": [
        "us-east-1a",
        "us-east-1b",
        "us-east-1e"
      ],
      "DBClusterSnapshotIdentifier": "rds:mydbcluster-2019-06-20-09-16",
      "DBClusterIdentifier": "mydbcluster",
      "SnapshotCreateTime": "2019-06-20T09:16:26.569Z",
      "Engine": "aurora-mysql",
      "AllocatedStorage": 0,
      "Status": "available",
      "Port": 0,
      "VpcId": "vpc-6594f31c",

```

```

    "ClusterCreateTime": "2019-04-15T14:18:42.785Z",
    "MasterUsername": "myadmin",
    "EngineVersion": "5.7.mysql_aurora.2.04.2",
    "LicenseModel": "aurora-mysql",
    "SnapshotType": "automated",
    "PercentProgress": 100,
    "StorageEncrypted": true,
    "KmsKeyId": "arn:aws:kms:us-east-1:814387698303:key/
AKIAIOSFODNN7EXAMPLE",
    "DBClusterSnapshotArn": "arn:aws:rds:us-east-1:123456789012:cluster-
snapshot:rds:mydbcluster-2019-06-20-09-16",
    "IAMDatabaseAuthenticationEnabled": false
  }
]
}

```

For more information, see [Creating a DB Cluster Snapshot](#) in the *Amazon Aurora User Guide*.

- For API details, see [DescribeDbClusterSnapshots](#) in *AWS CLI Command Reference*.

describe-db-clusters

The following code example shows how to use `describe-db-clusters`.

AWS CLI

Example 1: To describe a DB cluster

The following `describe-db-clusters` example retrieves the details of the specified DB cluster.

```

aws rds describe-db-clusters \
  --db-cluster-identifier mydbcluster

```

Output:

```

{
  "DBClusters": [
    {
      "AllocatedStorage": 1,
      "AvailabilityZones": [
        "us-east-1a",
        "us-east-1b",

```

```
    "us-east-1e"
  ],
  "BackupRetentionPeriod": 1,
  "DatabaseName": "mydbcluster",
  "DBClusterIdentifier": "mydbcluster",
  "DBClusterParameterGroup": "default.aurora-mysql5.7",
  "DBSubnetGroup": "default",
  "Status": "available",
  "EarliestRestorableTime": "2019-06-19T09:16:28.210Z",
  "Endpoint": "mydbcluster.cluster-cnpeexample.us-
east-1.rds.amazonaws.com",
  "ReaderEndpoint": "mydbcluster.cluster-ro-cnpeexample.us-
east-1.rds.amazonaws.com",
  "MultiAZ": true,
  "Engine": "aurora-mysql",
  "EngineVersion": "5.7.mysql_aurora.2.04.2",
  "LatestRestorableTime": "2019-06-20T22:38:14.908Z",
  "Port": 3306,
  "MasterUsername": "myadmin",
  "PreferredBackupWindow": "09:09-09:39",
  "PreferredMaintenanceWindow": "sat:04:09-sat:04:39",
  "ReadReplicaIdentifiers": [],
  "DBClusterMembers": [
    {
      "DBInstanceIdentifier": "dbinstance3",
      "IsClusterWriter": false,
      "DBClusterParameterGroupStatus": "in-sync",
      "PromotionTier": 1
    },
    {
      "DBInstanceIdentifier": "dbinstance1",
      "IsClusterWriter": false,
      "DBClusterParameterGroupStatus": "in-sync",
      "PromotionTier": 1
    },
    {
      "DBInstanceIdentifier": "dbinstance2",
      "IsClusterWriter": false,
      "DBClusterParameterGroupStatus": "in-sync",
      "PromotionTier": 1
    },
    {
      "DBInstanceIdentifier": "mydbcluster",
      "IsClusterWriter": false,
```

```

        "DBClusterParameterGroupStatus": "in-sync",
        "PromotionTier": 1
    },
    {
        "DBInstanceIdentifier": "mydbcluster-us-east-1b",
        "IsClusterWriter": false,
        "DBClusterParameterGroupStatus": "in-sync",
        "PromotionTier": 1
    },
    {
        "DBInstanceIdentifier": "mydbcluster",
        "IsClusterWriter": true,
        "DBClusterParameterGroupStatus": "in-sync",
        "PromotionTier": 1
    }
],
"VpcSecurityGroups": [
    {
        "VpcSecurityGroupId": "sg-0b9130572daf3dc16",
        "Status": "active"
    }
],
"HostedZoneId": "Z2R2ITUGPM61AM",
"StorageEncrypted": true,
"KmsKeyId": "arn:aws:kms:us-east-1:814387698303:key/
AKIAIOSFODNN7EXAMPLE",
"DbClusterResourceId": "cluster-AKIAIOSFODNN7EXAMPLE",
"DBClusterArn": "arn:aws:rds:us-
east-1:123456789012:cluster:mydbcluster",
"AssociatedRoles": [],
"IAMDatabaseAuthenticationEnabled": false,
"ClusterCreateTime": "2019-04-15T14:18:42.785Z",
"EngineMode": "provisioned",
"DeletionProtection": false,
"HttpEndpointEnabled": false
}
]
}

```

Example 2: To list certain attributes of all DB clusters

The following `describe-db-clusters` example retrieves only the `DBClusterIdentifier`, `Endpoint`, and `ReaderEndpoint` attributes of all your DB clusters in the current AWS Region.

```
aws rds describe-db-clusters \
  --query 'DBClusters[.]
{DBClusterIdentifier:DBClusterIdentifier,Endpoint:Endpoint,ReaderEndpoint:ReaderEndpoint}'
```

Output:

```
[
  {
    "Endpoint": "cluster-57-2020-05-01-2270.cluster-cnpxample.us-
east-1.rds.amazonaws.com",
    "ReaderEndpoint": "cluster-57-2020-05-01-2270.cluster-ro-cnpxample.us-
east-1.rds.amazonaws.com",
    "DBClusterIdentifier": "cluster-57-2020-05-01-2270"
  },
  {
    "Endpoint": "cluster-57-2020-05-01-4615.cluster-cnpxample.us-
east-1.rds.amazonaws.com",
    "ReaderEndpoint": "cluster-57-2020-05-01-4615.cluster-ro-cnpxample.us-
east-1.rds.amazonaws.com",
    "DBClusterIdentifier": "cluster-57-2020-05-01-4615"
  },
  {
    "Endpoint": "pg2-cluster.cluster-cnpxample.us-east-1.rds.amazonaws.com",
    "ReaderEndpoint": "pg2-cluster.cluster-ro-cnpxample.us-
east-1.rds.amazonaws.com",
    "DBClusterIdentifier": "pg2-cluster"
  },
  ...output omitted...
]
```

Example 3: To list DB clusters with a specific attribute

The following `describe-db-clusters` example retrieves only the `DBClusterIdentifier` and `Engine` attributes of your DB clusters that use the `aurora-postgresql` DB engine.

```
aws rds describe-db-clusters \
  --query 'DBClusters[.]{DBClusterIdentifier:DBClusterIdentifier,Engine:Engine} |
[?Engine == `aurora-postgresql`]'
```

Output:


```
[
  {
    "Engine": "aurora-postgresql",
    "DBClusterIdentifier": "pg2-cluster"
  }
]
```

For more information, see [Amazon Aurora DB Clusters](#) in the *Amazon Aurora User Guide*.

- For API details, see [DescribeDbClusters](#) in *AWS CLI Command Reference*.

describe-db-engine-versions

The following code example shows how to use `describe-db-engine-versions`.

AWS CLI

To describe the DB engine versions for the MySQL DB engine

The following `describe-db-engine-versions` example displays details about each of the DB engine versions for the specified DB engine.

```
aws rds describe-db-engine-versions \
  --engine mysql
```

Output:

```
{
  "DBEngineVersions": [
    {
      "Engine": "mysql",
      "EngineVersion": "5.5.46",
      "DBParameterGroupFamily": "mysql5.5",
      "DBEngineDescription": "MySQL Community Edition",
      "DBEngineVersionDescription": "MySQL 5.5.46",
      "ValidUpgradeTarget": [
        {
          "Engine": "mysql",
          "EngineVersion": "5.5.53",
          "Description": "MySQL 5.5.53",
          "AutoUpgrade": false,
          "IsMajorVersionUpgrade": false
        }
      ]
    }
  ]
}
```

```

    },
    {
      "Engine": "mysql",
      "EngineVersion": "5.5.54",
      "Description": "MySQL 5.5.54",
      "AutoUpgrade": false,
      "IsMajorVersionUpgrade": false
    },
    {
      "Engine": "mysql",
      "EngineVersion": "5.5.57",
      "Description": "MySQL 5.5.57",
      "AutoUpgrade": false,
      "IsMajorVersionUpgrade": false
    },
    ...some output truncated...
  ]
}

```

For more information, see [What Is Amazon Relational Database Service \(Amazon RDS\)?](#) in the *Amazon RDS User Guide*.

- For API details, see [DescribeDBEngineVersions](#) in *AWS CLI Command Reference*.

describe-db-instance-automated-backups

The following code example shows how to use `describe-db-instance-automated-backups`.

AWS CLI

To describe the automated backups for a DB instance

The following `describe-db-instance-automated-backups` example displays details about the automated backups for the specified DB instance. The details include replicated automated backups in other AWS Regions.

```
aws rds describe-db-instance-automated-backups \
  --db-instance-identifier new-orcl-db
```

Output:

```
{
  "DBInstanceAutomatedBackups": [
```

```
{
  "DBInstanceArn": "arn:aws:rds:us-east-1:123456789012:db:new-orcl-db",
  "DbiResourceId": "db-JKIB2GFQ5RV7REPLZA4EXAMPLE",
  "Region": "us-east-1",
  "DBInstanceIdentifier": "new-orcl-db",
  "RestoreWindow": {
    "EarliestTime": "2020-12-07T21:05:20.939Z",
    "LatestTime": "2020-12-07T21:05:20.939Z"
  },
  "AllocatedStorage": 20,
  "Status": "replicating",
  "Port": 1521,
  "InstanceCreateTime": "2020-12-04T15:28:31Z",
  "MasterUsername": "admin",
  "Engine": "oracle-se2",
  "EngineVersion": "12.1.0.2.v21",
  "LicenseModel": "bring-your-own-license",
  "OptionGroupName": "default:oracle-se2-12-1",
  "Encrypted": false,
  "StorageType": "gp2",
  "IAMDatabaseAuthenticationEnabled": false,
  "BackupRetentionPeriod": 14,
  "DBInstanceAutomatedBackupsArn": "arn:aws:rds:us-
west-2:123456789012:auto-backup:ab-jkib2gfg5rv7replzadabrktni2bn4example"
}
]
```

For more information, see [Finding information about replicated backups](#) in the *Amazon RDS User Guide*.

- For API details, see [DescribeDbInstanceAutomatedBackups](#) in *AWS CLI Command Reference*.

describe-db-instances

The following code example shows how to use describe-db-instances.

AWS CLI

To describe a DB instance

The following describe-db-instances example retrieves details about the specified DB instance.

```
aws rds describe-db-instances \  
  --db-instance-identifier mydbinstancecf
```

Output:

```
{  
  "DBInstances": [  
    {  
      "DBInstanceIdentifier": "mydbinstancecf",  
      "DBInstanceClass": "db.t3.small",  
      "Engine": "mysql",  
      "DBInstanceStatus": "available",  
      "MasterUsername": "masterawsuser",  
      "Endpoint": {  
        "Address": "mydbinstancecf.abcxample.us-east-1.rds.amazonaws.com",  
        "Port": 3306,  
        "HostedZoneId": "Z2R2ITUGPM61AM"  
      },  
      "...some output truncated..."  
    }  
  ]  
}
```

- For API details, see [DescribeDBInstances](#) in *AWS CLI Command Reference*.

describe-db-log-files

The following code example shows how to use `describe-db-log-files`.

AWS CLI

To describe the log files for a DB instance

The following `describe-db-log-files` example retrieves details about the log files for the specified DB instance.

```
aws rds describe-db-log-files -\  
  --db-instance-identifier test-instance
```

Output:

```
{
  "DescribeDBLogFiles": [
    {
      "Size": 0,
      "LastWritten": 1533060000000,
      "LogFileName": "error/mysql-error-running.log"
    },
    {
      "Size": 2683,
      "LastWritten": 1532994300000,
      "LogFileName": "error/mysql-error-running.log.0"
    },
    {
      "Size": 107,
      "LastWritten": 1533057300000,
      "LogFileName": "error/mysql-error-running.log.18"
    },
    {
      "Size": 13105,
      "LastWritten": 1532991000000,
      "LogFileName": "error/mysql-error-running.log.23"
    },
    {
      "Size": 0,
      "LastWritten": 1533061200000,
      "LogFileName": "error/mysql-error.log"
    },
    {
      "Size": 3519,
      "LastWritten": 1532989252000,
      "LogFileName": "mysqlUpgrade"
    }
  ]
}
```

- For API details, see [DescribeDbLogFiles](#) in *AWS CLI Command Reference*.

describe-db-parameter-groups

The following code example shows how to use `describe-db-parameter-groups`.

AWS CLI

To describe your DB parameter group

The following `describe-db-parameter-groups` example retrieves details about your DB parameter groups.

```
aws rds describe-db-parameter-groups
```

Output:

```
{
  "DBParameterGroups": [
    {
      "DBParameterGroupName": "default.aurora-mysql5.7",
      "DBParameterGroupFamily": "aurora-mysql5.7",
      "Description": "Default parameter group for aurora-mysql5.7",
      "DBParameterGroupArn": "arn:aws:rds:us-east-1:123456789012:pg:default.aurora-mysql5.7"
    },
    {
      "DBParameterGroupName": "default.aurora-postgresql9.6",
      "DBParameterGroupFamily": "aurora-postgresql9.6",
      "Description": "Default parameter group for aurora-postgresql9.6",
      "DBParameterGroupArn": "arn:aws:rds:us-east-1:123456789012:pg:default.aurora-postgresql9.6"
    },
    {
      "DBParameterGroupName": "default.aurora5.6",
      "DBParameterGroupFamily": "aurora5.6",
      "Description": "Default parameter group for aurora5.6",
      "DBParameterGroupArn": "arn:aws:rds:us-east-1:123456789012:pg:default.aurora5.6"
    },
    {
      "DBParameterGroupName": "default.mariadb10.1",
      "DBParameterGroupFamily": "mariadb10.1",
      "Description": "Default parameter group for mariadb10.1",
      "DBParameterGroupArn": "arn:aws:rds:us-east-1:123456789012:pg:default.mariadb10.1"
    },
    ...some output truncated...
  ]
}
```

```
}
```

For more information, see [Working with DB Parameter Groups](#) in the *Amazon RDS User Guide*.

- For API details, see [DescribeDBParameterGroups](#) in *AWS CLI Command Reference*.

describe-db-parameters

The following code example shows how to use `describe-db-parameters`.

AWS CLI

To describe the parameters in a DB parameter group

The following `describe-db-parameters` example retrieves the details of the specified DB parameter group.

```
aws rds describe-db-parameters \  
  --db-parameter-group-name mydbpg
```

Output:

```
{  
  "Parameters": [  
    {  
      "ParameterName": "allow-suspicious-udfs",  
      "Description": "Controls whether user-defined functions that have only  
an xxx symbol for the main function can be loaded",  
      "Source": "engine-default",  
      "ApplyType": "static",  
      "DataType": "boolean",  
      "AllowedValues": "0,1",  
      "IsModifiable": false,  
      "ApplyMethod": "pending-reboot"  
    },  
    {  
      "ParameterName": "auto_generate_certs",  
      "Description": "Controls whether the server autogenerates SSL key and  
certificate files in the data directory, if they do not already exist.",  
      "Source": "engine-default",  
      "ApplyType": "static",  
      "DataType": "boolean",  
      "AllowedValues": "0,1",
```

```

        "IsModifiable": false,
        "ApplyMethod": "pending-reboot"
    },
    ...some output truncated...
]
}

```

For more information, see [Working with DB Parameter Groups](#) in the *Amazon RDS User Guide*.

- For API details, see [DescribeDBParameters](#) in *AWS CLI Command Reference*.

describe-db-proxies

The following code example shows how to use describe-db-proxies.

AWS CLI

To describe a DB proxy for an RDS database

The following describe-db-proxies example returns information about DB proxies.

```
aws rds describe-db-proxies
```

Output:

```

{
  "DBProxies": [
    {
      "DBProxyName": "proxyExample1",
      "DBProxyArn": "arn:aws:rds:us-east-1:123456789012:db-proxy:prx-0123a01b12345c0ab",
      "Status": "available",
      "EngineFamily": "PostgreSQL",
      "VpcId": "vpc-1234567",
      "VpcSecurityGroupIds": [
        "sg-1234"
      ],
      "VpcSubnetIds": [
        "subnetgroup1",
        "subnetgroup2"
      ],
      "Auth": "[
        {

```



```

        "Description": "proxydescription1"
        "AuthScheme": "SECRETS",
        "SecretArn": "arn:aws:secretsmanager:us-
west-2:123456789123:secret:secretName-1234f",
        "IAMAuth": "DISABLED"
    }
  ],
  "RoleArn": "arn:aws:iam::12345678912?:role/ProxyPostgreSQLRole",
  "Endpoint": "proxyExample1.proxy-ab0cd1efghij.us-
east-1.rds.amazonaws.com",
  "RequireTLS": false,
  "IdleClientTimeout": 1800,
  "DebuggingLogging": false,
  "CreateDate": "2023-04-05T16:09:33.452000+00:00",
  "UpdatedDate": "2023-04-13T01:49:38.568000+00:00"
},
{
  "DBProxyName": "proxyExample2",
  "DBProxyArn": "arn:aws:rds:us-east-1:123456789012:db-
proxy:prx-1234a12b23456c1ab",
  "Status": "available",
  "EngineFamily": "PostgreSQL",
  "VpcId": "sg-1234567",
  "VpcSecurityGroupIds": [
    "sg-1234"
  ],
  "VpcSubnetIds": [
    "subnetgroup1",
    "subnetgroup2"
  ],
  "Auth": "[
    {
      "Description": "proxydescription2"
      "AuthScheme": "SECRETS",
      "SecretArn": "arn:aws:secretsmanager:us-
west-2:123456789123:secret:secretName-1234f",
      "IAMAuth": "DISABLED"
    }
  ],
  "RoleArn": "arn:aws:iam::12345678912:role/ProxyPostgreSQLRole",
  "Endpoint": "proxyExample2.proxy-ab0cd1efghij.us-
east-1.rds.amazonaws.com",
  "RequireTLS": false,
  "IdleClientTimeout": 1800,

```

```
        "DebuggingLogging": false,  
        "CreateDate": "2022-01-05T16:19:33.452000+00:00",  
        "UpdateDate": "2023-04-13T01:49:38.568000+00:00"  
    }  
]  
}
```

For more information, see [Viewing an RDS Proxy](#) in the *Amazon RDS User Guide* and [Viewing an RDS Proxy](#) in the *Amazon Aurora User Guide*.

- For API details, see [DescribeDbProxies](#) in *AWS CLI Command Reference*.

describe-db-proxy-endpoints

The following code example shows how to use `describe-db-proxy-endpoints`.

AWS CLI

To describe a DB proxy endpoints

The following `describe-db-proxy-endpoints` example returns information about DB proxy endpoints.

```
aws rds describe-db-proxy-endpoints
```

Output:

```
{  
  "DBProxyEndpoints": [  
    {  
      "DBProxyEndpointName": "proxyEndpoint1",  
      "DBProxyEndpointArn": "arn:aws:rds:us-east-1:123456789012:db-proxy-  
endpoint:prx-endpoint-0123a01b12345c0ab",  
      "DBProxyName": "proxyExample",  
      "Status": "available",  
      "VpcId": "vpc-1234567",  
      "VpcSecurityGroupIds": [  
        "sg-1234"  
      ],  
      "VpcSubnetIds": [  
        "subnetgroup1",  
      ]  
    }  
  ]  
}
```

```

        "subnetgroup2"
    ],
    "Endpoint": "proxyEndpoint1.endpoint.proxy-ab0cd1efghij.us-
east-1.rds.amazonaws.com",
    "CreateDate": "2023-04-05T16:09:33.452000+00:00",
    "TargetRole": "READ_WRITE",
    "IsDefault": false
  },
  {
    "DBProxyEndpointName": "proxyEndpoint2",
    "DBProxyEndpointArn": "arn:aws:rds:us-east-1:123456789012:db-proxy-
endpoint:prx-endpoint-4567a01b12345c0ab",
    "DBProxyName": "proxyExample2",
    "Status": "available",
    "VpcId": "vpc1234567",
    "VpcSecurityGroupIds": [
      "sg-5678"
    ],
    "VpcSubnetIds": [
      "subnetgroup1",
      "subnetgroup2"
    ],
    "Endpoint": "proxyEndpoint2.endpoint.proxy-cd1ef2klmnop.us-
east-1.rds.amazonaws.com",
    "CreateDate": "2023-04-05T16:09:33.452000+00:00",
    "TargetRole": "READ_WRITE",
    "IsDefault": false
  }
]
}

```

For more information, see [Viewing a proxy endpoint](#) in the *Amazon RDS User Guide* and [Creating a proxy endpoint](#) in the *Amazon Aurora User Guide*.

- For API details, see [DescribeDbProxyEndpoints](#) in *AWS CLI Command Reference*.

describe-db-proxy-target-groups

The following code example shows how to use `describe-db-proxy-target-groups`.

AWS CLI

To describe a DB proxy endpoints

The following `describe-db-proxy-target-groups` example returns information about DB proxy target groups.

```
aws rds describe-db-proxy-target-groups \
  --db-proxy-name proxyExample
```

Output:

```
{
  "TargetGroups":
  {
    "DBProxyName": "proxyExample",
    "TargetGroupName": "default",
    "TargetGroupArn": "arn:aws:rds:us-east-1:123456789012:target-group:prx-
tg-0123a01b12345c0ab",
    "IsDefault": true,
    "Status": "available",
    "ConnectionPoolConfig": {
      "MaxConnectionsPercent": 100,
      "MaxIdleConnectionsPercent": 50,
      "ConnectionBorrowTimeout": 120,
      "SessionPinningFilters": []
    },
    "CreateDate": "2023-05-02T18:41:19.495000+00:00",
    "UpdateDate": "2023-05-02T18:41:21.762000+00:00"
  }
}
```

For more information, see [Viewing an RDS Proxy](#) in the *Amazon RDS User Guide* and [Viewing an RDS Proxy](#) in the *Amazon Aurora User Guide*.

- For API details, see [DescribeDbProxyTargetGroups](#) in *AWS CLI Command Reference*.

describe-db-proxy-targets

The following code example shows how to use `describe-db-proxy-targets`.

AWS CLI

To describe DB proxy targets

The following `describe-db-proxy-targets` example returns information about DB proxy targets.

```
aws rds describe-db-proxy-targets \
  --db-proxy-name proxyExample
```

Output:

```
{
  "Targets": [
    {
      "Endpoint": "database1.ab0cd1efghij.us-east-1.rds.amazonaws.com",
      "TrackedClusterId": "database1",
      "RdsResourceId": "database1-instance-1",
      "Port": 3306,
      "Type": "RDS_INSTANCE",
      "Role": "READ_WRITE",
      "TargetHealth": {
        "State": "UNAVAILABLE",
        "Reason": "PENDING_PROXY_CAPACITY",
        "Description": "DBProxy Target is waiting for proxy to scale to
desired capacity"
      }
    }
  ]
}
```

For more information, see [Viewing an RDS proxy](#) in the *Amazon RDS User Guide* and [Viewing an RDS proxy](#) in the *Amazon Aurora User Guide*.

- For API details, see [DescribeDbProxyTargets](#) in *AWS CLI Command Reference*.

describe-db-recommendations

The following code example shows how to use `describe-db-recommendations`.

AWS CLI

Example 1: To list all DB recommendations

The following `describe-db-recommendations` example lists all DB recommendations in your AWS account.

```
aws rds describe-db-recommendations
```

Output:

```
{
  "DBRecommendations": [
    {
      "RecommendationId": "12ab3cde-f456-7g8h-9012-i3j45678k9lm",
      "TypeId": "config_recommendation::old_minor_version",
      "Severity": "informational",
      "ResourceArn": "arn:aws:rds:us-west-2:111122223333:db:database-1",
      "Status": "active",
      "CreatedTime": "2024-02-21T23:14:19.292000+00:00",
      "UpdatedTime": "2024-02-21T23:14:19+00:00",
      "Detection": "***[resource-name]** is not running the latest minor DB
engine version",
      "Recommendation": "Upgrade to latest engine version",
      "Description": "Your database resources aren't running the latest minor
DB engine version. The latest minor version contains the latest security fixes and
other improvements.",
      "RecommendedActions": [
        {
          "ActionId": "12ab34c5de6fg7h89i0jk1lm234n5678",
          "Operation": "modifyDbInstance",
          "Parameters": [
            {
              "Key": "EngineVersion",
              "Value": "5.7.44"
            },
            {
              "Key": "DBInstanceIdentifier",
              "Value": "database-1"
            }
          ],
          "ApplyModes": [
            "immediately",
            "next-maintenance-window"
          ],
          "Status": "ready",
          "ContextAttributes": [
            {
              "Key": "Recommended value",
              "Value": "5.7.44"
            }
          ]
        }
      ]
    }
  ]
}
```

```
    },
    {
      "Key": "Current engine version",
      "Value": "5.7.42"
    }
  ]
}
],
"Category": "security",
"Source": "RDS",
"TypeDetection": "***[resource-count] resources** are not running the
latest minor DB engine version",
"TypeRecommendation": "Upgrade to latest engine version",
"Impact": "Reduced database performance and data security at risk",
"AdditionalInfo": "We recommend that you maintain your database with the
latest DB engine minor version as this version includes the latest security and
functionality fixes. The DB engine minor version upgrades contain only the changes
which are backward-compatible with earlier minor versions of the same major version
of the DB engine.",
"Links": [
  {
    "Text": "Upgrading an RDS DB instance engine version",
    "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/
USER_UpgradeDBInstance.Upgrading.html"
  },
  {
    "Text": "Using Amazon RDS Blue/Green Deployments for database
updates for Amazon Aurora",
    "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/
AuroraUserGuide/blue-green-deployments.html"
  },
  {
    "Text": "Using Amazon RDS Blue/Green Deployments for database
updates for Amazon RDS",
    "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/
blue-green-deployments.html"
  }
]
}
]
```

For more information, see [Viewing and responding to Amazon RDS recommendations](#) in the *Amazon RDS User Guide* and [Viewing and responding to Amazon RDS recommendations](#) in the *Amazon Aurora User Guide*.

Example 2: To list high severity DB recommendations

The following `describe-db-recommendations` example lists high severity DB recommendations in your AWS account.

```
aws rds describe-db-recommendations \  
  --filters Name=severity,Values=high
```

Output:

```
{  
  "DBRecommendations": [  
    {  
      "RecommendationId": "12ab3cde-f456-7g8h-9012-i3j45678k9lm",  
      "TypeId": "config_recommendation::rds_extended_support",  
      "Severity": "high",  
      "ResourceArn": "arn:aws:rds:us-west-2:111122223333:db:database-1",  
      "Status": "active",  
      "CreatedTime": "2024-02-21T23:14:19.392000+00:00",  
      "UpdatedTime": "2024-02-21T23:14:19+00:00",  
      "Detection": "Your databases will be auto-enrolled to RDS Extended  
Support on February 29",  
      "Recommendation": "Upgrade your major version before February 29, 2024  
to avoid additional charges",  
      "Description": "Your PostgreSQL 11 and MySQL 5.7 databases will be  
automatically enrolled into RDS Extended Support on February 29, 2024. To avoid  
the increase in charges due to RDS Extended Support, we recommend upgrading your  
databases to a newer major engine version before February 29, 2024.\n\nTo learn more  
about the RDS Extended Support pricing, refer to the pricing page.",  
      "RecommendedActions": [  
        {  
          "ActionId": "12ab34c5de6fg7h89i0jk1lm234n5678",  
          "Parameters": [],  
          "ApplyModes": [  
            "manual"  
          ],  
          "Status": "ready",  
          "ContextAttributes": []  
        }  
      ]  
    }  
  ]  
}
```



```

    ],
    "Category": "cost optimization",
    "Source": "RDS",
    "TypeDetection": "Your database will be auto-enrolled to RDS Extended
Support on February 29",
    "TypeRecommendation": "Upgrade your major version before February 29,
2024 to avoid additional charges",
    "Impact": "Increase in charges due to RDS Extended Support",
    "AdditionalInfo": "With Amazon RDS Extended Support, you can continue
running your database on a major engine version past the RDS end of standard
support date for an additional cost. This paid feature gives you more time to
upgrade to a supported major engine version.\nDuring Extended Support, Amazon RDS
will supply critical CVE patches and bug fixes.",
    "Links": [
        {
            "Text": "Amazon RDS Extended Support pricing for RDS for MySQL",
            "Url": "https://aws.amazon.com/rds/mysql/pricing/"
        },
        {
            "Text": "Amazon RDS Extended Support for RDS for MySQL and
PostgreSQL databases",
            "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/
extended-support.html"
        },
        {
            "Text": "Amazon RDS Extended Support pricing for Amazon Aurora
PostgreSQL",
            "Url": "https://aws.amazon.com/rds/aurora/pricing/"
        },
        {
            "Text": "Amazon RDS Extended Support for Aurora PostgreSQL
databases",
            "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/
AuroraUserGuide/extended-support.html"
        },
        {
            "Text": "Amazon RDS Extended Support pricing for RDS for
PostgreSQL",
            "Url": "https://aws.amazon.com/rds/postgresql/pricing/"
        }
    ]
}
]

```

```
}
```

For more information, see [Viewing and responding to Amazon RDS recommendations](#) in the *Amazon RDS User Guide* and [Viewing and responding to Amazon RDS recommendations](#) in the *Amazon Aurora User Guide*.

Example 3: To list DB recommendations for a specified DB instance

The following `describe-db-recommendations` example lists all DB recommendations for a specified DB instance.

```
aws rds describe-db-recommendations \  
  --filters Name=dbi-resource-id,Values=database-1
```

Output:

```
{  
  "DBRecommendations": [  
    {  
      "RecommendationId": "12ab3cde-f456-7g8h-9012-i3j45678k9lm",  
      "TypeId": "config_recommendation::old_minor_version",  
      "Severity": "informational",  
      "ResourceArn": "arn:aws:rds:us-west-2:111122223333:db:database-1",  
      "Status": "active",  
      "CreatedTime": "2024-02-21T23:14:19.292000+00:00",  
      "UpdatedTime": "2024-02-21T23:14:19+00:00",  
      "Detection": "***[resource-name]** is not running the latest minor DB  
engine version",  
      "Recommendation": "Upgrade to latest engine version",  
      "Description": "Your database resources aren't running the latest minor  
DB engine version. The latest minor version contains the latest security fixes and  
other improvements.",  
      "RecommendedActions": [  
        {  
          "ActionId": "12ab34c5de6fg7h89i0jk1lm234n5678",  
          "Operation": "modifyDbInstance",  
          "Parameters": [  
            {  
              "Key": "EngineVersion",  
              "Value": "5.7.44"  
            },  
            {  
              "Key": "DBInstanceIdentifier",
```

```

        "Value": "database-1"
      }
    ],
    "ApplyModes": [
      "immediately",
      "next-maintenance-window"
    ],
    "Status": "ready",
    "ContextAttributes": [
      {
        "Key": "Recommended value",
        "Value": "5.7.44"
      },
      {
        "Key": "Current engine version",
        "Value": "5.7.42"
      }
    ]
  }
],
"Category": "security",
"Source": "RDS",
"TypeDetection": "**[resource-count] resources** are not running the
latest minor DB engine version",
"TypeRecommendation": "Upgrade to latest engine version",
"Impact": "Reduced database performance and data security at risk",
"AdditionalInfo": "We recommend that you maintain your database with the
latest DB engine minor version as this version includes the latest security and
functionality fixes. The DB engine minor version upgrades contain only the changes
which are backward-compatible with earlier minor versions of the same major version
of the DB engine.",
"Links": [
  {
    "Text": "Upgrading an RDS DB instance engine version",
    "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/
USER_UpgradeDBInstance.Upgrading.html"
  },
  {
    "Text": "Using Amazon RDS Blue/Green Deployments for database
updates for Amazon Aurora",
    "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/
AuroraUserGuide/blue-green-deployments.html"
  }
]

```

```

        "Text": "Using Amazon RDS Blue/Green Deployments for database
updates for Amazon RDS",
        "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/
blue-green-deployments.html"
    }
]
}
}
}

```

For more information, see [Viewing and responding to Amazon RDS recommendations](#) in the *Amazon RDS User Guide* and [Viewing and responding to Amazon RDS recommendations](#) in the *Amazon Aurora User Guide*.

Example 4: To list all active DB recommendations

The following `describe-db-recommendations` example lists all active DB recommendations in your AWS account.

```
aws rds describe-db-recommendations \
  --filters Name=status,Values=active
```

Output:

```

{
  "DBRecommendations": [
    {
      "RecommendationId": "12ab3cde-f456-7g8h-9012-i3j45678k9lm",
      "TypeId": "config_recommendation::old_minor_version",
      "Severity": "informational",
      "ResourceArn": "arn:aws:rds:us-west-2:111122223333:db:database-1",
      "Status": "active",
      "CreatedTime": "2024-02-21T23:14:19.292000+00:00",
      "UpdatedTime": "2024-02-21T23:14:19+00:00",
      "Detection": "***[resource-name]** is not running the latest minor DB
engine version",
      "Recommendation": "Upgrade to latest engine version",
      "Description": "Your database resources aren't running the latest minor
DB engine version. The latest minor version contains the latest security fixes and
other improvements.",
      "RecommendedActions": [
        {
          "ActionId": "12ab34c5de6fg7h89i0jk1lm234n5678",

```

```
    "Operation": "modifyDbInstance",
    "Parameters": [
      {
        "Key": "EngineVersion",
        "Value": "5.7.44"
      },
      {
        "Key": "DBInstanceIdentifier",
        "Value": "database-1"
      }
    ],
    "ApplyModes": [
      "immediately",
      "next-maintenance-window"
    ],
    "Status": "ready",
    "ContextAttributes": [
      {
        "Key": "Recommended value",
        "Value": "5.7.44"
      },
      {
        "Key": "Current engine version",
        "Value": "5.7.42"
      }
    ]
  }
],
"Category": "security",
"Source": "RDS",
"TypeDetection": "***[resource-count] resources** are not running the latest minor DB engine version",
"TypeRecommendation": "Upgrade to latest engine version",
"Impact": "Reduced database performance and data security at risk",
"AdditionalInfo": "We recommend that you maintain your database with the latest DB engine minor version as this version includes the latest security and functionality fixes. The DB engine minor version upgrades contain only the changes which are backward-compatible with earlier minor versions of the same major version of the DB engine.",
"Links": [
  {
    "Text": "Upgrading an RDS DB instance engine version",
    "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_UpgradeDBInstance.Upgrading.html"
  }
]
```

```
    },
    {
      "Text": "Using Amazon RDS Blue/Green Deployments for database
updates for Amazon Aurora",
      "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/
AuroraUserGuide/blue-green-deployments.html"
    },
    {
      "Text": "Using Amazon RDS Blue/Green Deployments for database
updates for Amazon RDS",
      "Url": "https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/
blue-green-deployments.html"
    }
  ]
}
]
```

For more information, see [Viewing and responding to Amazon RDS recommendations](#) in the *Amazon RDS User Guide* and [Viewing and responding to Amazon RDS recommendations](#) in the *Amazon Aurora User Guide*.

- For API details, see [DescribeDbRecommendations](#) in *AWS CLI Command Reference*.

describe-db-security-groups

The following code example shows how to use `describe-db-security-groups`.

AWS CLI

To list DB security groups

The following `describe-db-security-groups` example lists DB security groups.

```
aws rds describe-db-security-groups
```

Output:

```
{
  "DBSecurityGroups": [
    {
      "OwnerId": "123456789012",
```

```

        "DBSecurityGroupName": "default",
        "DBSecurityGroupDescription": "default",
        "EC2SecurityGroups": [],
        "IPRanges": [],
        "DBSecurityGroupArn": "arn:aws:rds:us-
west-1:111122223333:secgrp:default"
    },
    {
        "OwnerId": "123456789012",
        "DBSecurityGroupName": "mysecgroup",
        "DBSecurityGroupDescription": "My Test Security Group",
        "VpcId": "vpc-1234567f",
        "EC2SecurityGroups": [],
        "IPRanges": [],
        "DBSecurityGroupArn": "arn:aws:rds:us-
west-1:111122223333:secgrp:mysecgroup"
    }
]
}

```

For more information, see [Listing Available DB Security Groups](#) in the *Amazon RDS User Guide*.

- For API details, see [DescribeDbSecurityGroups](#) in *AWS CLI Command Reference*.

describe-db-shard-groups

The following code example shows how to use `describe-db-shard-groups`.

AWS CLI

Example 1: To describe DB shard groups

The following `describe-db-shard-groups` example retrieves the details of your DB shard groups.

```
aws rds describe-db-shard-groups
```

Output:

```

{
  "DBShardGroups": [
    {

```

```

    "DBShardGroupResourceId": "shardgroup-7bb446329da94788b3f957746example",
    "DBShardGroupIdentifier": "limitless-test-shard-grp",
    "DBClusterIdentifier": "limitless-test-cluster",
    "MaxACU": 768.0,
    "ComputeRedundancy": 0,
    "Status": "available",
    "PubliclyAccessible": true,
    "Endpoint": "limitless-test-cluster.limitless-cekyceexample.us-
east-2.rds.amazonaws.com"
  },
  {
    "DBShardGroupResourceId": "shardgroup-a6e3a0226aa243e2ac6c7a1234567890",
    "DBShardGroupIdentifier": "my-db-shard-group",
    "DBClusterIdentifier": "my-sv2-cluster",
    "MaxACU": 768.0,
    "ComputeRedundancy": 0,
    "Status": "available",
    "PubliclyAccessible": false,
    "Endpoint": "my-sv2-cluster.limitless-cekyceexample.us-
east-2.rds.amazonaws.com"
  }
]
}

```

For more information, see [Amazon Aurora DB Clusters](#) in the *Amazon Aurora User Guide*.

- For API details, see [DescribeDbShardGroups](#) in *AWS CLI Command Reference*.

describe-db-snapshot-attributes

The following code example shows how to use `describe-db-snapshot-attributes`.

AWS CLI

To describe the attribute names and values for a DB snapshot

The following `describe-db-snapshot-attributes` example describes the attribute names and values for a DB snapshot.

```
aws rds describe-db-snapshot-attributes \
  --db-snapshot-identifier mydbsnapshot
```

Output:


```
{
  "DBSnapshotAttributesResult": {
    "DBSnapshotIdentifier": "mydbsnapshot",
    "DBSnapshotAttributes": [
      {
        "AttributeName": "restore",
        "AttributeValues": [
          "123456789012",
          "210987654321"
        ]
      }
    ]
  }
}
```

For more information, see [Sharing a DB Snapshot](#) in the *Amazon RDS User Guide*.

- For API details, see [DescribeDbSnapshotAttributes](#) in *AWS CLI Command Reference*.

describe-db-snapshots

The following code example shows how to use `describe-db-snapshots`.

AWS CLI

Example 1: To describe a DB snapshot for a DB instance

The following `describe-db-snapshots` example retrieves the details of a DB snapshot for a DB instance.

```
aws rds describe-db-snapshots \
  --db-snapshot-identifier mydbsnapshot
```

Output:

```
{
  "DBSnapshots": [
    {
      "DBSnapshotIdentifier": "mydbsnapshot",
      "DBInstanceIdentifier": "mysqldb",
      "SnapshotCreateTime": "2018-02-08T22:28:08.598Z",
      "Engine": "mysql",
    }
  ]
}
```

```

    "AllocatedStorage": 20,
    "Status": "available",
    "Port": 3306,
    "AvailabilityZone": "us-east-1f",
    "VpcId": "vpc-6594f31c",
    "InstanceCreateTime": "2018-02-08T22:24:55.973Z",
    "MasterUsername": "mysqladmin",
    "EngineVersion": "5.6.37",
    "LicenseModel": "general-public-license",
    "SnapshotType": "manual",
    "OptionGroupName": "default:mysql-5-6",
    "PercentProgress": 100,
    "StorageType": "gp2",
    "Encrypted": false,
    "DBSnapshotArn": "arn:aws:rds:us-
east-1:123456789012:snapshot:mydbsnapshot",
    "IAMDatabaseAuthenticationEnabled": false,
    "ProcessorFeatures": [],
    "DbiResourceId": "db-AKIAIOSFODNN7EXAMPLE"
  }
]
}

```

For more information, see [Creating a DB Snapshot](#) in the *Amazon RDS User Guide*.

Example 2: To find the number of manual snapshots taken

The following `describe-db-snapshots` example uses the `length` operator in the `--query` option to return the number of manual snapshots that have been taken in a particular AWS Region.

```

aws rds describe-db-snapshots \
  --snapshot-type manual \
  --query "length(*[].[DBSnapshots:SnapshotType])" \
  --region eu-central-1

```

Output:

```
35
```

For more information, see [Creating a DB Snapshot](#) in the *Amazon RDS User Guide*.

- For API details, see [DescribeDBSnapshots](#) in *AWS CLI Command Reference*.

describe-db-subnet-groups

The following code example shows how to use `describe-db-subnet-groups`.

AWS CLI

To describe a DB subnet group

The following `describe-db-subnet-groups` example retrieves the details of the specified DB subnet group.

```
aws rds describe-db-subnet-groups
```

Output:

```
{
  "DBSubnetGroups": [
    {
      "DBSubnetGroupName": "mydbsubnetgroup",
      "DBSubnetGroupDescription": "My DB Subnet Group",
      "VpcId": "vpc-971c12ee",
      "SubnetGroupStatus": "Complete",
      "Subnets": [
        {
          "SubnetIdentifier": "subnet-d8c8e7f4",
          "SubnetAvailabilityZone": {
            "Name": "us-east-1a"
          },
          "SubnetStatus": "Active"
        },
        {
          "SubnetIdentifier": "subnet-718fdc7d",
          "SubnetAvailabilityZone": {
            "Name": "us-east-1f"
          },
          "SubnetStatus": "Active"
        },
        {
          "SubnetIdentifier": "subnet-cbc8e7e7",
          "SubnetAvailabilityZone": {
            "Name": "us-east-1a"
          },
          "SubnetStatus": "Active"
        }
      ]
    }
  ]
}
```

```
    },
    {
      "SubnetIdentifier": "subnet-0ccde220",
      "SubnetAvailabilityZone": {
        "Name": "us-east-1a"
      },
      "SubnetStatus": "Active"
    }
  ],
  "DBSubnetGroupArn": "arn:aws:rds:us-
east-1:123456789012:subgrp:mydbsubnetgroup"
}
]
```

For more information, see [Amazon Virtual Private Cloud VPCs and Amazon RDS](#) in the *Amazon RDS User Guide*.

- For API details, see [DescribeDbSubnetGroups](#) in *AWS CLI Command Reference*.

describe-engine-default-cluster-parameters

The following code example shows how to use `describe-engine-default-cluster-parameters`.

AWS CLI

To describe the default engine and system parameter information for the Aurora database engine

The following `describe-engine-default-cluster-parameters` example retrieves the details of the default engine and system parameter information for Aurora DB clusters with MySQL 5.7 compatibility.

```
aws rds describe-engine-default-cluster-parameters \
  --db-parameter-group-family aurora-mysql5.7
```

Output:

```
{
  "EngineDefaults": {
    "Parameters": [
```

```

    {
      "ParameterName": "aurora_load_from_s3_role",
      "Description": "IAM role ARN used to load data from AWS S3",
      "Source": "engine-default",
      "ApplyType": "dynamic",
      "DataType": "string",
      "IsModifiable": true,
      "SupportedEngineModes": [
        "provisioned"
      ]
    },
    ...some output truncated...
  ]
}

```

For more information, see [Working with DB Parameter Groups and DB Cluster Parameter Groups](#) in the *Amazon Aurora User Guide*.

- For API details, see [DescribeEngineDefaultClusterParameters](#) in *AWS CLI Command Reference*.

describe-engine-default-parameters

The following code example shows how to use `describe-engine-default-parameters`.

AWS CLI

To describe the default engine and system parameter information for the database engine

The following `describe-engine-default-parameters` example retrieves details for the default engine and system parameter information for MySQL 5.7 DB instances.

```
aws rds describe-engine-default-parameters \
  --db-parameter-group-family mysql5.7
```

Output:

```

{
  "EngineDefaults": {
    "Parameters": [
      {
        "ParameterName": "allow-suspicious-udfs",

```

```

        "Description": "Controls whether user-defined functions that have
only an xxx symbol for the main function can be loaded",
        "Source": "engine-default",
        "ApplyType": "static",
        "DataType": "boolean",
        "AllowedValues": "0,1",
        "IsModifiable": false
    },
    ...some output truncated...
]
}
}

```

For more information, see [Working with DB Parameter Groups](#) in the *Amazon RDS User Guide*.

- For API details, see [DescribeEngineDefaultParameters](#) in *AWS CLI Command Reference*.

describe-event-categories

The following code example shows how to use `describe-event-categories`.

AWS CLI

To describe event categories

The following `describe-event-categories` example retrieves details about the event categories for all available event sources.

```
aws rds describe-event-categories
```

Output:

```

{
  "EventCategoriesMapList": [
    {
      "SourceType": "db-instance",
      "EventCategories": [
        "deletion",
        "read replica",
        "failover",
        "restoration",
        "maintenance",
        "low storage",

```

```
        "configuration change",
        "backup",
        "creation",
        "availability",
        "recovery",
        "failure",
        "backtrack",
        "notification"
    ]
},
{
    "SourceType": "db-security-group",
    "EventCategories": [
        "configuration change",
        "failure"
    ]
},
{
    "SourceType": "db-parameter-group",
    "EventCategories": [
        "configuration change"
    ]
},
{
    "SourceType": "db-snapshot",
    "EventCategories": [
        "deletion",
        "creation",
        "restoration",
        "notification"
    ]
},
{
    "SourceType": "db-cluster",
    "EventCategories": [
        "failover",
        "failure",
        "notification"
    ]
},
{
    "SourceType": "db-cluster-snapshot",
    "EventCategories": [
        "backup"
```

```
    ]
  }
]
}
```

- For API details, see [DescribeEventCategories](#) in *AWS CLI Command Reference*.

describe-event-subscriptions

The following code example shows how to use `describe-event-subscriptions`.

AWS CLI

To describe event subscriptions

This example describes all of the Amazon RDS event subscriptions for the current AWS account.

```
aws rds describe-event-subscriptions
```

Output:

```
{
  "EventSubscriptionsList": [
    {
      "EventCategoriesList": [
        "backup",
        "recovery"
      ],
      "Enabled": true,
      "EventSubscriptionArn": "arn:aws:rds:us-east-1:123456789012:es:my-
instance-events",
      "Status": "creating",
      "SourceType": "db-instance",
      "CustomerAwsId": "123456789012",
      "SubscriptionCreationTime": "2018-07-31 23:22:01.893",
      "CustSubscriptionId": "my-instance-events",
      "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:interesting-events"
    },
    ...some output truncated...
  ]
}
```


- For API details, see [DescribeEventSubscriptions](#) in *AWS CLI Command Reference*.

describe-events

The following code example shows how to use describe-events.

AWS CLI

To describe events

The following describe-events example retrieves details for the events that have occurred for the specified DB instance.

```
aws rds describe-events \  
  --source-identifier test-instance \  
  --source-type db-instance
```

Output:

```
{  
  "Events": [  
    {  
      "SourceType": "db-instance",  
      "SourceIdentifier": "test-instance",  
      "EventCategories": [  
        "backup"  
      ],  
      "Message": "Backing up DB instance",  
      "Date": "2018-07-31T23:09:23.983Z",  
      "SourceArn": "arn:aws:rds:us-east-1:123456789012:db:test-instance"  
    },  
    {  
      "SourceType": "db-instance",  
      "SourceIdentifier": "test-instance",  
      "EventCategories": [  
        "backup"  
      ],  
      "Message": "Finished DB Instance backup",  
      "Date": "2018-07-31T23:15:13.049Z",  
      "SourceArn": "arn:aws:rds:us-east-1:123456789012:db:test-instance"  
    }  
  ]  
}
```

```
}
```

- For API details, see [DescribeEvents](#) in *AWS CLI Command Reference*.

describe-export-tasks

The following code example shows how to use describe-export-tasks.

AWS CLI

To describe snapshot export tasks

The following describe-export-tasks example returns information about snapshot exports to Amazon S3.

```
aws rds describe-export-tasks
```

Output:

```
{
  "ExportTasks": [
    {
      "ExportTaskIdentifier": "test-snapshot-export",
      "SourceArn": "arn:aws:rds:us-west-2:123456789012:snapshot:test-
snapshot",
      "SnapshotTime": "2020-03-02T18:26:28.163Z",
      "TaskStartTime": "2020-03-02T18:57:56.896Z",
      "TaskEndTime": "2020-03-02T19:10:31.985Z",
      "S3Bucket": "mybucket",
      "S3Prefix": "",
      "IamRoleArn": "arn:aws:iam::123456789012:role/service-role/ExportRole",
      "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/
abcd0000-7fca-4128-82f2-aabbccddeeff",
      "Status": "COMPLETE",
      "PercentProgress": 100,
      "TotalExtractedDataInGB": 0
    },
    {
      "ExportTaskIdentifier": "my-s3-export",
      "SourceArn": "arn:aws:rds:us-west-2:123456789012:snapshot:db5-snapshot-
test",
      "SnapshotTime": "2020-03-27T20:48:42.023Z",
```

```

        "S3Bucket": "mybucket",
        "S3Prefix": "",
        "IamRoleArn": "arn:aws:iam::123456789012:role/service-role/ExportRole",
        "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/
abcd0000-7fca-4128-82f2-aabbccddeeff",
        "Status": "STARTING",
        "PercentProgress": 0,
        "TotalExtractedDataInGB": 0
    }
]
}

```

For more information, see [Monitoring Snapshot Exports](#) in the *Amazon RDS User Guide*.

- For API details, see [DescribeExportTasks](#) in *AWS CLI Command Reference*.

describe-global-clusters

The following code example shows how to use `describe-global-clusters`.

AWS CLI

To describe global DB clusters

The following `describe-global-clusters` example lists Aurora global DB clusters in the current AWS Region.

```
aws rds describe-global-clusters
```

Output:

```

{
  "GlobalClusters": [
    {
      "GlobalClusterIdentifier": "myglobalcluster",
      "GlobalClusterResourceId": "cluster-f5982077e3b5aabb",
      "GlobalClusterArn": "arn:aws:rds::123456789012:global-
cluster:myglobalcluster",
      "Status": "available",
      "Engine": "aurora-mysql",
      "EngineVersion": "5.7.mysql_aurora.2.07.2",
      "StorageEncrypted": false,
      "DeletionProtection": false,

```

```

        "GlobalClusterMembers": []
      }
    ]
  }

```

For more information, see [Managing an Aurora global database](#) in the *Amazon Aurora User Guide*.

- For API details, see [DescribeGlobalClusters](#) in *AWS CLI Command Reference*.

describe-option-group-options

The following code example shows how to use describe-option-group-options.

AWS CLI

To describe all available options

The following describe-option-group-options example lists two options for an Oracle Database 19c instance.

```

aws rds describe-option-group-options \
  --engine-name oracle-ee \
  --major-engine-version 19 \
  --max-items 2

```

Output:

```

{
  "OptionGroupOptions": [
    {
      "Name": "APEX",
      "Description": "Oracle Application Express Runtime Environment",
      "EngineName": "oracle-ee",
      "MajorEngineVersion": "19",
      "MinimumRequiredMinorEngineVersion": "0.0.0.ru-2019-07.rur-2019-07.r1",
      "PortRequired": false,
      "OptionsDependedOn": [],
      "OptionsConflictsWith": [],
      "Persistent": false,
      "Permanent": false,
      "RequiresAutoMinorEngineVersionUpgrade": false,

```

```

    "VpcOnly": false,
    "SupportsOptionVersionDowngrade": false,
    "OptionGroupOptionSettings": [],
    "OptionGroupOptionVersions": [
      {
        "Version": "19.1.v1",
        "IsDefault": true
      },
      {
        "Version": "19.2.v1",
        "IsDefault": false
      }
    ]
  },
  {
    "Name": "APEX-DEV",
    "Description": "Oracle Application Express Development Environment",
    "EngineName": "oracle-ee",
    "MajorEngineVersion": "19",
    "MinimumRequiredMinorEngineVersion": "0.0.0.ru-2019-07.rur-2019-07.r1",
    "PortRequired": false,
    "OptionsDependedOn": [
      "APEX"
    ],
    "OptionsConflictsWith": [],
    "Persistent": false,
    "Permanent": false,
    "RequiresAutoMinorEngineVersionUpgrade": false,
    "VpcOnly": false,
    "OptionGroupOptionSettings": []
  }
],
"NextToken": "eyJNYXJrZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAyfQ=="
}

```

For more information, see [Listing the Options and Option Settings for an Option Group](#) in the *Amazon RDS User Guide*.

- For API details, see [DescribeOptionGroupOptions](#) in *AWS CLI Command Reference*.

describe-option-groups

The following code example shows how to use describe-option-groups.

AWS CLI

To describe the available option groups

The following `describe-option-groups` example lists the options groups for an Oracle Database 19c instance.

```
aws rds describe-option-groups \  
  --engine-name oracle-ee \  
  --major-engine-version 19
```

Output:

```
{  
  "OptionGroupsList": [  
    {  
      "OptionGroupName": "default:oracle-ee-19",  
      "OptionGroupDescription": "Default option group for oracle-ee 19",  
      "EngineName": "oracle-ee",  
      "MajorEngineVersion": "19",  
      "Options": [],  
      "AllowsVpcAndNonVpcInstanceMemberships": true,  
      "OptionGroupArn": "arn:aws:rds:us-west-1:111122223333:og:default:oracle-  
ee-19"  
    }  
  ]  
}
```

For more information, see [Listing the Options and Option Settings for an Option Group](#) in the *Amazon RDS User Guide*.

- For API details, see [DescribeOptionGroups](#) in *AWS CLI Command Reference*.

describe-orderable-db-instance-options

The following code example shows how to use `describe-orderable-db-instance-options`.

AWS CLI

To describe orderable DB instance options

The following `describe-orderable-db-instance-options` example retrieves details about the orderable options for a DB instances running the MySQL DB engine.

```
aws rds describe-orderable-db-instance-options \  
--engine mysql
```

Output:

```
{  
  "OrderableDBInstanceOptions": [  
    {  
      "MinStorageSize": 5,  
      "ReadReplicaCapable": true,  
      "MaxStorageSize": 6144,  
      "AvailabilityZones": [  
        {  
          "Name": "us-east-1a"  
        },  
        {  
          "Name": "us-east-1b"  
        },  
        {  
          "Name": "us-east-1c"  
        },  
        {  
          "Name": "us-east-1d"  
        }  
      ],  
      "SupportsIops": false,  
      "AvailableProcessorFeatures": [],  
      "MultiAZCapable": true,  
      "DBInstanceClass": "db.m1.large",  
      "Vpc": true,  
      "StorageType": "gp2",  
      "LicenseModel": "general-public-license",  
      "EngineVersion": "5.5.46",  
      "SupportsStorageEncryption": false,  
      "SupportsEnhancedMonitoring": true,  
      "Engine": "mysql",  
      "SupportsIAMDatabaseAuthentication": false,  
      "SupportsPerformanceInsights": false  
    }  
  ]  
  ...some output truncated...  
}
```

- For API details, see [DescribeOrderableDBInstanceOptions](#) in *AWS CLI Command Reference*.

describe-pending-maintenance-actions

The following code example shows how to use `describe-pending-maintenance-actions`.

AWS CLI

To list resources with at least one pending maintenance action

The following `describe-pending-maintenance-actions` example lists the pending maintenance action for a DB instance.

```
aws rds describe-pending-maintenance-actions
```

Output:

```
{
  "PendingMaintenanceActions": [
    {
      "ResourceIdentifier": "arn:aws:rds:us-
west-2:123456789012:cluster:global-db1-cl1",
      "PendingMaintenanceActionDetails": [
        {
          "Action": "system-update",
          "Description": "Upgrade to Aurora PostgreSQL 2.4.2"
        }
      ]
    }
  ]
}
```

For more information, see [Maintaining a DB Instance](#) in the *Amazon RDS User Guide*.

- For API details, see [DescribePendingMaintenanceActions](#) in *AWS CLI Command Reference*.

describe-reserved-db-instances-offerings

The following code example shows how to use `describe-reserved-db-instances-offerings`.

AWS CLI

To describe reserved DB instance offerings

The following `describe-reserved-db-instances-offerings` example retrieves details about reserved DB instance options for oracle.

```
aws rds describe-reserved-db-instances-offerings \  
  --product-description oracle
```

Output:

```
{  
  "ReservedDBInstancesOfferings": [  
    {  
      "CurrencyCode": "USD",  
      "UsagePrice": 0.0,  
      "ProductDescription": "oracle-se2(li)",  
      "ReservedDBInstancesOfferingId": "005bdee3-9ef4-4182-aa0c-58ef7cb6c2f8",  
      "MultiAZ": true,  
      "DBInstanceClass": "db.m4.xlarge",  
      "OfferingType": "Partial Upfront",  
      "RecurringCharges": [  
        {  
          "RecurringChargeAmount": 0.594,  
          "RecurringChargeFrequency": "Hourly"  
        }  
      ],  
      "FixedPrice": 4089.0,  
      "Duration": 31536000  
    },  
    ...some output truncated...  
  ]  
}
```

- For API details, see [DescribeReservedDbInstancesOfferings](#) in *AWS CLI Command Reference*.

`describe-reserved-db-instances`

The following code example shows how to use `describe-reserved-db-instances`.

AWS CLI

To describe reserved DB instances

The following `describe-reserved-db-instances` example retrieves details about any reserved DB instances in the current AWS account.

```
aws rds describe-reserved-db-instances
```

Output:

```
{
  "ReservedDBInstances": [
    {
      "ReservedDBInstanceId": "myreservedinstance",
      "ReservedDBInstancesOfferingId": "12ab34cd-59af-4b2c-a660-1abcdef23456",
      "DBInstanceClass": "db.t3.micro",
      "StartTime": "2020-06-01T13:44:21.436Z",
      "Duration": 31536000,
      "FixedPrice": 0.0,
      "UsagePrice": 0.0,
      "CurrencyCode": "USD",
      "DBInstanceCount": 1,
      "ProductDescription": "sqlserver-ex(li)",
      "OfferingType": "No Upfront",
      "MultiAZ": false,
      "State": "payment-pending",
      "RecurringCharges": [
        {
          "RecurringChargeAmount": 0.014,
          "RecurringChargeFrequency": "Hourly"
        }
      ],
      "ReservedDBInstanceArn": "arn:aws:rds:us-west-2:123456789012:ri:myreservedinstance",
      "LeaseId": "a1b2c3d4-6b69-4a59-be89-5e11aa446666"
    }
  ]
}
```

For more information, see [Reserved DB Instances for Amazon RDS](#) in the *Amazon RDS User Guide*.

- For API details, see [DescribeReservedDbInstances](#) in *AWS CLI Command Reference*.

describe-source-regions

The following code example shows how to use `describe-source-regions`.

AWS CLI

To describe source Regions

The following `describe-source-regions` example retrieves details about all source AWS Regions. It also shows that automated backups can be replicated only from US West (Oregon) to the destination AWS Region, US East (N. Virginia).

```
aws rds describe-source-regions \  
  --region us-east-1
```

Output:

```
{  
  "SourceRegions": [  
    {  
      "RegionName": "af-south-1",  
      "Endpoint": "https://rds.af-south-1.amazonaws.com",  
      "Status": "available",  
      "SupportsDBInstanceAutomatedBackupsReplication": false  
    },  
    {  
      "RegionName": "ap-east-1",  
      "Endpoint": "https://rds.ap-east-1.amazonaws.com",  
      "Status": "available",  
      "SupportsDBInstanceAutomatedBackupsReplication": false  
    },  
    {  
      "RegionName": "ap-northeast-1",  
      "Endpoint": "https://rds.ap-northeast-1.amazonaws.com",  
      "Status": "available",  
      "SupportsDBInstanceAutomatedBackupsReplication": true  
    },  
    {  
      "RegionName": "ap-northeast-2",  
      "Endpoint": "https://rds.ap-northeast-2.amazonaws.com",
```

```
    "Status": "available",
    "SupportsDBInstanceAutomatedBackupsReplication": true
  },
  {
    "RegionName": "ap-northeast-3",
    "Endpoint": "https://rds.ap-northeast-3.amazonaws.com",
    "Status": "available",
    "SupportsDBInstanceAutomatedBackupsReplication": false
  },
  {
    "RegionName": "ap-south-1",
    "Endpoint": "https://rds.ap-south-1.amazonaws.com",
    "Status": "available",
    "SupportsDBInstanceAutomatedBackupsReplication": true
  },
  {
    "RegionName": "ap-southeast-1",
    "Endpoint": "https://rds.ap-southeast-1.amazonaws.com",
    "Status": "available",
    "SupportsDBInstanceAutomatedBackupsReplication": true
  },
  {
    "RegionName": "ap-southeast-2",
    "Endpoint": "https://rds.ap-southeast-2.amazonaws.com",
    "Status": "available",
    "SupportsDBInstanceAutomatedBackupsReplication": true
  },
  {
    "RegionName": "ap-southeast-3",
    "Endpoint": "https://rds.ap-southeast-3.amazonaws.com",
    "Status": "available",
    "SupportsDBInstanceAutomatedBackupsReplication": false
  },
  {
    "RegionName": "ca-central-1",
    "Endpoint": "https://rds.ca-central-1.amazonaws.com",
    "Status": "available",
    "SupportsDBInstanceAutomatedBackupsReplication": true
  },
  {
    "RegionName": "eu-north-1",
    "Endpoint": "https://rds.eu-north-1.amazonaws.com",
    "Status": "available",
    "SupportsDBInstanceAutomatedBackupsReplication": true
  }
```

```
  },
  {
    "RegionName": "eu-south-1",
    "Endpoint": "https://rds.eu-south-1.amazonaws.com",
    "Status": "available",
    "SupportsDBInstanceAutomatedBackupsReplication": false
  },
  {
    "RegionName": "eu-west-1",
    "Endpoint": "https://rds.eu-west-1.amazonaws.com",
    "Status": "available",
    "SupportsDBInstanceAutomatedBackupsReplication": true
  },
  {
    "RegionName": "eu-west-2",
    "Endpoint": "https://rds.eu-west-2.amazonaws.com",
    "Status": "available",
    "SupportsDBInstanceAutomatedBackupsReplication": true
  },
  {
    "RegionName": "eu-west-3",
    "Endpoint": "https://rds.eu-west-3.amazonaws.com",
    "Status": "available",
    "SupportsDBInstanceAutomatedBackupsReplication": true
  },
  {
    "RegionName": "me-central-1",
    "Endpoint": "https://rds.me-central-1.amazonaws.com",
    "Status": "available",
    "SupportsDBInstanceAutomatedBackupsReplication": false
  },
  {
    "RegionName": "me-south-1",
    "Endpoint": "https://rds.me-south-1.amazonaws.com",
    "Status": "available",
    "SupportsDBInstanceAutomatedBackupsReplication": false
  },
  {
    "RegionName": "sa-east-1",
    "Endpoint": "https://rds.sa-east-1.amazonaws.com",
    "Status": "available",
    "SupportsDBInstanceAutomatedBackupsReplication": true
  },
  {
```

```
    "RegionName": "us-east-2",
    "Endpoint": "https://rds.us-east-2.amazonaws.com",
    "Status": "available",
    "SupportsDBInstanceAutomatedBackupsReplication": true
  },
  {
    "RegionName": "us-west-1",
    "Endpoint": "https://rds.us-west-1.amazonaws.com",
    "Status": "available",
    "SupportsDBInstanceAutomatedBackupsReplication": true
  },
  {
    "RegionName": "us-west-2",
    "Endpoint": "https://rds.us-west-2.amazonaws.com",
    "Status": "available",
    "SupportsDBInstanceAutomatedBackupsReplication": true
  }
]
```

For more information, see [Finding information about replicated backups](#) in the *Amazon RDS User Guide*.

- For API details, see [DescribeSourceRegions](#) in *AWS CLI Command Reference*.

describe-valid-db-instance-modifications

The following code example shows how to use `describe-valid-db-instance-modifications`.

AWS CLI

To describe valid modifications for a DB instance

The following `describe-valid-db-instance-modifications` example retrieves details about the valid modifications for the specified DB instance.

```
aws rds describe-valid-db-instance-modifications \
  --db-instance-identifier test-instance
```

Output:

```
{
  "ValidDBInstanceModificationsMessage": {
    "ValidProcessorFeatures": [],
    "Storage": [
      {
        "StorageSize": [
          {
            "Step": 1,
            "To": 20,
            "From": 20
          },
          {
            "Step": 1,
            "To": 6144,
            "From": 22
          }
        ],
        "ProvisionedIops": [
          {
            "Step": 1,
            "To": 0,
            "From": 0
          }
        ],
        "IopsToStorageRatio": [
          {
            "To": 0.0,
            "From": 0.0
          }
        ],
        "StorageType": "gp2"
      },
      {
        "StorageSize": [
          {
            "Step": 1,
            "To": 6144,
            "From": 100
          }
        ],
        "ProvisionedIops": [
          {
            "Step": 1,
```

```
        "To": 40000,
        "From": 1000
      }
    ],
    "IopsToStorageRatio": [
      {
        "To": 50.0,
        "From": 1.0
      }
    ],
    "StorageType": "io1"
  },
  {
    "StorageSize": [
      {
        "Step": 1,
        "To": 20,
        "From": 20
      },
      {
        "Step": 1,
        "To": 3072,
        "From": 22
      }
    ],
    "ProvisionedIops": [
      {
        "Step": 1,
        "To": 0,
        "From": 0
      }
    ],
    "IopsToStorageRatio": [
      {
        "To": 0.0,
        "From": 0.0
      }
    ],
    "StorageType": "magnetic"
  }
]
}
```


- For API details, see [DescribeValidDbInstanceModifications](#) in *AWS CLI Command Reference*.

download-db-log-file-portion

The following code example shows how to use `download-db-log-file-portion`.

AWS CLI

To download a DB log file

The following `download-db-log-file-portion` example downloads only the latest part of your log file, saving it to a local file named `tail.txt`.

```
aws rds download-db-log-file-portion \  
  --db-instance-identifier test-instance \  
  --log-file-name log.txt \  
  --output text > tail.txt
```

To download the entire file, you need to include the `--starting-token 0` parameter. The following example saves the output to a local file named `full.txt`.

```
aws rds download-db-log-file-portion \  
  --db-instance-identifier test-instance \  
  --log-file-name log.txt \  
  --starting-token 0 \  
  --output text > full.txt
```

The saved file might contain blank lines. They appear at the end of each part of the log file while being downloaded. This generally doesn't cause any trouble in your log file analysis.

- For API details, see [DownloadDbLogFilePortion](#) in *AWS CLI Command Reference*.

generate-auth-token

The following code example shows how to use `generate-auth-token`.

AWS CLI

To generate an authentication token

The following `generate-db-auth-token` example generates an authentication token for use with IAM database authentication.

```
aws rds generate-db-auth-token \  
  --hostname aumysql-test.cdgmuiadpid.us-west-2.rds.amazonaws.com \  
  --port 3306 \  
  --region us-east-1 \  
  --username jane_doe
```

Output:

```
aumysql-test.cdgmuiadpid.us-west-2.rds.amazonaws.com:3306/?  
Action=connect&DBUser=jane_doe&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-  
Credential=AKIAIESZCNJ30EXAMPLE%2F20180731%2Fus-east-1%2Frds-db%2Faws4_request&X-  
Amz-Date=20180731T235209Z&X-Amz-Expires=900&X-Amz-SignedHeaders=host&X-Amz-  
Signature=5a8753ebEXAMPLEEa2c724e5667797EXAMPLE9d6ec6e3f427191fa41aeEXAMPLE
```

- For API details, see [GenerateAuthToken](#) in *AWS CLI Command Reference*.

generate-db-auth-token

The following code example shows how to use `generate-db-auth-token`.

AWS CLI

To generate an IAM authentication token

The following `generate-db-auth-token` example generates IAM authentication token to connect to a database.

```
aws rds generate-db-auth-token \  
  --hostname mydb.123456789012.us-east-1.rds.amazonaws.com \  
  --port 3306 \  
  --region us-east-1 \  
  --username db_user
```

Output:

```
mydb.123456789012.us-east-1.rds.amazonaws.com:3306/?  
Action=connect&DBUser=db_user&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-  
Credential=AKIAIEXAMPLE%2Fus-east-1%2Frds-db%2Faws4_request&X-Amz-  
Date=20210123T011543Z&X-Amz-Expires=900&X-Amz-SignedHeaders=host&X-Amz-  
Signature=88987EXAMPLE1EXAMPLE2EXAMPLE3EXAMPLE4EXAMPLE5EXAMPLE6
```

For more information, see [Connecting to your DB instance using IAM authentication](#) in the *Amazon RDS User Guide* and [Connecting to your DB cluster using IAM authentication](#) in the *Amazon Aurora User Guide*.

- For API details, see [GenerateDbAuthToken](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags on an Amazon RDS resource

The following `list-tags-for-resource` example lists all tags on a DB instance.

```
aws rds list-tags-for-resource \  
  --resource-name arn:aws:rds:us-east-1:123456789012:db:orc11
```

Output:

```
{  
  "TagList": [  
    {  
      "Key": "Environment",  
      "Value": "test"  
    },  
    {  
      "Key": "Name",  
      "Value": "MyDatabase"  
    }  
  ]  
}
```

For more information, see [Tagging Amazon RDS Resources](#) in the *Amazon RDS User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

modify-certificates

The following code example shows how to use `modify-certificates`.

AWS CLI

To temporarily override the system-default SSL/TLS certificate for new DB instances

The following `modify-certificates` example temporarily overrides the system-default SSL/TLS certificate for new DB instances.

```
aws rds modify-certificates \  
  --certificate-identifier rds-ca-2019
```

Output:

```
{  
  "Certificate": {  
    "CertificateIdentifier": "rds-ca-2019",  
    "CertificateType": "CA",  
    "Thumbprint": "EXAMPLE123456789012",  
    "ValidFrom": "2019-09-19T18:16:53Z",  
    "ValidTill": "2024-08-22T17:08:50Z",  
    "CertificateArn": "arn:aws:rds:us-east-1::cert:rds-ca-2019",  
    "CustomerOverride": true,  
    "CustomerOverrideValidTill": "2024-08-22T17:08:50Z"  
  }  
}
```

For more information, see [Rotating your SSL/TLS certificate](#) in the *Amazon RDS User Guide* and [Rotating your SSL/TLS certificate](#) in the *Amazon Aurora User Guide*.

- For API details, see [ModifyCertificates](#) in *AWS CLI Command Reference*.

`modify-current-db-cluster-capacity`

The following code example shows how to use `modify-current-db-cluster-capacity`.

AWS CLI

To scale the capacity of an Aurora Serverless DB cluster

The following `modify-current-db-cluster-capacity` example scales the capacity of an Aurora Serverless DB cluster to 8.

```
aws rds modify-current-db-cluster-capacity \  
  --db-cluster-identifier mydbcluster \  
  --target-capacity 8
```

```
--capacity 8
```

Output:

```
{
  "DBClusterIdentifier": "mydbcluster",
  "PendingCapacity": 8,
  "CurrentCapacity": 1,
  "SecondsBeforeTimeout": 300,
  "TimeoutAction": "ForceApplyCapacityChange"
}
```

For more information, see [Scaling Aurora Serverless v1 DB cluster capacity manually](#) in the *Amazon Aurora User Guide*.

- For API details, see [ModifyCurrentDbClusterCapacity](#) in *AWS CLI Command Reference*.

modify-db-cluster-endpoint

The following code example shows how to use `modify-db-cluster-endpoint`.

AWS CLI

To modify a custom DB cluster endpoint

The following `modify-db-cluster-endpoint` example modifies the specified custom DB cluster endpoint.

```
aws rds modify-db-cluster-endpoint \
  --db-cluster-endpoint-identifier mycustomendpoint \
  --static-members dbinstance1 dbinstance2 dbinstance3
```

Output:

```
{
  "DBClusterEndpointIdentifier": "mycustomendpoint",
  "DBClusterIdentifier": "mydbcluster",
  "DBClusterEndpointResourceIdentifier": "cluster-endpoint-ANPAJ4AE5446DAEXAMPLE",
  "Endpoint": "mycustomendpoint.cluster-custom-cnpxexample.us-east-1.rds.amazonaws.com",
  "Status": "modifying",
  "EndpointType": "CUSTOM",
}
```

```

    "CustomEndpointType": "READER",
    "StaticMembers": [
        "dbinstance1",
        "dbinstance2",
        "dbinstance3"
    ],
    "ExcludedMembers": [],
    "DBClusterEndpointArn": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:mycustomendpoint"
}

```

For more information, see [Amazon Aurora Connection Management](#) in the *Amazon Aurora User Guide*.

- For API details, see [ModifyDbClusterEndpoint](#) in *AWS CLI Command Reference*.

modify-db-cluster-parameter-group

The following code example shows how to use `modify-db-cluster-parameter-group`.

AWS CLI

To modify parameters in a DB cluster parameter group

The following `modify-db-cluster-parameter-group` example modifies the values of parameters in a DB cluster parameter group.

```

aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name mydbclusterpg \
  --parameters
  "ParameterName=server_audit_logging,ParameterValue=1,ApplyMethod=immediate" \
  "ParameterName=server_audit_logs_upload,ParameterValue=1,ApplyMethod=immediate"

```

Output:

```

{
  "DBClusterParameterGroupName": "mydbclusterpg"
}

```

For more information, see [Working with DB parameter groups and DB cluster parameter groups](#) in the *Amazon Aurora User Guide*.

- For API details, see [ModifyDbClusterParameterGroup](#) in *AWS CLI Command Reference*.

modify-db-cluster-snapshot-attribute

The following code example shows how to use `modify-db-cluster-snapshot-attribute`.

AWS CLI

To modify a DB cluster snapshot attribute

The following `modify-db-cluster-snapshot-attribute` example makes changes to the specified DB cluster snapshot attribute.

```
aws rds modify-db-cluster-snapshot-attribute \  
  --db-cluster-snapshot-identifier myclustersnapshot \  
  --attribute-name restore \  
  --values-to-add 123456789012
```

Output:

```
{  
  "DBClusterSnapshotAttributesResult": {  
    "DBClusterSnapshotIdentifier": "myclustersnapshot",  
    "DBClusterSnapshotAttributes": [  
      {  
        "AttributeName": "restore",  
        "AttributeValues": [  
          "123456789012"  
        ]  
      }  
    ]  
  }  
}
```

For more information, see [Restoring from a DB Cluster Snapshot](#) in the *Amazon Aurora User Guide*.

- For API details, see [ModifyDbClusterSnapshotAttribute](#) in *AWS CLI Command Reference*.

modify-db-cluster

The following code example shows how to use `modify-db-cluster`.

AWS CLI

Example 1: To modify a DB cluster

The following `modify-db-cluster` example changes the master user password for the DB cluster named `cluster-2` and sets the backup retention period to 14 days. The `--apply-immediately` parameter causes the changes to be made immediately, instead of waiting until the next maintenance window.

```
aws rds modify-db-cluster \  
  --db-cluster-identifier cluster-2 \  
  --backup-retention-period 14 \  
  --master-user-password newpassword99 \  
  --apply-immediately
```

Output:

```
{  
  "DBCluster": {  
    "AllocatedStorage": 1,  
    "AvailabilityZones": [  
      "eu-central-1b",  
      "eu-central-1c",  
      "eu-central-1a"  
    ],  
    "BackupRetentionPeriod": 14,  
    "DatabaseName": "",  
    "DBClusterIdentifier": "cluster-2",  
    "DBClusterParameterGroup": "default.aurora5.6",  
    "DBSubnetGroup": "default-vpc-2305ca49",  
    "Status": "available",  
    "EarliestRestorableTime": "2020-06-03T02:07:29.637Z",  
    "Endpoint": "cluster-2.cluster-#####.eu-central-1.rds.amazonaws.com",  
    "ReaderEndpoint": "cluster-2.cluster-ro-#####.eu-  
central-1.rds.amazonaws.com",  
    "MultiAZ": false,  
    "Engine": "aurora",  
    "EngineVersion": "5.6.10a",  
    "LatestRestorableTime": "2020-06-04T15:11:25.748Z",  
    "Port": 3306,  
    "MasterUsername": "admin",  
    "PreferredBackupWindow": "01:55-02:25",  
    "PreferredMaintenanceWindow": "thu:21:14-thu:21:44",
```



```

    "ReadReplicaIdentifiers": [],
    "DBClusterMembers": [
      {
        "DBInstanceIdentifier": "cluster-2-instance-1",
        "IsClusterWriter": true,
        "DBClusterParameterGroupStatus": "in-sync",
        "PromotionTier": 1
      }
    ],
    "VpcSecurityGroups": [
      {
        "VpcSecurityGroupId": "sg-20a5c047",
        "Status": "active"
      }
    ],
    "HostedZoneId": "Z1RLNU0EXAMPLE",
    "StorageEncrypted": true,
    "KmsKeyId": "arn:aws:kms:eu-central-1:123456789012:key/
d1bd7c8f-5cdb-49ca-8a62-a1b2c3d4e5f6",
    "DbClusterResourceId": "cluster-AGJ7XI77XVIS6FUXHU1EXAMPLE",
    "DBClusterArn": "arn:aws:rds:eu-central-1:123456789012:cluster:cluster-2",
    "AssociatedRoles": [],
    "IAMDatabaseAuthenticationEnabled": false,
    "ClusterCreateTime": "2020-04-03T14:44:02.764Z",
    "EngineMode": "provisioned",
    "DeletionProtection": false,
    "HttpEndpointEnabled": false,
    "CopyTagsToSnapshot": true,
    "CrossAccountClone": false,
    "DomainMemberships": []
  }
}

```

For more information, see [Modifying an Amazon Aurora DB Cluster](#) in the *Amazon Aurora User Guide*.

Example 2: To associate VPC security group with a DB cluster

The following `modify-db-instance` example associates a specific VPC security group and removes DB security groups from a DB cluster.

```

aws rds modify-db-cluster \
  --db-cluster-identifier dbName \

```

```
--vpc-security-group-ids sg-ID
```

Output:

```
{
  "DBCluster": {
    "AllocatedStorage": 1,
    "AvailabilityZones": [
      "us-west-2c",
      "us-west-2b",
      "us-west-2a"
    ],
    "BackupRetentionPeriod": 1,
    "DBClusterIdentifier": "dbName",
    "DBClusterParameterGroup": "default.aurora-mysql8.0",
    "DBSubnetGroup": "default",
    "Status": "available",
    "EarliestRestorableTime": "2024-02-15T01:12:13.966000+00:00",
    "Endpoint": "dbName.cluster-abcdefghji.us-west-2.rds.amazonaws.com",
    "ReaderEndpoint": "dbName.cluster-ro-abcdefghji.us-
west-2.rds.amazonaws.com",
    "MultiAZ": false,
    "Engine": "aurora-mysql",
    "EngineVersion": "8.0.mysql_aurora.3.04.1",
    "LatestRestorableTime": "2024-02-15T02:25:33.696000+00:00",
    "Port": 3306,
    "MasterUsername": "admin",
    "PreferredBackupWindow": "10:59-11:29",
    "PreferredMaintenanceWindow": "thu:08:54-thu:09:24",
    "ReadReplicaIdentifiers": [],
    "DBClusterMembers": [
      {
        "DBInstanceIdentifier": "dbName-instance-1",
        "IsClusterWriter": true,
        "DBClusterParameterGroupStatus": "in-sync",
        "PromotionTier": 1
      }
    ],
    "VpcSecurityGroups": [
      {
        "VpcSecurityGroupId": "sg-ID",
        "Status": "active"
      }
    ]
  }
}
```

```
    ],  
    ...output omitted...  
  }  
}
```

For more information, see [Controlling access with security groups](#) in the *Amazon Aurora User Guide*.

- For API details, see [ModifyDbCluster](#) in *AWS CLI Command Reference*.

modify-db-instance

The following code example shows how to use `modify-db-instance`.

AWS CLI

Example 1: To modify a DB instance

The following `modify-db-instance` example associates an option group and a parameter group with a compatible Microsoft SQL Server DB instance. The `--apply-immediately` parameter causes the option and parameter groups to be associated immediately, instead of waiting until the next maintenance window.

```
aws rds modify-db-instance \  
  --db-instance-identifier database-2 \  
  --option-group-name test-se-2017 \  
  --db-parameter-group-name test-sqlserver-se-2017 \  
  --apply-immediately
```

Output:

```
{  
  "DBInstance": {  
    "DBInstanceIdentifier": "database-2",  
    "DBInstanceClass": "db.r4.large",  
    "Engine": "sqlserver-se",  
    "DBInstanceStatus": "available",  
  
    ...output omitted...  
  
    "DBParameterGroups": [  

```

```

        {
            "DBParameterGroupName": "test-sqlserver-se-2017",
            "ParameterApplyStatus": "applying"
        }
    ],
    "AvailabilityZone": "us-west-2d",

    ...output omitted...

    "MultiAZ": true,
    "EngineVersion": "14.00.3281.6.v1",
    "AutoMinorVersionUpgrade": false,
    "ReadReplicaDBInstanceIdentifiers": [],
    "LicenseModel": "license-included",
    "OptionGroupMemberships": [
        {
            "OptionGroupName": "test-se-2017",
            "Status": "pending-apply"
        }
    ],
    "CharacterSetName": "SQL_Latin1_General_CP1_CI_AS",
    "SecondaryAvailabilityZone": "us-west-2c",
    "PubliclyAccessible": true,
    "StorageType": "gp2",

    ...output omitted...

    "DeletionProtection": false,
    "AssociatedRoles": [],
    "MaxAllocatedStorage": 1000
}
}

```

For more information, see [Modifying an Amazon RDS DB Instance](#) in the *Amazon RDS User Guide*.

Example 2: To associate VPC security group with a DB instance

The following `modify-db-instance` example associates a specific VPC security group and removes DB security groups from a DB instance:

```

aws rds modify-db-instance \
    --db-instance-identifier dbName \

```

```
--vpc-security-group-ids sg-ID
```

Output:

```
{
  "DBInstance": {
    "DBInstanceIdentifier": "dbName",
    "DBInstanceClass": "db.t3.micro",
    "Engine": "mysql",
    "DBInstanceStatus": "available",
    "MasterUsername": "admin",
    "Endpoint": {
      "Address": "dbName.abcdefghijkl.us-west-2.rds.amazonaws.com",
      "Port": 3306,
      "HostedZoneId": "ABCDEFGHIJK1234"
    },
    "AllocatedStorage": 20,
    "InstanceCreateTime": "2024-02-15T00:37:58.793000+00:00",
    "PreferredBackupWindow": "11:57-12:27",
    "BackupRetentionPeriod": 7,
    "DBSecurityGroups": [],
    "VpcSecurityGroups": [
      {
        "VpcSecurityGroupId": "sg-ID",
        "Status": "active"
      }
    ],
    ... output omitted ...
    "MultiAZ": false,
    "EngineVersion": "8.0.35",
    "AutoMinorVersionUpgrade": true,
    "ReadReplicaDBInstanceIdentifiers": [],
    "LicenseModel": "general-public-license",
    ... output omitted ...
  }
}
```

For more information, see [Controlling access with security groups](#) in the *Amazon RDS User Guide*.

- For API details, see [ModifyDBInstance](#) in *AWS CLI Command Reference*.

modify-db-parameter-group

The following code example shows how to use `modify-db-parameter-group`.

AWS CLI

To modify a DB parameter group

The following `modify-db-parameter-group` example changes the value of the `clr` enabled parameter in a DB parameter group. The `--apply-immediately` parameter causes the DB parameter group to be modified immediately, instead of waiting until the next maintenance window.

```
aws rds modify-db-parameter-group \  
  --db-parameter-group-name test-sqlserver-se-2017 \  
  --parameters "ParameterName='clr  
enabled',ParameterValue=1,ApplyMethod=immediate"
```

Output:

```
{  
  "DBParameterGroupName": "test-sqlserver-se-2017"  
}
```

For more information, see [Modifying Parameters in a DB Parameter Group](#) in the *Amazon RDS User Guide*.

- For API details, see [ModifyDBParameterGroup](#) in *AWS CLI Command Reference*.

modify-db-proxy-endpoint

The following code example shows how to use `modify-db-proxy-endpoint`.

AWS CLI

To modify a DB proxy endpoint for an RDS database

The following `modify-db-proxy-endpoint` example modifies a DB proxy endpoint `proxyEndpoint` to set the read-timeout to 65 seconds.

```
aws rds modify-db-proxy-endpoint \  
  --db-proxy-endpoint-name proxyEndpoint \  
  --read-timeout-seconds 65
```

```
--cli-read-timeout 65
```

Output:

```
{
  "DBProxyEndpoint":
    {
      "DBProxyEndpointName": "proxyEndpoint",
      "DBProxyEndpointArn": "arn:aws:rds:us-east-1:123456789012:db-proxy-
endpoint:prx-endpoint-0123a01b12345c0ab",
      "DBProxyName": "proxyExample",
      "Status": "available",
      "VpcId": "vpc-1234567",
      "VpcSecurityGroupIds": [
        "sg-1234"
      ],
      "VpcSubnetIds": [
        "subnetgroup1",
        "subnetgroup2"
      ],
      "Endpoint": "proxyEndpoint.endpoint.proxyExample-ab0cd1efghij.us-
east-1.rds.amazonaws.com",
      "CreateDate": "2023-04-05T16:09:33.452000+00:00",
      "TargetRole": "READ_WRITE",
      "IsDefault": "false"
    }
}
```

For more information, see [Modifying a proxy endpoint](#) in the *Amazon RDS User Guide* and [Modifying a proxy endpoint](#) in the *Amazon Aurora User Guide*.

- For API details, see [ModifyDbProxyEndpoint](#) in *AWS CLI Command Reference*.

modify-db-proxy-target-group

The following code example shows how to use `modify-db-proxy-target-group`.

AWS CLI

To modify a DB proxy endpoints

The following `modify-db-proxy-target-group` example modifies a DB proxy target group to set the maximum connections to 80 percent and maximum idle connections to 10 percent.

```
aws rds modify-db-proxy-target-group \  
  --target-group-name default \  
  --db-proxy-name proxyExample \  
  --connection-pool-config MaxConnectionsPercent=80,MaxIdleConnectionsPercent=10
```

Output:

```
{  
  "DBProxyTargetGroup":  
    {  
      "DBProxyName": "proxyExample",  
      "TargetGroupName": "default",  
      "TargetGroupArn": "arn:aws:rds:us-east-1:123456789012:target-group:prx-  
tg-0123a01b12345c0ab",  
      "IsDefault": true,  
      "Status": "available",  
      "ConnectionPoolConfig": {  
        "MaxConnectionsPercent": 80,  
        "MaxIdleConnectionsPercent": 10,  
        "ConnectionBorrowTimeout": 120,  
        "SessionPinningFilters": []  
      },  
      "CreateDate": "2023-05-02T18:41:19.495000+00:00",  
      "UpdateDate": "2023-05-02T18:41:21.762000+00:00"  
    }  
}
```

For more information, see [Modifying an RDS Proxy](#) in the *Amazon RDS User Guide* and [Modifying an RDS Proxy](#) in the *Amazon Aurora User Guide*.

- For API details, see [ModifyDbProxyTargetGroup](#) in *AWS CLI Command Reference*.

modify-db-proxy

The following code example shows how to use `modify-db-proxy`.

AWS CLI

To modify a DB proxy for an RDS database

The following `modify-db-proxy` example modifies a DB proxy named `proxyExample` to require SSL for its connections.


```
aws rds modify-db-proxy \  
  --db-proxy-name proxyExample \  
  --require-tls
```

Output:

```
{  
  "DBProxy":  
    {  
      "DBProxyName": "proxyExample",  
      "DBProxyArn": "arn:aws:rds:us-east-1:123456789012:db-  
proxy:prx-0123a01b12345c0ab",  
      "Status": "modifying"  
      "EngineFamily": "PostgreSQL",  
      "VpcId": "sg-1234567",  
      "VpcSecurityGroupIds": [  
        "sg-1234"  
      ],  
      "VpcSubnetIds": [  
        "subnetgroup1",  
        "subnetgroup2"  
      ],  
      "Auth": "[  
        {  
          "Description": "proxydescription1",  
          "AuthScheme": "SECRETS",  
          "SecretArn": "arn:aws:secretsmanager:us-  
west-2:123456789123:secret:proxysecret1-Abcd1e",  
          "IAMAuth": "DISABLED"  
        }  
      ]",  
      "RoleArn": "arn:aws:iam::12345678912:role/ProxyPostgreSQLRole",  
      "Endpoint": "proxyExample.proxy-ab0cd1efghij.us-east-1.rds.amazonaws.com",  
      "RequireTLS": true,  
      "IdleClientTimeout": 1800,  
      "DebuggingLogging": false,  
      "CreateDate": "2023-04-05T16:09:33.452000+00:00",  
      "UpdatedDate": "2023-04-13T01:49:38.568000+00:00"  
    }  
}
```

For more information, see [Modify an RDS Proxy](#) in the *Amazon RDS User Guide* and [Creating an RDS Proxy](#) in the *Amazon Aurora User Guide*.

- For API details, see [ModifyDbProxy](#) in *AWS CLI Command Reference*.

modify-db-shard-group

The following code example shows how to use `modify-db-shard-group`.

AWS CLI

Example 1: To modify a DB shard group

The following `modify-db-shard-group` example changes the maximum capacity of a DB shard group.

```
aws rds modify-db-shard-group \  
  --db-shard-group-identifier my-db-shard-group \  
  --max-acu 1000
```

Output:

```
{  
  "DBShardGroups": [  
    {  
      "DBShardGroupResourceId": "shardgroup-a6e3a0226aa243e2ac6c7a1234567890",  
      "DBShardGroupIdentifier": "my-db-shard-group",  
      "DBClusterIdentifier": "my-sv2-cluster",  
      "MaxACU": 768.0,  
      "ComputeRedundancy": 0,  
      "Status": "available",  
      "PubliclyAccessible": false,  
      "Endpoint": "my-sv2-cluster.limitless-cekyceexample.us-  
east-2.rds.amazonaws.com"  
    }  
  ]  
}
```

For more information, see [Amazon Aurora DB Clusters](#) in the *Amazon Aurora User Guide*.

Example 2: To describe your DB shard groups

The following `describe-db-shard-groups` example retrieves the details of your DB shard groups after you run the `modify-db-shard-group` command. The maximum capacity of the DB shard group `my-db-shard-group` is now 1000 Aurora capacity units (ACUs).

```
aws rds describe-db-shard-groups
```

Output:

```
{
  "DBShardGroups": [
    {
      "DBShardGroupResourceId": "shardgroup-7bb446329da94788b3f957746example",
      "DBShardGroupIdentifier": "limitless-test-shard-grp",
      "DBClusterIdentifier": "limitless-test-cluster",
      "MaxACU": 768.0,
      "ComputeRedundancy": 0,
      "Status": "available",
      "PubliclyAccessible": true,
      "Endpoint": "limitless-test-cluster.limitless-cekycexample.us-east-2.rds.amazonaws.com"
    },
    {
      "DBShardGroupResourceId": "shardgroup-a6e3a0226aa243e2ac6c7a1234567890",
      "DBShardGroupIdentifier": "my-db-shard-group",
      "DBClusterIdentifier": "my-sv2-cluster",
      "MaxACU": 1000.0,
      "ComputeRedundancy": 0,
      "Status": "available",
      "PubliclyAccessible": false,
      "Endpoint": "my-sv2-cluster.limitless-cekycexample.us-east-2.rds.amazonaws.com"
    }
  ]
}
```

For more information, see [Amazon Aurora DB Clusters](#) in the *Amazon Aurora User Guide*.

- For API details, see [ModifyDbShardGroup](#) in *AWS CLI Command Reference*.

modify-db-snapshot-attribute

The following code example shows how to use `modify-db-snapshot-attribute`.

AWS CLI

Example 1: To enable two AWS accounts to restore a DB snapshot

The following `modify-db-snapshot-attribute` example grants permission to two AWS accounts, with the identifiers 111122223333 and 444455556666, to restore the DB snapshot named `mydbsnapshot`.

```
aws rds modify-db-snapshot-attribute \  
  --db-snapshot-identifier mydbsnapshot \  
  --attribute-name restore \  
  --values-to-add {"111122223333","444455556666"}
```

Output:

```
{  
  "DBSnapshotAttributesResult": {  
    "DBSnapshotIdentifier": "mydbsnapshot",  
    "DBSnapshotAttributes": [  
      {  
        "AttributeName": "restore",  
        "AttributeValues": [  
          "111122223333",  
          "444455556666"  
        ]  
      }  
    ]  
  }  
}
```

For more information, see [Sharing a Snapshot](#) in the *Amazon RDS User Guide*.

Example 2: To prevent an AWS account from restoring a DB snapshot

The following `modify-db-snapshot-attribute` example removes permission from a particular AWS account to restore the DB snapshot named `mydbsnapshot`. When specifying a single account, the account identifier can't be surrounded by quotations marks or braces.

```
aws rds modify-db-snapshot-attribute \  
  --db-snapshot-identifier mydbsnapshot \  
  --attribute-name restore \  
  --values-to-remove 444455556666
```

Output:

```
{
  "DBSnapshotAttributesResult": {
    "DBSnapshotIdentifier": "mydbsnapshot",
    "DBSnapshotAttributes": [
      {
        "AttributeName": "restore",
        "AttributeValues": [
          "111122223333"
        ]
      }
    ]
  }
}
```

For more information, see [Sharing a Snapshot](#) in the *Amazon RDS User Guide*.

- For API details, see [ModifyDbSnapshotAttribute](#) in *AWS CLI Command Reference*.

modify-db-snapshot-attributes

The following code example shows how to use `modify-db-snapshot-attributes`.

AWS CLI**To modify a DB snapshot attribute**

The following `modify-db-snapshot-attribute` example permits two AWS account identifiers, 111122223333 and 444455556666, to restore the DB snapshot named `mydbsnapshot`.

```
aws rds modify-db-snapshot-attribute \
  --db-snapshot-identifier mydbsnapshot \
  --attribute-name restore \
  --values-to-add ["111122223333","444455556666"]'
```

Output:

```
{
  "DBSnapshotAttributesResult": {
    "DBSnapshotIdentifier": "mydbsnapshot",
```

```
    "DBSnapshotAttributes": [
      {
        "AttributeName": "restore",
        "AttributeValues": [
          "111122223333",
          "444455556666"
        ]
      }
    ]
  }
}
```

For more information, see [Sharing a Snapshot](#) in the *Amazon RDS User Guide*.

- For API details, see [ModifyDbSnapshotAttributes](#) in *AWS CLI Command Reference*.

modify-db-snapshot

The following code example shows how to use `modify-db-snapshot`.

AWS CLI

To modify a DB snapshot

The following `modify-db-snapshot` example upgrades a PostgreSQL 10.6 snapshot named `db5-snapshot-upg-test` to PostgreSQL 11.7. The new DB engine version is shown after the snapshot has finished upgrading and its status is **available**.

```
aws rds modify-db-snapshot \  
  --db-snapshot-identifier db5-snapshot-upg-test \  
  --engine-version 11.7
```

Output:

```
{  
  "DBSnapshot": {  
    "DBSnapshotIdentifier": "db5-snapshot-upg-test",  
    "DBInstanceIdentifier": "database-5",  
    "SnapshotCreateTime": "2020-03-27T20:49:17.092Z",  
    "Engine": "postgres",  
    "AllocatedStorage": 20,  
    "Status": "upgrading",  
    "Port": 5432,
```

```

    "AvailabilityZone": "us-west-2a",
    "VpcId": "vpc-2ff27557",
    "InstanceCreateTime": "2020-03-27T19:59:04.735Z",
    "MasterUsername": "postgres",
    "EngineVersion": "10.6",
    "LicenseModel": "postgresql-license",
    "SnapshotType": "manual",
    "OptionGroupName": "default:postgres-11",
    "PercentProgress": 100,
    "StorageType": "gp2",
    "Encrypted": false,
    "DBSnapshotArn": "arn:aws:rds:us-west-2:123456789012:snapshot:db5-snapshot-
upg-test",
    "IAMDatabaseAuthenticationEnabled": false,
    "ProcessorFeatures": [],
    "DbiResourceId": "db-GJMF75LM42IL6BTFRE4UZJ5YM4"
  }
}

```

For more information, see [Upgrading a PostgreSQL DB Snapshot](#) in the *Amazon RDS User Guide*.

- For API details, see [ModifyDbSnapshot](#) in *AWS CLI Command Reference*.

modify-db-subnet-group

The following code example shows how to use `modify-db-subnet-group`.

AWS CLI

To modify a DB subnet group

The following `modify-db-subnet-group` example adds a subnet with the ID `subnet-08e41f9e230222222` to the DB subnet group named `mysubnetgroup`. To keep the existing subnets in the subnet group, include their IDs as values in the `--subnet-ids` option. Make sure to have subnets with at least two different Availability Zones in the DB subnet group.

```

aws rds modify-db-subnet-group \
  --db-subnet-group-name mysubnetgroup \
  --subnet-ids
  ["subnet-0a1dc4e1a6f123456", "subnet-070dd7ecb3aaaaaaaa", "subnet-00f5b198bc0abcdef", "subnet-

```

Output:

```
{
  "DBSubnetGroup": {
    "DBSubnetGroupName": "mysubnetgroup",
    "DBSubnetGroupDescription": "test DB subnet group",
    "VpcId": "vpc-0f08e7610a1b2c3d4",
    "SubnetGroupStatus": "Complete",
    "Subnets": [
      {
        "SubnetIdentifier": "subnet-08e41f9e230222222",
        "SubnetAvailabilityZone": {
          "Name": "us-west-2a"
        },
        "SubnetStatus": "Active"
      },
      {
        "SubnetIdentifier": "subnet-070dd7ecb3aaaaaaaa",
        "SubnetAvailabilityZone": {
          "Name": "us-west-2b"
        },
        "SubnetStatus": "Active"
      },
      {
        "SubnetIdentifier": "subnet-00f5b198bc0abcdef",
        "SubnetAvailabilityZone": {
          "Name": "us-west-2d"
        },
        "SubnetStatus": "Active"
      },
      {
        "SubnetIdentifier": "subnet-0a1dc4e1a6f123456",
        "SubnetAvailabilityZone": {
          "Name": "us-west-2b"
        },
        "SubnetStatus": "Active"
      }
    ],
    "DBSubnetGroupArn": "arn:aws:rds:us-west-2:534026745191:subgrp:mysubnetgroup"
  }
}
```

For more information, see [Step 3: Create a DB Subnet Group](#) in the *Amazon RDS User Guide*.

- For API details, see [ModifyDbSubnetGroup](#) in *AWS CLI Command Reference*.

modify-event-subscription

The following code example shows how to use `modify-event-subscription`.

AWS CLI

To modify an event subscription

The following `modify-event-subscription` example disables the specified event subscription, so that it no longer publishes notifications to the specified Amazon Simple Notification Service topic.

```
aws rds modify-event-subscription \  
  --subscription-name my-instance-events \  
  --no-enabled
```

Output:

```
{  
  "EventSubscription": {  
    "EventCategoriesList": [  
      "backup",  
      "recovery"  
    ],  
    "CustomerAwsId": "123456789012",  
    "SourceType": "db-instance",  
    "SubscriptionCreationTime": "Tue Jul 31 23:22:01 UTC 2018",  
    "EventSubscriptionArn": "arn:aws:rds:us-east-1:123456789012:es:my-instance-  
events",  
    "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:interesting-events",  
    "CustSubscriptionId": "my-instance-events",  
    "Status": "modifying",  
    "Enabled": false  
  }  
}
```

- For API details, see [ModifyEventSubscription](#) in *AWS CLI Command Reference*.

modify-global-cluster

The following code example shows how to use `modify-global-cluster`.

AWS CLI

To modify a global DB cluster

The following `modify-global-cluster` example enables deletion protection for an Aurora MySQL-compatible global DB cluster.

```
aws rds modify-global-cluster \  
  --global-cluster-identifier myglobalcluster \  
  --deletion-protection
```

Output:

```
{  
  "GlobalCluster": {  
    "GlobalClusterIdentifier": "myglobalcluster",  
    "GlobalClusterResourceId": "cluster-f0e523bfe07aabb",  
    "GlobalClusterArn": "arn:aws:rds::123456789012:global-  
cluster:myglobalcluster",  
    "Status": "available",  
    "Engine": "aurora-mysql",  
    "EngineVersion": "5.7.mysql_aurora.2.07.2",  
    "StorageEncrypted": false,  
    "DeletionProtection": true,  
    "GlobalClusterMembers": []  
  }  
}
```

For more information, see [Managing an Aurora global database](#) in the *Amazon Aurora User Guide*.

- For API details, see [ModifyGlobalCluster](#) in *AWS CLI Command Reference*.

promote-read-replica-db-cluster

The following code example shows how to use `promote-read-replica-db-cluster`.

AWS CLI

To promote a DB cluster read replica

The following `promote-read-replica-db-cluster` example promotes the specified read replica to become a standalone DB cluster.

```
aws rds promote-read-replica-db-cluster \  
  --db-cluster-identifier mydbcluster-1
```

Output:

```
{  
  "DBCluster": {  
    "AllocatedStorage": 1,  
    "AvailabilityZones": [  
      "us-east-1a",  
      "us-east-1b",  
      "us-east-1c"  
    ],  
    "BackupRetentionPeriod": 1,  
    "DatabaseName": "",  
    "DBClusterIdentifier": "mydbcluster-1",  
    "...some output truncated..."  
  }  
}
```

For more information, see [Promoting a read replica to be a DB cluster](#) in the *Amazon Aurora User Guide*.

- For API details, see [PromoteReadReplicaDbCluster](#) in *AWS CLI Command Reference*.

promote-read-replica

The following code example shows how to use `promote-read-replica`.

AWS CLI

To promote a read replica

The following `promote-read-replica` example promotes the specified read replica to become a standalone DB instance.

```
aws rds promote-read-replica \  
  --db-instance-identifier test-instance-repl
```

Output:

```
{  
  "DBInstance": {  
    "DBInstanceArn": "arn:aws:rds:us-east-1:123456789012:db:test-instance-repl",  
    "StorageType": "standard",  
    "ReadReplicaSourceDBInstanceIdentifier": "test-instance",  
    "DBInstanceStatus": "modifying",  
    ...some output truncated...  
  }  
}
```

- For API details, see [PromoteReadReplica](#) in *AWS CLI Command Reference*.

purchase-reserved-db-instance

The following code example shows how to use `purchase-reserved-db-instance`.

AWS CLI

To purchase a reserved DB instance offering

The following `purchase-reserved-db-instances-offering` example purchases a reserved DB instance offering. The `reserved-db-instances-offering-id` must be a valid offering ID, as returned by the `describe-reserved-db-instances-offering` command.

```
aws rds purchase-reserved-db-instances-offering --reserved-db-instances-offering-id  
438012d3-4a52-4cc7-b2e3-8dff72e0e706
```

- For API details, see [PurchaseReservedDbInstance](#) in *AWS CLI Command Reference*.

purchase-reserved-db-instances-offerings

The following code example shows how to use `purchase-reserved-db-instances-offerings`.

AWS CLI

Example 1: To find a reserved DB instance to purchase

The following `describe-reserved-db-instances-offerings` example lists the available reserved MySQL DB instances with the `db.t2.micro` instance class and a duration of one year. The offering ID is required for purchasing a reserved DB instance.

```
aws rds describe-reserved-db-instances-offerings \  
  --product-description mysql \  
  --db-instance-class db.t2.micro \  
  --duration 1
```

Output:

```
{  
  "ReservedDBInstancesOfferings": [  
    {  
      "ReservedDBInstancesOfferingId": "8ba30be1-b9ec-447f-8f23-6114e3f4c7b4",  
      "DBInstanceClass": "db.t2.micro",  
      "Duration": 31536000,  
      "FixedPrice": 51.0,  
      "UsagePrice": 0.0,  
      "CurrencyCode": "USD",  
      "ProductDescription": "mysql",  
      "OfferingType": "Partial Upfront",  
      "MultiAZ": false,  
      "RecurringCharges": [  
        {  
          "RecurringChargeAmount": 0.006,  
          "RecurringChargeFrequency": "Hourly"  
        }  
      ]  
    },  
    ... some output truncated ...  
  ]  
}
```

For more information, see [Reserved DB Instances for Amazon RDS](#) in the *Amazon RDS User Guide*.

Example 2: To purchase a reserved DB instance

The following `purchase-reserved-db-instances-offering` example shows how to buy the reserved DB instance offering from the previous example.

```
aws rds purchase-reserved-db-instances-offering --reserved-db-instances-offering-id
8ba30be1-b9ec-447f-8f23-6114e3f4c7b4
```

Output:

```
{
  "ReservedDBInstance": {
    "ReservedDBInstanceId": "ri-2020-06-29-16-54-57-670",
    "ReservedDBInstancesOfferingId": "8ba30be1-b9ec-447f-8f23-6114e3f4c7b4",
    "DBInstanceClass": "db.t2.micro",
    "StartTime": "2020-06-29T16:54:57.670Z",
    "Duration": 31536000,
    "FixedPrice": 51.0,
    "UsagePrice": 0.0,
    "CurrencyCode": "USD",
    "DBInstanceCount": 1,
    "ProductDescription": "mysql",
    "OfferingType": "Partial Upfront",
    "MultiAZ": false,
    "State": "payment-pending",
    "RecurringCharges": [
      {
        "RecurringChargeAmount": 0.006,
        "RecurringChargeFrequency": "Hourly"
      }
    ],
    "ReservedDBInstanceArn": "arn:aws:rds:us-
west-2:123456789012:ri:ri-2020-06-29-16-54-57-670"
  }
}
```

For more information, see [Reserved DB Instances for Amazon RDS](#) in the *Amazon RDS User Guide*.

- For API details, see [PurchaseReservedDbInstancesOfferings](#) in *AWS CLI Command Reference*.

reboot-db-instance

The following code example shows how to use `reboot-db-instance`.

AWS CLI

To reboot a DB instance

The following `reboot-db-instance` example starts a reboot of the specified DB instance.

```
aws rds reboot-db-instance \  
  --db-instance-identifier test-mysql-instance
```

Output:

```
{  
  "DBInstance": {  
    "DBInstanceIdentifier": "test-mysql-instance",  
    "DBInstanceClass": "db.t3.micro",  
    "Engine": "mysql",  
    "DBInstanceStatus": "rebooting",  
    "MasterUsername": "admin",  
    "Endpoint": {  
      "Address": "test-mysql-instance.#####.us-  
west-2.rds.amazonaws.com",  
      "Port": 3306,  
      "HostedZoneId": "Z1PVIF0EXAMPLE"  
    },  
    ... output omitted...  
  }  
}
```

For more information, see [Rebooting a DB Instance](#) in the *Amazon RDS User Guide*.

- For API details, see [RebootDBInstance](#) in *AWS CLI Command Reference*.

reboot-db-shard-group

The following code example shows how to use `reboot-db-shard-group`.

AWS CLI

Example 1: To reboot a DB shard group

The following `reboot-db-shard-group` example reboots a DB shard group.

```
aws rds reboot-db-shard-group \  
  --db-shard-group-identifier test-mysql-instance
```

```
--db-shard-group-identifier my-db-shard-group
```

Output:

```
{
  "DBShardGroups": [
    {
      "DBShardGroupResourceId": "shardgroup-a6e3a0226aa243e2ac6c7a1234567890",
      "DBShardGroupIdentifier": "my-db-shard-group",
      "DBClusterIdentifier": "my-sv2-cluster",
      "MaxACU": 1000.0,
      "ComputeRedundancy": 0,
      "Status": "available",
      "PubliclyAccessible": false,
      "Endpoint": "my-sv2-cluster.limitless-cekyceexample.us-east-2.rds.amazonaws.com"
    }
  ]
}
```

For more information, see [Rebooting an Amazon Aurora DB cluster or Amazon Aurora DB instance](#) in the *Amazon Aurora User Guide*.

Example 2: To describe your DB shard groups

The following `describe-db-shard-groups` example retrieves the details of your DB shard groups after you run the `reboot-db-shard-group` command. The DB shard group `my-db-shard-group` is now rebooting.

```
aws rds describe-db-shard-groups
```

Output:

```
{
  "DBShardGroups": [
    {
      "DBShardGroupResourceId": "shardgroup-7bb446329da94788b3f957746example",
      "DBShardGroupIdentifier": "limitless-test-shard-grp",
      "DBClusterIdentifier": "limitless-test-cluster",
      "MaxACU": 768.0,
      "ComputeRedundancy": 0,

```



```

        "Status": "available",
        "PubliclyAccessible": true,
        "Endpoint": "limitless-test-cluster.limitless-cekyceexample.us-
east-2.rds.amazonaws.com"
    },
    {
        "DBShardGroupResourceId": "shardgroup-a6e3a0226aa243e2ac6c7a1234567890",
        "DBShardGroupIdentifier": "my-db-shard-group",
        "DBClusterIdentifier": "my-sv2-cluster",
        "MaxACU": 1000.0,
        "ComputeRedundancy": 0,
        "Status": "rebooting",
        "PubliclyAccessible": false,
        "Endpoint": "my-sv2-cluster.limitless-cekyceexample.us-
east-2.rds.amazonaws.com"
    }
]
}

```

For more information, see [Rebooting an Amazon Aurora DB cluster or Amazon Aurora DB instance](#) in the *Amazon Aurora User Guide*.

- For API details, see [RebootDbShardGroup](#) in *AWS CLI Command Reference*.

register-db-proxy-targets

The following code example shows how to use `register-db-proxy-targets`.

AWS CLI

To register a DB proxy with a database

The following `register-db-proxy-targets` example creates the association between a database and a proxy.

```

aws rds register-db-proxy-targets \
  --db-proxy-name proxyExample \
  --db-cluster-identifiers database-5

```

Output:

```
{
```

```
"DBProxyTargets": [  
  {  
    "RdsResourceId": "database-5",  
    "Port": 3306,  
    "Type": "TRACKED_CLUSTER",  
    "TargetHealth": {  
      "State": "REGISTERING"  
    }  
  },  
  {  
    "Endpoint": "database-5instance-1.ab0cd1efghij.us-  
east-1.rds.amazonaws.com",  
    "RdsResourceId": "database-5",  
    "Port": 3306,  
    "Type": "RDS_INSTANCE",  
    "TargetHealth": {  
      "State": "REGISTERING"  
    }  
  }  
]  
}
```

For more information, see [Creating an RDS proxy](#) in the *Amazon RDS User Guide* and [Creating an RDS proxy](#) in the *Amazon Aurora User Guide*.

- For API details, see [RegisterDbProxyTargets](#) in *AWS CLI Command Reference*.

remove-from-global-cluster

The following code example shows how to use `remove-from-global-cluster`.

AWS CLI

To detach an Aurora secondary cluster from an Aurora global database cluster

The following `remove-from-global-cluster` example detaches an Aurora secondary cluster from an Aurora global database cluster. The cluster changes from being read-only to a standalone cluster with read-write capability.

```
aws rds remove-from-global-cluster \  
  --region us-west-2 \  
  --global-cluster-identifier myglobalcluster \  
  --secondary-cluster-identifier mysecondarycluster
```

```
--db-cluster-identifier arn:aws:rds:us-west-2:123456789012:cluster:DB-1
```

Output:

```
{
  "GlobalCluster": {
    "GlobalClusterIdentifier": "myglobalcluster",
    "GlobalClusterResourceId": "cluster-abc123def456gh",
    "GlobalClusterArn": "arn:aws:rds::123456789012:global-
cluster:myglobalcluster",
    "Status": "available",
    "Engine": "aurora-postgresql",
    "EngineVersion": "10.11",
    "StorageEncrypted": true,
    "DeletionProtection": false,
    "GlobalClusterMembers": [
      {
        "DBClusterArn": "arn:aws:rds:us-east-1:123456789012:cluster:js-
global-cluster",
        "Readers": [
          "arn:aws:rds:us-west-2:123456789012:cluster:DB-1"
        ],
        "IsWriter": true
      },
      {
        "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:DB-1",
        "Readers": [],
        "IsWriter": false,
        "GlobalWriteForwardingStatus": "disabled"
      }
    ]
  }
}
```

For more information, see [Removing a cluster from an Amazon Aurora global database](#) in the *Amazon Aurora User Guide*.

- For API details, see [RemoveFromGlobalCluster](#) in *AWS CLI Command Reference*.

remove-option-from-option-group

The following code example shows how to use `remove-option-from-option-group`.

AWS CLI

To delete an option from an option group

The following `remove-option-from-option-group` example removes the OEM option from `myoptiongroup`.

```
aws rds remove-option-from-option-group \  
  --option-group-name myoptiongroup \  
  --options OEM \  
  --apply-immediately
```

Output:

```
{  
  "OptionGroup": {  
    "OptionGroupName": "myoptiongroup",  
    "OptionGroupDescription": "Test",  
    "EngineName": "oracle-ee",  
    "MajorEngineVersion": "19",  
    "Options": [],  
    "AllowsVpcAndNonVpcInstanceMemberships": true,  
    "OptionGroupArn": "arn:aws:rds:us-east-1:123456789012:og:myoptiongroup"  
  }  
}
```

For more information, see [Removing an Option from an Option Group](#) in the *Amazon Aurora User Guide*.

- For API details, see [RemoveOptionFromOptionGroup](#) in *AWS CLI Command Reference*.

remove-role-from-db-cluster

The following code example shows how to use `remove-role-from-db-cluster`.

AWS CLI

To disassociate an AWS Identity and Access Management (IAM) role from a DB cluster

The following `remove-role-from-db-cluster` example removes a role from a DB cluster.

```
aws rds remove-role-from-db-cluster \  
  \
```

```
--db-cluster-identifier mydbcluster \  
--role-arn arn:aws:iam::123456789012:role/RDSLoadFromS3
```

This command produces no output.

For more information, see [Associating an IAM role with an Amazon Aurora MySQL DB cluster](#) in the *Amazon Aurora User Guide*.

- For API details, see [RemoveRoleFromDbCluster](#) in *AWS CLI Command Reference*.

remove-role-from-db-instance

The following code example shows how to use `remove-role-from-db-instance`.

AWS CLI

To disassociate an AWS Identity and Access Management (IAM) role from a DB instance

The following `remove-role-from-db-instance` example removes the role named `rds-s3-integration-role` from an Oracle DB instance named `test-instance`.

```
aws rds remove-role-from-db-instance \  
  --db-instance-identifier test-instance \  
  --feature-name S3_INTEGRATION \  
  --role-arn arn:aws:iam::111122223333:role/rds-s3-integration-role
```

This command produces no output.

For more information, see [Disabling RDS SQL Server Integration with S3](#) in the *Amazon RDS User Guide*.

- For API details, see [RemoveRoleFromDbInstance](#) in *AWS CLI Command Reference*.

remove-source-identifier-from-subscription

The following code example shows how to use `remove-source-identifier-from-subscription`.

AWS CLI

To remove a source identifier from a subscription

The following `remove-source-identifier` example removes the specified source identifier from an existing subscription.

```
aws rds remove-source-identifier-from-subscription \  
  --subscription-name my-instance-events \  
  --source-identifier test-instance-repl
```

Output:

```
{  
  "EventSubscription": {  
    "EventSubscriptionArn": "arn:aws:rds:us-east-1:123456789012:es:my-instance-  
events",  
    "SubscriptionCreationTime": "Tue Jul 31 23:22:01 UTC 2018",  
    "EventCategoriesList": [  
      "backup",  
      "recovery"  
    ],  
    "SnsTopicArn": "arn:aws:sns:us-east-1:123456789012:interesting-events",  
    "Status": "modifying",  
    "CustSubscriptionId": "my-instance-events",  
    "CustomerAwsId": "123456789012",  
    "SourceIdsList": [  
      "test-instance"  
    ],  
    "SourceType": "db-instance",  
    "Enabled": false  
  }  
}
```

- For API details, see [RemoveSourceIdentifierFromSubscription](#) in *AWS CLI Command Reference*.

remove-tags-from-resource

The following code example shows how to use `remove-tags-from-resource`.

AWS CLI

To remove tags from a resource

The following `remove-tags-from-resource` example removes tags from a resource.

```
aws rds remove-tags-from-resource \  
  --resource-name arn:aws:rds:us-east-1:123456789012:db:mydbinstance \  
  --tag-keys Name Environment
```

This command produces no output.

For more information, see [Tagging Amazon RDS resources](#) in the *Amazon RDS User Guide* and [Tagging Amazon RDS resources](#) in the *Amazon Aurora User Guide*.

- For API details, see [RemoveTagsFromResource](#) in *AWS CLI Command Reference*.

reset-db-cluster-parameter-group

The following code example shows how to use `reset-db-cluster-parameter-group`.

AWS CLI

Example 1: To reset all parameters to their default values

The following `reset-db-cluster-parameter-group` example resets all parameter values in a customer-created DB cluster parameter group to their default values.

```
aws rds reset-db-cluster-parameter-group \  
  --db-cluster-parameter-group-name mydbclpg \  
  --reset-all-parameters
```

Output:

```
{  
  "DBClusterParameterGroupName": "mydbclpg"  
}
```

For more information, see [Working with DB parameter groups and DB cluster parameter groups](#) in the *Amazon Aurora User Guide*.

Example 2: To reset specific parameters to their default values

The following `reset-db-cluster-parameter-group` example resets the parameter values for specific parameters to their default values in a customer-created DB cluster parameter group.

```
aws rds reset-db-cluster-parameter-group \  
  --db-cluster-parameter-group-name mydbclpg \  
  --parameter-name myparam
```

```
--db-cluster-parameter-group-name mydbclpgy \  
--parameters "ParameterName=max_connections,ApplyMethod=immediate" \  
             "ParameterName=max_allowed_packet,ApplyMethod=immediate"
```

Output:

```
{  
  "DBClusterParameterGroupName": "mydbclpg"  
}
```

For more information, see [Working with DB parameter groups and DB cluster parameter groups](#) in the *Amazon Aurora User Guide*.

- For API details, see [ResetDbClusterParameterGroup](#) in *AWS CLI Command Reference*.

reset-db-parameter-group

The following code example shows how to use `reset-db-parameter-group`.

AWS CLI

Example 1: To reset all parameters to their default values

The following `reset-db-parameter-group` example resets all parameter values in a customer-created DB parameter group to their default values.

```
aws rds reset-db-parameter-group \  
  --db-parameter-group-name mypg \  
  --reset-all-parameters
```

Output:

```
{  
  "DBParameterGroupName": "mypg"  
}
```

For more information, see [Working with DB parameter groups](#) in the *Amazon RDS User Guide* and [Working with DB parameter groups and DB cluster parameter groups](#) in the *Amazon Aurora User Guide*.

Example 2: To reset specific parameters to their default values

The following `reset-db-parameter-group` example resets the parameter values for specific parameters to their default values in a customer-created DB parameter group.

```
aws rds reset-db-parameter-group \  
  --db-parameter-group-name mypg \  
  --parameters "ParameterName=max_connections,ApplyMethod=immediate" \  
               "ParameterName=max_allowed_packet,ApplyMethod=immediate"
```

Output:

```
{  
  "DBParameterGroupName": "mypg"  
}
```

For more information, see [Working with DB parameter groups](#) in the *Amazon RDS User Guide* and [Working with DB parameter groups and DB cluster parameter groups](#) in the *Amazon Aurora User Guide*.

- For API details, see [ResetDbParameterGroup](#) in *AWS CLI Command Reference*.

restore-db-cluster-from-s3

The following code example shows how to use `restore-db-cluster-from-s3`.

AWS CLI

To restore an Amazon Aurora DB cluster from Amazon S3

The following `restore-db-cluster-from-s3` example restores an Amazon Aurora MySQL version 5.7-compatible DB cluster from a MySQL 5.7 DB backup file in Amazon S3.

```
aws rds restore-db-cluster-from-s3 \  
  --db-cluster-identifier cluster-s3-restore \  
  --engine aurora-mysql \  
  --master-username admin \  
  --master-user-password mypassword \  
  --s3-bucket-name mybucket \  
  --s3-prefix test-backup \  
  --s3-ingestion-role-arn arn:aws:iam::123456789012:role/service-role/TestBackup \  
  --source-engine mysql \  
  --source-engine-version 5.7.28
```

Output:

```
{
  "DBCluster": {
    "AllocatedStorage": 1,
    "AvailabilityZones": [
      "us-west-2c",
      "us-west-2a",
      "us-west-2b"
    ],
    "BackupRetentionPeriod": 1,
    "DBClusterIdentifier": "cluster-s3-restore",
    "DBClusterParameterGroup": "default.aurora-mysql5.7",
    "DBSubnetGroup": "default",
    "Status": "creating",
    "Endpoint": "cluster-s3-restore.cluster-co3xyzabc123.us-
west-2.rds.amazonaws.com",
    "ReaderEndpoint": "cluster-s3-restore.cluster-ro-co3xyzabc123.us-
west-2.rds.amazonaws.com",
    "MultiAZ": false,
    "Engine": "aurora-mysql",
    "EngineVersion": "5.7.12",
    "Port": 3306,
    "MasterUsername": "admin",
    "PreferredBackupWindow": "11:15-11:45",
    "PreferredMaintenanceWindow": "thu:12:19-thu:12:49",
    "ReadReplicaIdentifiers": [],
    "DBClusterMembers": [],
    "VpcSecurityGroups": [
      {
        "VpcSecurityGroupId": "sg-#####",
        "Status": "active"
      }
    ],
    "HostedZoneId": "Z1PVIF0EXAMPLE",
    "StorageEncrypted": false,
    "DbClusterResourceId": "cluster-SU5THYQQHOWCXZZDGXREXAMPLE",
    "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:cluster-s3-
restore",
    "AssociatedRoles": [],
    "IAMDatabaseAuthenticationEnabled": false,
    "ClusterCreateTime": "2020-07-27T14:22:08.095Z",
    "EngineMode": "provisioned",
    "DeletionProtection": false,
  }
}
```

```
    "HttpEndpointEnabled": false,  
    "CopyTagsToSnapshot": false,  
    "CrossAccountClone": false,  
    "DomainMemberships": []  
  }  
}
```

For more information, see [Migrating Data from MySQL by Using an Amazon S3 Bucket](#) in the *Amazon Aurora User Guide*.

- For API details, see [RestoreDbClusterFromS3](#) in *AWS CLI Command Reference*.

restore-db-cluster-from-snapshot

The following code example shows how to use `restore-db-cluster-from-snapshot`.

AWS CLI

To restore a DB cluster from a snapshot

The following `restore-db-cluster-from-snapshot` restores an Aurora PostgreSQL DB cluster compatible with PostgreSQL version 10.7 from a DB cluster snapshot named `test-instance-snapshot`.

```
aws rds restore-db-cluster-from-snapshot \  
  --db-cluster-identifier newdbcluster \  
  --snapshot-identifier test-instance-snapshot \  
  --engine aurora-postgresql \  
  --engine-version 10.7
```

Output:

```
{  
  "DBCluster": {  
    "AllocatedStorage": 1,  
    "AvailabilityZones": [  
      "us-west-2c",  
      "us-west-2a",  
      "us-west-2b"  
    ],  
    "BackupRetentionPeriod": 7,  
    "DatabaseName": "",
```

```

    "DBClusterIdentifier": "newdbcluster",
    "DBClusterParameterGroup": "default.aurora-postgresql10",
    "DBSubnetGroup": "default",
    "Status": "creating",
    "Endpoint": "newdbcluster.cluster-#####.us-west-2.rds.amazonaws.com",
    "ReaderEndpoint": "newdbcluster.cluster-ro-#####.us-
west-2.rds.amazonaws.com",
    "MultiAZ": false,
    "Engine": "aurora-postgresql",
    "EngineVersion": "10.7",
    "Port": 5432,
    "MasterUsername": "postgres",
    "PreferredBackupWindow": "09:33-10:03",
    "PreferredMaintenanceWindow": "sun:12:22-sun:12:52",
    "ReadReplicaIdentifiers": [],
    "DBClusterMembers": [],
    "VpcSecurityGroups": [
      {
        "VpcSecurityGroupId": "sg-#####",
        "Status": "active"
      }
    ],
    "HostedZoneId": "Z1PVIF0EXAMPLE",
    "StorageEncrypted": true,
    "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/287364e4-33e3-4755-a3b0-
a1b2c3d4e5f6",
    "DbClusterResourceId": "cluster-5DSB5IFQDDUVAWUWM1EXAMPLE",
    "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:newdbcluster",
    "AssociatedRoles": [],
    "IAMDatabaseAuthenticationEnabled": false,
    "ClusterCreateTime": "2020-06-05T15:06:58.634Z",
    "EngineMode": "provisioned",
    "DeletionProtection": false,
    "HttpEndpointEnabled": false,
    "CopyTagsToSnapshot": false,
    "CrossAccountClone": false,
    "DomainMemberships": []
  }
}

```

For more information, see [Restoring from a DB Cluster Snapshot](#) in the *Amazon Aurora User Guide*.

- For API details, see [RestoreDbClusterFromSnapshot](#) in *AWS CLI Command Reference*.

restore-db-cluster-to-point-in-time

The following code example shows how to use `restore-db-cluster-to-point-in-time`.

AWS CLI

To restore a DB cluster to a specified time

The following `restore-db-cluster-to-point-in-time` example restores the DB cluster named `database-4` to the latest possible time. Using `copy-on-write` as the restore type restores the new DB cluster as a clone of the source DB cluster.

```
aws rds restore-db-cluster-to-point-in-time \  
  --source-db-cluster-identifier database-4 \  
  --db-cluster-identifier sample-cluster-clone \  
  --restore-type copy-on-write \  
  --use-latest-restorable-time
```

Output:

```
{  
  "DBCluster": {  
    "AllocatedStorage": 1,  
    "AvailabilityZones": [  
      "us-west-2c",  
      "us-west-2a",  
      "us-west-2b"  
    ],  
    "BackupRetentionPeriod": 7,  
    "DatabaseName": "",  
    "DBClusterIdentifier": "sample-cluster-clone",  
    "DBClusterParameterGroup": "default.aurora-postgresql10",  
    "DBSubnetGroup": "default",  
    "Status": "creating",  
    "Endpoint": "sample-cluster-clone.cluster-#####.us-  
west-2.rds.amazonaws.com",  
    "ReaderEndpoint": "sample-cluster-clone.cluster-ro-#####.us-  
west-2.rds.amazonaws.com",  
    "MultiAZ": false,  
    "Engine": "aurora-postgresql",  
    "EngineVersion": "10.7",  
    "Port": 5432,
```

```

    "MasterUsername": "postgres",
    "PreferredBackupWindow": "09:33-10:03",
    "PreferredMaintenanceWindow": "sun:12:22-sun:12:52",
    "ReadReplicaIdentifiers": [],
    "DBClusterMembers": [],
    "VpcSecurityGroups": [
      {
        "VpcSecurityGroupId": "sg-#####",
        "Status": "active"
      }
    ],
    "HostedZoneId": "Z1PVIIF0EXAMPLE",
    "StorageEncrypted": true,
    "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/287364e4-33e3-4755-a3b0-
a1b2c3d4e5f6",
    "DbClusterResourceId": "cluster-BIZ77GDSA2XBSTNPFW1EXAMPLE",
    "DBClusterArn": "arn:aws:rds:us-west-2:123456789012:cluster:sample-cluster-
clone",
    "AssociatedRoles": [],
    "IAMDatabaseAuthenticationEnabled": false,
    "CloneGroupId": "8d19331a-099a-45a4-b4aa-11aa22bb33cc44dd",
    "ClusterCreateTime": "2020-03-10T19:57:38.967Z",
    "EngineMode": "provisioned",
    "DeletionProtection": false,
    "HttpEndpointEnabled": false,
    "CopyTagsToSnapshot": false,
    "CrossAccountClone": false
  }
}

```

For more information, see [Restoring a DB Cluster to a Specified Time](#) in the *Amazon Aurora User Guide*.

- For API details, see [RestoreDbClusterToPointInTime](#) in *AWS CLI Command Reference*.

restore-db-instance-from-db-snapshot

The following code example shows how to use `restore-db-instance-from-db-snapshot`.

AWS CLI

To restore a DB instance from a DB snapshot

The following `restore-db-instance-from-db-snapshot` example creates a new DB instance named `db7-new-instance` with the `db.t3.small` DB instance class from the specified DB snapshot. The source DB instance from which the snapshot was taken uses a deprecated DB instance class, so you can't upgrade it.

```
aws rds restore-db-instance-from-db-snapshot \  
  --db-instance-identifier db7-new-instance \  
  --db-snapshot-identifier db7-test-snapshot \  
  --db-instance-class db.t3.small
```

Output:

```
{  
  "DBInstance": {  
    "DBInstanceIdentifier": "db7-new-instance",  
    "DBInstanceClass": "db.t3.small",  
    "Engine": "mysql",  
    "DBInstanceStatus": "creating",  
  
    ...output omitted...  
  
    "PreferredMaintenanceWindow": "mon:07:37-mon:08:07",  
    "PendingModifiedValues": {},  
    "MultiAZ": false,  
    "EngineVersion": "5.7.22",  
    "AutoMinorVersionUpgrade": true,  
    "ReadReplicaDBInstanceIdentifiers": [],  
    "LicenseModel": "general-public-license",  
  
    ...output omitted...  
  
    "DBInstanceArn": "arn:aws:rds:us-west-2:123456789012:db:db7-new-instance",  
    "IAMDatabaseAuthenticationEnabled": false,  
    "PerformanceInsightsEnabled": false,  
    "DeletionProtection": false,  
    "AssociatedRoles": []  
  }  
}
```

For more information, see [Restoring from a DB Snapshot](#) in the *Amazon RDS User Guide*.

- For API details, see [RestoreDbInstanceFromDbSnapshot](#) in *AWS CLI Command Reference*.

restore-db-instance-from-s3

The following code example shows how to use `restore-db-instance-from-s3`.

AWS CLI

To restore a DB instance from a backup in Amazon S3

The following `restore-db-instance-from-s3` example creates a new DB instance named `restored-test-instance` from an existing backup in the `my-backups` S3 bucket.

```
aws rds restore-db-instance-from-s3 \  
  --db-instance-identifier restored-test-instance \  
  --allocated-storage 250 --db-instance-class db.m4.large --engine mysql \  
  --master-username master --master-user-password secret99 \  
  --s3-bucket-name my-backups --s3-ingestion-role-arn \  
  arn:aws:iam::123456789012:role/my-role \  
  --source-engine mysql --source-engine-version 5.6.27
```

- For API details, see [RestoreDbInstanceFromS3](#) in *AWS CLI Command Reference*.

restore-db-instance-to-point-in-time

The following code example shows how to use `restore-db-instance-to-point-in-time`.

AWS CLI

Example 1: To restore a DB instance to a point in time

The following `restore-db-instance-to-point-in-time` example restores `test-instance` to a new DB instance named `restored-test-instance`, as of the specified time.

```
aws rds restore-db-instance-to-point-in-time \  
  --source-db-instance-identifier test-instance \  
  --target-db-instance restored-test-instance \  
  --restore-time 2018-07-30T23:45:00.000Z
```

Output:

```
{
```



```

    "DBInstance": {
      "AllocatedStorage": 20,
      "DBInstanceArn": "arn:aws:rds:us-east-1:123456789012:db:restored-test-
instance",
      "DBInstanceStatus": "creating",
      "DBInstanceIdentifier": "restored-test-instance",
      ...some output omitted...
    }
  }
}

```

For more information, see [Restoring a DB instance to a specified time](#) in the *Amazon RDS User Guide*.

Example 2: To restore a DB instance to a specified time from a replicated backup

The following `restore-db-instance-to-point-in-time` example restores an Oracle DB instance to the specified time from a replicated automated backup.

```

aws rds restore-db-instance-to-point-in-time \
  --source-db-instance-automated-backups-arn "arn:aws:rds:us-
west-2:123456789012:auto-backup:ab-jkib2gfq5rv7replzadausbrktni2bn4example" \
  --target-db-instance-identifier myorclinstance-from-replicated-backup \
  --restore-time 2020-12-08T18:45:00.000Z

```

Output:

```

{
  "DBInstance": {
    "DBInstanceIdentifier": "myorclinstance-from-replicated-backup",
    "DBInstanceClass": "db.t3.micro",
    "Engine": "oracle-se2",
    "DBInstanceStatus": "creating",
    "MasterUsername": "admin",
    "DBName": "ORCL",
    "AllocatedStorage": 20,
    "PreferredBackupWindow": "07:45-08:15",
    "BackupRetentionPeriod": 14,
    ... some output omitted ...
    "DbiResourceId": "db-KGLXG75BGVIWKQT7NQ4EXAMPLE",
    "CACertificateIdentifier": "rds-ca-2019",
    "DomainMemberships": [],
    "CopyTagsToSnapshot": false,
  }
}

```

```
    "MonitoringInterval": 0,
    "DBInstanceArn": "arn:aws:rds:us-west-2:123456789012:db:myorclinstance-from-
replicated-backup",
    "IAMDatabaseAuthenticationEnabled": false,
    "PerformanceInsightsEnabled": false,
    "DeletionProtection": false,
    "AssociatedRoles": [],
    "TagList": []
  }
}
```

For more information, see [Restoring to a specified time from a replicated backup](#) in the *Amazon RDS User Guide*.

- For API details, see [RestoreDbInstanceToPointInTime](#) in *AWS CLI Command Reference*.

start-activity-stream

The following code example shows how to use `start-activity-stream`.

AWS CLI

To start a database activity stream

The following `start-activity-stream` example starts an asynchronous activity stream to monitor an Aurora cluster named `my-pg-cluster`.

```
aws rds start-activity-stream \
  --region us-east-1 \
  --mode async \
  --kms-key-id arn:aws:kms:us-east-1:1234567890123:key/a12c345d-6ef7-890g-
h123-456i789jk011 \
  --resource-arn arn:aws:rds:us-east-1:1234567890123:cluster:my-pg-cluster \
  --apply-immediately
```

Output:

```
{
  "KmsKeyId": "arn:aws:kms:us-east-1:1234567890123:key/a12c345d-6ef7-890g-
h123-456i789jk011",
  "KinesisStreamName": "aws-rds-das-cluster-0ABCDEFGH11JKLM2NOPQ3R4S",
  "Status": "starting",
```

```
"Mode": "async",
"ApplyImmediately": true
}
```

For more information, see [Starting a database activity stream](#) in the *Amazon Aurora User Guide*.

- For API details, see [StartActivityStream](#) in *AWS CLI Command Reference*.

start-db-cluster

The following code example shows how to use `start-db-cluster`.

AWS CLI

To start a DB cluster

The following `start-db-cluster` example starts a DB cluster and its DB instances.

```
aws rds start-db-cluster \
  --db-cluster-identifier mydbcluster
```

Output:

```
{
  "DBCluster": {
    "AllocatedStorage": 1,
    "AvailabilityZones": [
      "us-east-1a",
      "us-east-1e",
      "us-east-1b"
    ],
    "BackupRetentionPeriod": 1,
    "DatabaseName": "mydb",
    "DBClusterIdentifier": "mydbcluster",
    ...some output truncated...
  }
}
```

For more information, see [Stopping and starting an Amazon Aurora DB cluster](#) in the *Amazon Aurora User Guide*.

- For API details, see [StartDbCluster](#) in *AWS CLI Command Reference*.

start-db-instance-automated-backups-replication

The following code example shows how to use `start-db-instance-automated-backups-replication`.

AWS CLI

To enable cross-Region automated backups

The following `start-db-instance-automated-backups-replication` example replicates automated backups from a DB instance in the US East (N. Virginia) Region to US West (Oregon). The backup retention period is 14 days.

```
aws rds start-db-instance-automated-backups-replication \  
  --region us-west-2 \  
  --source-db-instance-arn "arn:aws:rds:us-east-1:123456789012:db:new-orcl-db" \  
  --backup-retention-period 14
```

Output:

```
{  
  "DBInstanceAutomatedBackup": {  
    "DBInstanceArn": "arn:aws:rds:us-east-1:123456789012:db:new-orcl-db",  
    "DbiResourceId": "db-JKIB2GFQ5RV7REPLZA4EXAMPLE",  
    "Region": "us-east-1",  
    "DBInstanceIdentifier": "new-orcl-db",  
    "RestoreWindow": {},  
    "AllocatedStorage": 20,  
    "Status": "pending",  
    "Port": 1521,  
    "InstanceCreateTime": "2020-12-04T15:28:31Z",  
    "MasterUsername": "admin",  
    "Engine": "oracle-se2",  
    "EngineVersion": "12.1.0.2.v21",  
    "LicenseModel": "bring-your-own-license",  
    "OptionGroupName": "default:oracle-se2-12-1",  
    "Encrypted": false,  
    "StorageType": "gp2",  
    "IAMDatabaseAuthenticationEnabled": false,  
    "BackupRetentionPeriod": 14,  
    "DBInstanceAutomatedBackupsArn": "arn:aws:rds:us-west-2:123456789012:auto-backup:ab-jkib2gfgq5rv7replzadtausbrktni2bn4example"  
  }  
}
```

```
}
```

For more information, see [Enabling cross-Region automated backups](#) in the *Amazon RDS User Guide*.

- For API details, see [StartDbInstanceAutomatedBackupsReplication](#) in *AWS CLI Command Reference*.

start-db-instance

The following code example shows how to use `start-db-instance`.

AWS CLI

To start a DB instance

The following `start-db-instance` example starts the specified DB instance.

```
aws rds start-db-instance \  
  --db-instance-identifier test-instance
```

Output:

```
{  
  "DBInstance": {  
    "DBInstanceStatus": "starting",  
    ...some output truncated...  
  }  
}
```

- For API details, see [StartDbInstance](#) in *AWS CLI Command Reference*.

start-export-task

The following code example shows how to use `start-export-task`.

AWS CLI

To export a snapshot to Amazon S3

The following `start-export-task` example exports a DB snapshot named `db5-snapshot-test` to the Amazon S3 bucket named `mybucket`.

```
aws rds start-export-task \  
  --export-task-identifier my-s3-export \  
  --source-arn arn:aws:rds:us-west-2:123456789012:snapshot:db5-snapshot-test \  
  --s3-bucket-name mybucket \  
  --iam-role-arn arn:aws:iam::123456789012:role/service-role/ExportRole \  
  --kms-key-id arn:aws:kms:us-west-2:123456789012:key/abcd0000-7fca-4128-82f2-  
  aabbccddeeff
```

Output:

```
{  
  "ExportTaskIdentifier": "my-s3-export",  
  "SourceArn": "arn:aws:rds:us-west-2:123456789012:snapshot:db5-snapshot-test",  
  "SnapshotTime": "2020-03-27T20:48:42.023Z",  
  "S3Bucket": "mybucket",  
  "IamRoleArn": "arn:aws:iam::123456789012:role/service-role/ExportRole",  
  "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/abcd0000-7fca-4128-82f2-  
  aabbccddeeff",  
  "Status": "STARTING",  
  "PercentProgress": 0,  
  "TotalExtractedDataInGB": 0  
}
```

For more information, see [Exporting a Snapshot to an Amazon S3 Bucket](#) in the *Amazon RDS User Guide*.

- For API details, see [StartExportTask](#) in *AWS CLI Command Reference*.

stop-activity-stream

The following code example shows how to use stop-activity-stream.

AWS CLI

To stop a database activity stream

The following stop-activity-stream example stops an activity stream in an Aurora cluster named my-pg-cluster.

```
aws rds stop-activity-stream \  
  --region us-east-1 \  
  --cluster-id my-pg-cluster
```

```
--resource-arn arn:aws:rds:us-east-1:1234567890123:cluster:my-pg-cluster \  
--apply-immediately
```

Output:

```
{  
  "KmsKeyId": "arn:aws:kms:us-east-1:1234567890123:key/a12c345d-6ef7-890g-  
h123-456i789jk011",  
  "KinesisStreamName": "aws-rds-das-cluster-0ABCDEFGH11JKLM2NOPQ3R4S",  
  "Status": "stopping"  
}
```

For more information, see [Stopping an activity stream](#) in the *Amazon Aurora User Guide*.

- For API details, see [StopActivityStream](#) in *AWS CLI Command Reference*.

stop-db-cluster

The following code example shows how to use `stop-db-cluster`.

AWS CLI

To stop a DB cluster

The following `stop-db-cluster` example stops a DB cluster and its DB instances.

```
aws rds stop-db-cluster \  
--db-cluster-identifier mydbcluster
```

Output:

```
{  
  "DBCluster": {  
    "AllocatedStorage": 1,  
    "AvailabilityZones": [  
      "us-east-1a",  
      "us-east-1e",  
      "us-east-1b"  
    ],  
    "BackupRetentionPeriod": 1,  
    "DatabaseName": "mydb",  
    "DBClusterIdentifier": "mydbcluster",
```

```
    ...some output truncated...
  }
}
```

For more information, see [Stopping and starting an Amazon Aurora DB cluster](#) in the *Amazon Aurora User Guide*.

- For API details, see [StopDbCluster](#) in *AWS CLI Command Reference*.

stop-db-instance-automated-backups-replication

The following code example shows how to use `stop-db-instance-automated-backups-replication`.

AWS CLI

To stop replicating automated backups

The following `stop-db-instance-automated-backups-replication` ends replication of automated backups to the US West (Oregon) Region. Replicated backups are retained according to the set backup retention period.

```
aws rds stop-db-instance-automated-backups-replication \
  --region us-west-2 \
  --source-db-instance-arn "arn:aws:rds:us-east-1:123456789012:db:new-orcl-db"
```

Output:

```
{
  "DBInstanceAutomatedBackup": {
    "DBInstanceArn": "arn:aws:rds:us-east-1:123456789012:db:new-orcl-db",
    "DbiResourceId": "db-JKIB2GFQ5RV7REPLZA4EXAMPLE",
    "Region": "us-east-1",
    "DBInstanceIdentifier": "new-orcl-db",
    "RestoreWindow": {
      "EarliestTime": "2020-12-04T23:13:21.030Z",
      "LatestTime": "2020-12-07T19:59:57Z"
    },
    "AllocatedStorage": 20,
    "Status": "replicating",
    "Port": 1521,
    "InstanceCreateTime": "2020-12-04T15:28:31Z",
```



```
"MasterUsername": "admin",
"Engine": "oracle-se2",
"EngineVersion": "12.1.0.2.v21",
"LicenseModel": "bring-your-own-license",
"OptionGroupName": "default:oracle-se2-12-1",
"Encrypted": false,
"StorageType": "gp2",
"IAMDatabaseAuthenticationEnabled": false,
"BackupRetentionPeriod": 7,
"DBInstanceAutomatedBackupsArn": "arn:aws:rds:us-west-2:123456789012:auto-
backup:ab-jkib2gfg5rv7replzadabrktni2bn4example"
}
}
```

For more information, see [Stopping automated backup replication](#) in the *Amazon RDS User Guide*.

- For API details, see [StopDbInstanceAutomatedBackupsReplication](#) in *AWS CLI Command Reference*.

stop-db-instance

The following code example shows how to use `stop-db-instance`.

AWS CLI

To stop a DB instance

The following `stop-db-instance` example stops the specified DB instance.

```
aws rds stop-db-instance \
  --db-instance-identifier test-instance
```

Output:

```
{
  "DBInstance": {
    "DBInstanceStatus": "stopping",
    ...some output truncated...
  }
}
```

- For API details, see [StopDbInstance](#) in *AWS CLI Command Reference*.

switchover-blue-green-deployment

The following code example shows how to use `switchover-blue-green-deployment`.

AWS CLI

Example 1: To switch a blue/green deployment for an RDS DB instance

The following `switchover-blue-green-deployment` example promotes the specified green environment as the new production environment.

```
aws rds switchover-blue-green-deployment \  
  --blue-green-deployment-identifier bgd-wi89nwzglccsfake \  
  --switchover-timeout 300
```

Output:

```
{  
  "BlueGreenDeployment": {  
    "BlueGreenDeploymentIdentifier": "bgd-v53303651eexfake",  
    "BlueGreenDeploymentName": "bgd-cli-test-instance",  
    "Source": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance",  
    "Target": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance-green-  
blhile",  
    "SwitchoverDetails": [  
      {  
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-  
instance",  
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-  
instance-green-blhile",  
        "Status": "AVAILABLE"  
      },  
      {  
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-  
instance-replica-1",  
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-  
instance-replica-1-green-k5fv7u",  
        "Status": "AVAILABLE"  
      },  
      {
```

```

        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-2",
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-2-green-ggsh8m",
        "Status": "AVAILABLE"
    },
    {
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-3",
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-db-
instance-replica-3-green-o2vwm0",
        "Status": "AVAILABLE"
    }
],
"Tasks": [
    {
        "Name": "CREATING_READ_REPLICA_OF_SOURCE",
        "Status": "COMPLETED"
    },
    {
        "Name": "DB_ENGINE_VERSION_UPGRADE",
        "Status": "COMPLETED"
    },
    {
        "Name": "CONFIGURE_BACKUPS",
        "Status": "COMPLETED"
    },
    {
        "Name": "CREATING_TOPOLOGY_OF_SOURCE",
        "Status": "COMPLETED"
    }
],
"Status": "SWITCHOVER_IN_PROGRESS",
"CreateTime": "2022-02-25T22:33:22.225000+00:00"
}
}

```

For more information, see [Switching a blue/green deployment](#) in the *Amazon RDS User Guide*.

Example 2: To promote a blue/green deployment for an Aurora MySQL DB cluster

The following switchover-blue-green-deployment example promotes the specified green environment as the new production environment.

```
aws rds switchover-blue-green-deployment \  
  --blue-green-deployment-identifier bgd-wi89nwzglccsfake \  
  --switchover-timeout 300
```

Output:

```
{  
  "BlueGreenDeployment": {  
    "BlueGreenDeploymentIdentifier": "bgd-wi89nwzglccsfake",  
    "BlueGreenDeploymentName": "my-blue-green-deployment",  
    "Source": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-  
cluster",  
    "Target": "arn:aws:rds:us-east-1:123456789012:cluster:my-aurora-mysql-  
cluster-green-3ud8z6",  
    "SwitchoverDetails": [  
      {  
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-  
aurora-mysql-cluster",  
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster:my-  
aurora-mysql-cluster-green-3ud8z6",  
        "Status": "AVAILABLE"  
      },  
      {  
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-  
mysql-cluster-1",  
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-  
mysql-cluster-1-green-bvxc73",  
        "Status": "AVAILABLE"  
      },  
      {  
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-  
mysql-cluster-2",  
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-  
mysql-cluster-2-green-7wc4ie",  
        "Status": "AVAILABLE"  
      },  
      {  
        "SourceMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-  
mysql-cluster-3",  
        "TargetMember": "arn:aws:rds:us-east-1:123456789012:db:my-aurora-  
mysql-cluster-3-green-p4xxkz",  
        "Status": "AVAILABLE"  
      },  
    ]  
  }  
}
```

```

    {
      "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint",
      "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-excluded-member-endpoint-green-np1ik1",
      "Status": "AVAILABLE"
    },
    {
      "SourceMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint",
      "TargetMember": "arn:aws:rds:us-east-1:123456789012:cluster-
endpoint:my-reader-endpoint-green-miszlf",
      "Status": "AVAILABLE"
    }
  ],
  "Tasks": [
    {
      "Name": "CREATING_READ_REPLICA_OF_SOURCE",
      "Status": "COMPLETED"
    },
    {
      "Name": "DB_ENGINE_VERSION_UPGRADE",
      "Status": "COMPLETED"
    },
    {
      "Name": "CREATE_DB_INSTANCES_FOR_CLUSTER",
      "Status": "COMPLETED"
    },
    {
      "Name": "CREATE_CUSTOM_ENDPOINTS",
      "Status": "COMPLETED"
    }
  ],
  "Status": "SWITCHOVER_IN_PROGRESS",
  "CreateTime": "2022-02-25T22:38:49.522000+00:00"
}

```

For more information, see [Switching a blue/green deployment](#) in the *Amazon Aurora User Guide*.

- For API details, see [SwitchoverBlueGreenDeployment](#) in *AWS CLI Command Reference*.

Amazon RDS Data Service examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon RDS Data Service.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

batch-execute-statement

The following code example shows how to use batch-execute-statement.

AWS CLI

To execute a batch SQL statement

The following batch-execute-statement example executes a batch SQL statement over an array of data with a parameter set.

```
aws rds-data batch-execute-statement \
  --resource-arn "arn:aws:rds:us-west-2:123456789012:cluster:mydbcluster" \
  --database "mydb" \
  --secret-arn "arn:aws:secretsmanager:us-west-2:123456789012:secret:mysecret" \
  --sql "insert into mytable values (:id, :val)" \
  --parameter-sets "[[{"name\": \"id\", \"value\": {\"longValue\": 1}}, {\"name\": \"val\", \"value\": {\"stringValue\": \"ValueOne\"}}], [\"name\": \"id\", \"value\": {\"longValue\": 2}}, {\"name\": \"val\", \"value\": {\"stringValue\": \"ValueTwo\"}}], [\"name\": \"id\", \"value\": {\"longValue\": 3}}, {\"name\": \"val\", \"value\": {\"stringValue\": \"ValueThree\"}}]]]"
```

This command produces no output.

For more information, see [Using the Data API for Aurora Serverless](#) in the *Amazon RDS User Guide*.

- For API details, see [BatchExecuteStatement](#) in *AWS CLI Command Reference*.

begin-transaction

The following code example shows how to use `begin-transaction`.

AWS CLI

To start a SQL transaction

The following `begin-transaction` example starts a SQL transaction.

```
aws rds-data begin-transaction \  
  --resource-arn "arn:aws:rds:us-west-2:123456789012:cluster:mydbcluster" \  
  --database "mydb" \  
  --secret-arn "arn:aws:secretsmanager:us-west-2:123456789012:secret:mysecret"
```

Output:

```
{  
  "transactionId": "ABC1234567890xyz"  
}
```

For more information, see [Using the Data API for Aurora Serverless](#) in the *Amazon RDS User Guide*.

- For API details, see [BeginTransaction](#) in *AWS CLI Command Reference*.

commit-transaction

The following code example shows how to use `commit-transaction`.

AWS CLI

To commit a SQL transaction

The following `commit-transaction` example ends the specified SQL transaction and commits the changes that you made as part of it.

```
aws rds-data commit-transaction \  
  --resource-arn "arn:aws:rds:us-west-2:123456789012:cluster:mydbcluster" \  
  --secret-arn "arn:aws:secretsmanager:us-west-2:123456789012:secret:mysecret" \  
  --transaction-id "ABC1234567890xyz"
```

Output:

```
{  
  "transactionStatus": "Transaction Committed"  
}
```

For more information, see [Using the Data API for Aurora Serverless](#) in the *Amazon RDS User Guide*.

- For API details, see [CommitTransaction](#) in *AWS CLI Command Reference*.

execute-statement

The following code example shows how to use `execute-statement`.

AWS CLI

Example 1: To execute a SQL statement that is part of a transaction

The following `execute-statement` example runs a SQL statement that is part of a transaction.

```
aws rds-data execute-statement \  
  --resource-arn "arn:aws:rds:us-west-2:123456789012:cluster:mydbcluster" \  
  --database "mydb" \  
  --secret-arn "arn:aws:secretsmanager:us-west-2:123456789012:secret:mysecret" \  
  --sql "update mytable set quantity=5 where id=201" \  
  --transaction-id "ABC1234567890xyz"
```

Output:

```
{  
  "numberOfRecordsUpdated": 1  
}
```

Example 2: To execute a SQL statement with parameters

The following `execute-statement` example runs a SQL statement with parameters.

```
aws rds-data execute-statement \  
  --resource-arn "arn:aws:rds:us-east-1:123456789012:cluster:mydbcluster" \  
  --database "mydb" \  
  --secret-arn "arn:aws:secretsmanager:us-east-1:123456789012:secret:mysecret" \  
  --sql "insert into mytable values (:id, :val)" \  
  --parameters "[{\"name\": \"id\", \"value\": {\"longValue\": 1}}, {\"name\":  
  \"val\", \"value\": {\"stringValue\": \"value1\"}}]"
```

Output:

```
{  
  "numberOfRecordsUpdated": 1  
}
```

For more information, see [Using the Data API for Aurora Serverless](#) in the *Amazon RDS User Guide*.

- For API details, see [ExecuteStatement](#) in *AWS CLI Command Reference*.

rollback-transaction

The following code example shows how to use `rollback-transaction`.

AWS CLI

To roll back a SQL transaction

The following `rollback-transaction` example rolls back the specified SQL transaction.

```
aws rds-data rollback-transaction \  
  --resource-arn "arn:aws:rds:us-west-2:123456789012:cluster:mydbcluster" \  
  --secret-arn "arn:aws:secretsmanager:us-west-2:123456789012:secret:mysecret" \  
  --transaction-id "ABC1234567890xyz"
```

Output:

```
{  
  "transactionStatus": "Rollback Complete"  
}
```

For more information, see [Using the Data API for Aurora Serverless](#) in the *Amazon RDS User Guide*.

- For API details, see [RollbackTransaction](#) in *AWS CLI Command Reference*.

Amazon RDS Performance Insights examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon RDS Performance Insights.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

describe-dimension-keys

The following code example shows how to use describe-dimension-keys.

AWS CLI

To describe dimension keys

This example requests the names of all wait events. The data is summarized by event name, and the aggregate values of those events over the specified time period.

Command:

```
aws pi describe-dimension-keys --service-type RDS --identifier db-
LKCG0BK26374TPTDFX0IWVCPMM --start-time 1527026400 --end-time 1527080400 --metric
db.load.avg --group-by '{"Group":"db.wait_event"}
```

Output:

```
{
  "AlignedEndTime": 1.5270804E9,
  "AlignedStartTime": 1.5270264E9,
  "Keys": [
    {
      "Dimensions": {"db.wait_event.name": "wait/synch/mutex/innodb/aurora_lock_thread_slot_futex"},
      "Total": 0.05906906851195666
    },
    {
      "Dimensions": {"db.wait_event.name": "wait/io/aurora_redo_log_flush"},
      "Total": 0.015824722186149193
    },
    {
      "Dimensions": {"db.wait_event.name": "CPU"},
      "Total": 0.008014396230265477
    },
    {
      "Dimensions": {"db.wait_event.name": "wait/io/aurora_respond_to_client"},
      "Total": 0.0036361612526204477
    },
    {
      "Dimensions": {"db.wait_event.name": "wait/io/table/sql/handler"},
      "Total": 0.0019108398419382965
    },
    {
      "Dimensions": {"db.wait_event.name": "wait/synch/cond/mysys/my_thread_var::suspend"},
      "Total": 8.533847837782684E-4
    },
    {
      "Dimensions": {"db.wait_event.name": "wait/io/file/csv/data"},
      "Total": 6.864181956477376E-4
    },
    {
      "Dimensions": {"db.wait_event.name": "Unknown"},
      "Total": 3.895887056379051E-4
    },
    {
      "Dimensions": {"db.wait_event.name": "wait/synch/mutex/sql/FILE_AS_TABLE::LOCK_shim_lists"},

```

```

        "Total": 3.710368625122906E-5
      },
      {
        "Dimensions": {"db.wait_event.name": "wait/lock/table/sql/handler"},
        "Total": 0
      }
    ]
  }

```

- For API details, see [DescribeDimensionKeys](#) in *AWS CLI Command Reference*.

get-resource-metrics

The following code example shows how to use `get-resource-metrics`.

AWS CLI

To get resource metrics

This example requests data points for the `db.wait_event` dimension group, and for the `db.wait_event.name` dimension within that group. In the response, the relevant data points are grouped by the requested dimension (`db.wait_event.name`).

Command:

```

aws pi get-resource-metrics --service-type RDS --identifier db-
LKCG0BK26374TPTDFX0IWVCPM --start-time 1527026400 --end-time 1527080400 --period-
in-seconds 300 --metric db.load.avg --metric-queries file://metric-queries.json

```

The arguments for `--metric-queries` are stored in a JSON file, `metric-queries.json`. Here are the contents of that file:

```

[
  {
    "Metric": "db.load.avg",
    "GroupBy": {
      "Group": "db.wait_event"
    }
  }
]

```

Output:

```

{
  "AlignedEndTime": 1.5270804E9,
  "AlignedStartTime": 1.5270264E9,
  "Identifier": "db-LKCG0BK26374TPTDFX0IWVCPM",
  "MetricList": [
    {
      "Key": {
        "Metric": "db.load.avg"
      },
      "DataPoints": [
        {
          "Timestamp": 1527026700.0,
          "Value": 1.3533333333333333
        },
        {
          "Timestamp": 1527027000.0,
          "Value": 0.88
        },
        <...remaining output omitted...>
      ]
    },
    {
      "Key": {
        "Metric": "db.load.avg",
        "Dimensions": {
          "db.wait_event.name": "wait/synch/mutex/innodb/
aurora_lock_thread_slot_futex"
        }
      },
      "DataPoints": [
        {
          "Timestamp": 1527026700.0,
          "Value": 0.8566666666666667
        },
        {
          "Timestamp": 1527027000.0,
          "Value": 0.8633333333333333
        },
        <...remaining output omitted...>
      ]
    },
    <...remaining output omitted...>
  ]
}

```

```
]
}
```

- For API details, see [GetResourceMetrics](#) in *AWS CLI Command Reference*.

Amazon Redshift examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon Redshift.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

accept-reserved-node-exchange

The following code example shows how to use `accept-reserved-node-exchange`.

AWS CLI

To accept reserved node exchange

The following `accept-reserved-node-exchange` example accepts exchange of a DC1 reserved node for a DC2 reserved node.

```
aws redshift accept-reserved-node-exchange /
  --reserved-node-id 12345678-12ab-12a1-1a2a-12ab-12a12EXAMPLE /
  --target-reserved-node-offering-id 12345678-12ab-12a1-1a2a-12ab-12a12EXAMPLE
```

Output:

```
{
  "ExchangedReservedNode": {
    "ReservedNodeId": "12345678-12ab-12a1-1a2a-12ab-12a12EXAMPLE",
    "ReservedNodeOfferingId": "12345678-12ab-12a1-1a2a-12ab-12a12EXAMPLE",
    "NodeType": "dc2.large",
    "StartTime": "2019-12-06T21:17:26Z",
    "Duration": 31536000,
    "FixedPrice": 0.0,
    "UsagePrice": 0.0,
    "CurrencyCode": "USD",
    "NodeCount": 1,
    "State": "exchanging",
    "OfferingType": "All Upfront",
    "RecurringCharges": [
      {
        "RecurringChargeAmount": 0.0,
        "RecurringChargeFrequency": "Hourly"
      }
    ],
    "ReservedNodeOfferingType": "Regular"
  }
}
```

For more information, see [Upgrading Reserved Nodes With the AWS CLI](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [AcceptReservedNodeExchange](#) in *AWS CLI Command Reference*.

authorize-cluster-security-group-ingress

The following code example shows how to use `authorize-cluster-security-group-ingress`.

AWS CLI

Authorizing Access to an EC2 Security Group This example authorizes access to a named Amazon EC2 security group.
Command:

```
aws redshift authorize-cluster-security-group-ingress --cluster-security-group-name
mysecuritygroup --ec2-security-group-name myec2securitygroup --ec2-security-group-
owner-id 123445677890
```

Authorizing Access to a CIDR rangeThis example authorizes access to a CIDR range.**Command:**

```
aws redshift authorize-cluster-security-group-ingress --cluster-security-group-name
mysecuritygroup --cidrip 192.168.100.100/32
```

- For API details, see [AuthorizeClusterSecurityGroupIngress](#) in *AWS CLI Command Reference*.

authorize-snapshot-access

The following code example shows how to use `authorize-snapshot-access`.

AWS CLI

Authorize an AWS Account to Restore a SnapshotThis example authorizes the AWS account 444455556666 to restore the snapshot `my-snapshot-id`. By default, the output is in JSON format.**Command:**

```
aws redshift authorize-snapshot-access --snapshot-id my-snapshot-id --account-with-
restore-access 444455556666
```

Result:

```
{
  "Snapshot": {
    "Status": "available",
    "SnapshotCreateTime": "2013-07-17T22:04:18.947Z",
    "EstimatedSecondsToCompletion": 0,
    "AvailabilityZone": "us-east-1a",
    "ClusterVersion": "1.0",
    "MasterUsername": "adminuser",
    "Encrypted": false,
    "OwnerAccount": "111122223333",
    "BackupProgressInMegabytes": 11.0,
    "ElapsedTimeInSeconds": 0,
    "DBName": "dev",
    "CurrentBackupRateInMegabytesPerSecond": 0.1534,
    "ClusterCreateTime": "2013-01-22T21:59:29.559Z",
    "ActualIncrementalBackupSizeInMegabytes": 11.0,
    "SnapshotType": "manual",
    "NodeType": "dw.hs1.xlarge",
    "ClusterIdentifier": "mycluster",
```



```
    "TotalBackupSizeInMegabytes": 20.0,  
    "Port": 5439,  
    "NumberOfNodes": 2,  
    "SnapshotIdentifier": "my-snapshot-id"  
  }  
}
```

- For API details, see [AuthorizeSnapshotAccess](#) in *AWS CLI Command Reference*.

batch-delete-cluster-snapshots

The following code example shows how to use `batch-delete-cluster-snapshots`.

AWS CLI

To delete a set of cluster snapshots

The following `batch-delete-cluster-snapshots` example deletes a set of manual cluster snapshots.

```
aws redshift batch-delete-cluster-snapshots \  
    --identifiers SnapshotIdentifier=mycluster-2019-11-06-14-12  
    SnapshotIdentifier=mycluster-2019-11-06-14-20
```

Output:

```
{  
  "Resources": [  
    "mycluster-2019-11-06-14-12",  
    "mycluster-2019-11-06-14-20"  
  ]  
}
```

For more information, see [Amazon Redshift Snapshots](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [BatchDeleteClusterSnapshots](#) in *AWS CLI Command Reference*.

batch-modify-cluster-snapshots

The following code example shows how to use `batch-modify-cluster-snapshots`.

AWS CLI

To modify a set of cluster snapshots

The following `batch-modify-cluster-snapshots` example modifies the settings for a set of cluster snapshots.

```
aws redshift batch-modify-cluster-snapshots \  
  --snapshot-identifier-list mycluster-2019-11-06-16-31 mycluster-2019-11-06-16-32 \  
  \  
  --manual-snapshot-retention-period 30
```

Output:

```
{  
  "Resources": [  
    "mycluster-2019-11-06-16-31",  
    "mycluster-2019-11-06-16-32"  
  ],  
  "Errors": [],  
  "ResponseMetadata": {  
    "RequestId": "12345678-12ab-12a1-1a2a-12ab-12a12EXAMPLE",  
    "HTTPStatusCode": 200,  
    "HTTPHeaders": {  
      "x-amzn-requestid": "12345678-12ab-12a1-1a2a-12ab-12a12EXAMPLE",  
      "content-type": "text/xml",  
      "content-length": "480",  
      "date": "Sat, 07 Dec 2019 00:36:09 GMT",  
      "connection": "keep-alive"  
    },  
    "RetryAttempts": 0  
  }  
}
```

For more information, see [Amazon Redshift Snapshots](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [BatchModifyClusterSnapshots](#) in *AWS CLI Command Reference*.

cancel-resize

The following code example shows how to use `cancel-resize`.

AWS CLI

To cancel resize of a cluster

The following `cancel-resize` example cancels a classic resize operation for a cluster.

```
aws redshift cancel-resize \  
  --cluster-identifier mycluster
```

Output:

```
{  
  "TargetNodeType": "dc2.large",  
  "TargetNumberOfNodes": 2,  
  "TargetClusterType": "multi-node",  
  "Status": "CANCELLING",  
  "ResizeType": "ClassicResize",  
  "TargetEncryptionType": "NONE"  
}
```

For more information, see [Resizing Clusters in Amazon Redshift](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [CancelResize](#) in *AWS CLI Command Reference*.

copy-cluster-snapshot

The following code example shows how to use `copy-cluster-snapshot`.

AWS CLI

Get a Description of All Cluster Versions This example returns a description of all cluster versions. By default, the output is in JSON format. Command:

```
aws redshift copy-cluster-snapshot --source-snapshot-identifier  
  cm:examplecluster-2013-01-22-19-27-58 --target-snapshot-identifier my-saved-  
  snapshot-copy
```

Result:

```
{  
  "Snapshot": {
```

```

    "Status": "available",
    "SnapshotCreateTime": "2013-01-22T19:27:58.931Z",
    "AvailabilityZone": "us-east-1c",
    "ClusterVersion": "1.0",
    "MasterUsername": "adminuser",
    "DBName": "dev",
    "ClusterCreateTime": "2013-01-22T19:23:59.368Z",
    "SnapshotType": "manual",
    "NodeType": "dw.hs1.xlarge",
    "ClusterIdentifier": "examplecluster",
    "Port": 5439,
    "NumberOfNodes": "2",
    "SnapshotIdentifier": "my-saved-snapshot-copy"
  },
  "ResponseMetadata": {
    "RequestId": "3b279691-64e3-11e2-bec0-17624ad140dd"
  }
}

```

- For API details, see [CopyClusterSnapshot](#) in *AWS CLI Command Reference*.

create-cluster-parameter-group

The following code example shows how to use `create-cluster-parameter-group`.

AWS CLI

Create a Cluster Parameter Group This example creates a new cluster parameter group.
Command:

```

aws redshift create-cluster-parameter-group --parameter-group-name
myclusterparametergroup --parameter-group-family redshift-1.0 --description "My
first cluster parameter group"

```

Result:

```

{
  "ClusterParameterGroup": {
    "ParameterGroupFamily": "redshift-1.0",
    "Description": "My first cluster parameter group",
    "ParameterGroupName": "myclusterparametergroup"
  },
}

```

```

    "ResponseMetadata": {
      "RequestId": "739448f0-64cc-11e2-8f7d-3b939af52818"
    }
  }
}

```

- For API details, see [CreateClusterParameterGroup](#) in *AWS CLI Command Reference*.

create-cluster-security-group

The following code example shows how to use `create-cluster-security-group`.

AWS CLI

Creating a Cluster Security Group This example creates a new cluster security group. By default, the output is in JSON format. Command:

```
aws redshift create-cluster-security-group --cluster-security-group-name
mysecuritygroup --description "This is my cluster security group"
```

Result:

```

{
  "create_cluster_security_group_response": {
    "create_cluster_security_group_result": {
      "cluster_security_group": {
        "description": "This is my cluster security group",
        "owner_id": "300454760768",
        "cluster_security_group_name": "mysecuritygroup",
        "ec2_security_groups": \[],
        "ip_ranges": \[]
      }
    },
    "response_metadata": {
      "request_id": "5df486a0-343a-11e2-b0d8-d15d0ef48549"
    }
  }
}

```

You can also obtain the same information in text format using the `--output text` option. Command:

`--output text` option. Command:

option.Command:

```
aws redshift create-cluster-security-group --cluster-security-group-name
mysecuritygroup --description "This is my cluster security group" --output text
```

Result:

```
This is my cluster security group 300454760768 mysecuritygroup
a0c0bfab-343a-11e2-95d2-c3dc9fe8ab57
```

- For API details, see [CreateClusterSecurityGroup](#) in *AWS CLI Command Reference*.

create-cluster-snapshot

The following code example shows how to use `create-cluster-snapshot`.

AWS CLI

Create a Cluster Snapshot This example creates a new cluster snapshot. By default, the output is in JSON format.

Command:

```
aws redshift create-cluster-snapshot --cluster-identifier mycluster --snapshot-
identifier my-snapshot-id
```

Result:

```
{
  "Snapshot": {
    "Status": "creating",
    "SnapshotCreateTime": "2013-01-22T22:20:33.548Z",
    "AvailabilityZone": "us-east-1a",
    "ClusterVersion": "1.0",
    "MasterUsername": "adminuser",
    "DBName": "dev",
    "ClusterCreateTime": "2013-01-22T21:59:29.559Z",
    "SnapshotType": "manual",
    "NodeType": "dw.hs1.xlarge",
    "ClusterIdentifier": "mycluster",
    "Port": 5439,
    "NumberOfNodes": "2",
    "SnapshotIdentifier": "my-snapshot-id"
  }
}
```

```

    },
    "ResponseMetadata": {
      "RequestId": "f024d1a5-64e1-11e2-88c5-53eb05787dfb"
    }
  }
}

```

- For API details, see [CreateClusterSnapshot](#) in *AWS CLI Command Reference*.

create-cluster-subnet-group

The following code example shows how to use `create-cluster-subnet-group`.

AWS CLI

Create a Cluster Subnet Group This example creates a new cluster subnet group. Command:

```
aws redshift create-cluster-subnet-group --cluster-subnet-group-name mysubnetgroup
--description "My subnet group" --subnet-ids subnet-763fdd1c
```

Result:

```

{
  "ClusterSubnetGroup": {
    "Subnets": [
      {
        "SubnetStatus": "Active",
        "SubnetIdentifier": "subnet-763fdd1c",
        "SubnetAvailabilityZone": {
          "Name": "us-east-1a"
        }
      }
    ],
    "VpcId": "vpc-7e3fdd14",
    "SubnetGroupStatus": "Complete",
    "Description": "My subnet group",
    "ClusterSubnetGroupName": "mysubnetgroup"
  },
  "ResponseMetadata": {
    "RequestId": "500b8ce2-698f-11e2-9790-fd67517fb6fd"
  }
}

```

- For API details, see [CreateClusterSubnetGroup](#) in *AWS CLI Command Reference*.

create-cluster

The following code example shows how to use `create-cluster`.

AWS CLI

Create a Cluster with Minimal Parameters This example creates a cluster with the minimal set of parameters. By default, the output is in JSON format. Command:

```
aws redshift create-cluster --node-type dw.hs1.xlarge --number-of-nodes 2 --master-username adminuser --master-user-password TopSecret1 --cluster-identifier mycluster
```

Result:

```
{
  "Cluster": {
    "NodeType": "dw.hs1.xlarge",
    "ClusterVersion": "1.0",
    "PubliclyAccessible": "true",
    "MasterUsername": "adminuser",
    "ClusterParameterGroups": [
      {
        "ParameterApplyStatus": "in-sync",
        "ParameterGroupName": "default.redshift-1.0"
      }
    ],
    "ClusterSecurityGroups": [
      {
        "Status": "active",
        "ClusterSecurityGroupName": "default"
      }
    ],
    "AllowVersionUpgrade": true,
    "VpcSecurityGroups": [],
    "PreferredMaintenanceWindow": "sat:03:30-sat:04:00",
    "AutomatedSnapshotRetentionPeriod": 1,
    "ClusterStatus": "creating",
    "ClusterIdentifier": "mycluster",
    "DBName": "dev",
    "NumberOfNodes": 2,
    "PendingModifiedValues": {
      "MasterUserPassword": "\*****"
    }
  },
  "ResponseMetadata": {
```



```

    "RequestId": "7cf4bcfc-64dd-11e2-bea9-49e0ce183f07"
  }
}

```

- For API details, see [CreateCluster](#) in *AWS CLI Command Reference*.

create-event-subscription

The following code example shows how to use `create-event-subscription`.

AWS CLI

To create a notification subscription for an event

The following `create-event-subscription` example creates an event notification subscription.

```

aws redshift create-event-subscription \
  --subscription-name mysubscription \
  --sns-topic-arn arn:aws:sns:us-west-2:123456789012:MySNSStopic \
  --source-type cluster \
  --source-ids mycluster

```

Output:

```

{
  "EventSubscription": {
    "CustomerAwsId": "123456789012",
    "CustSubscriptionId": "mysubscription",
    "SnsTopicArn": "arn:aws:sns:us-west-2:123456789012:MySNSStopic",
    "Status": "active",
    "SubscriptionCreationTime": "2019-12-09T20:05:19.365Z",
    "SourceType": "cluster",
    "SourceIdsList": [
      "mycluster"
    ],
    "EventCategoriesList": [],
    "Severity": "INFO",
    "Enabled": true,
    "Tags": []
  }
}

```

For more information, see [Subscribing to Amazon Redshift Event Notifications](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [CreateEventSubscription](#) in *AWS CLI Command Reference*.

create-hsm-client-certificate

The following code example shows how to use `create-hsm-client-certificate`.

AWS CLI

To create an HSM client certificate

The following `create-hsm-client-certificate` example generates an HSM client certificate that a cluster can use to connect to an HSM.

```
aws redshift create-hsm-client-certificate \
  --hsm-client-certificate-identifier myhsmclientcert
```

Output:

```
{
  "HsmClientCertificate": {
    "HsmClientCertificateIdentifier": "myhsmclientcert",
    "HsmClientCertificatePublicKey": "-----BEGIN CERTIFICATE-----
MIICiEXAMPLECQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMakGA1UEBhMC
VVMxCzAJBgNVBAGTEXAMPLEEwDgYDVQQHEwDTZWf0dGx1MQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBA5TC01BTSBDb25EXAMPLEIwEAYDVQQDEw1UZjN0Q21sYWxhZAd
BgkqhkiG9w0BCQEWEG5vb251QGftYXpvbi5jb2EXAMPLETEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMakGA1UEBhMCVVMxCzAJBgNVBEXAMPLEMRAwDgYD
EXAMPLETZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBA5TC01BTSBDb25z
b2x1MRIwEAEXAMPLEEw1UZjN0Q21sYWxhZAdBgkqhkiG9w0BCQEWEG5vb251QGft
YXpvbi5jb20wgZ8wDQYJKEXAMPLEAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySwTC2XADZ4nB+BLYgVIk6EXAMPLE3G93vUEI03IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugEXAMPLEzZswY6786m86gpE
Ibb30hjZnzcvcQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEEEXAMPLEEAtCu4
nUHVvxYUEXAMPLEh8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFbjvSfpJI1J00zbhNYS5f6GEXAMPLE10ZxBHjJnyp3780D8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rEXAMPLE=-----END CERTIFICATE-----\n",
    "Tags": []
  }
}
```

For more information, see [Amazon Redshift API Permissions Reference](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [CreateHsmClientCertificate](#) in *AWS CLI Command Reference*.

create-hsm-configuration

The following code example shows how to use `create-hsm-configuration`.

AWS CLI

To create an HSM configuration

The following `create-hsm-configuration` example creates the specified HSM configuration that contains information required by a cluster to store and use database encryption keys in a hardware security module (HSM).

```
aws redshift create-hsm-configuration /
  --hsm-configuration-identifier myhsmconnection
  --description "My HSM connection"
  --hsm-ip-address 192.0.2.09
  --hsm-partition-name myhsmpartition /
  --hsm-partition-password A1b2c3d4 /
  --hsm-server-public-certificate myhsmclientcert
```

Output:

```
{
  "HsmConfiguration": {
    "HsmConfigurationIdentifier": "myhsmconnection",
    "Description": "My HSM connection",
    "HsmIpAddress": "192.0.2.09",
    "HsmPartitionName": "myhsmpartition",
    "Tags": []
  }
}
```

- For API details, see [CreateHsmConfiguration](#) in *AWS CLI Command Reference*.

create-snapshot-copy-grant

The following code example shows how to use `create-snapshot-copy-grant`.

AWS CLI

To create a snapshot copy grant

The following `create-snapshot-copy-grant` example creates a snapshot copy grant and encrypts copied snapshots in a destination AWS Region.

```
aws redshift create-snapshot-copy-grant \  
  --snapshot-copy-grant-name mysnapshotcopygrantname
```

Output:

```
{  
  "SnapshotCopyGrant": {  
    "SnapshotCopyGrantName": "mysnapshotcopygrantname",  
    "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/  
bPxRfih3yCo8nvbEXAMPLEKEY",  
    "Tags": []  
  }  
}
```

For more information, see [Amazon Redshift Database Encryption](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [CreateSnapshotCopyGrant](#) in *AWS CLI Command Reference*.

create-snapshot-schedule

The following code example shows how to use `create-snapshot-schedule`.

AWS CLI

To create snapshot schedule

The following `create-snapshot-schedule` example creates a snapshot schedule with the specified description and a rate of every 12 hours.

```
aws redshift create-snapshot-schedule \  
  --schedule-definitions "rate(12 hours)" \  
  --schedule-identifier mysnapshotschedule \  
  --schedule-description "My schedule description"
```

Output:

```
{
  "ScheduleDefinitions": [
    "rate(12 hours)"
  ],
  "ScheduleIdentifier": "mysnapshotschedule",
  "ScheduleDescription": "My schedule description",
  "Tags": []
}
```

For more information, see [Automated Snapshot Schedules](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [CreateSnapshotSchedule](#) in *AWS CLI Command Reference*.

create-tags

The following code example shows how to use `create-tags`.

AWS CLI**To create tags for a cluster**

The following `create-tags` example adds the specified tag key/value pair to the specified cluster.

```
aws redshift create-tags \
  --resource-name arn:aws:redshift:us-west-2:123456789012:cluster:mycluster \
  --tags "Key"="mytags", "Value"="tag1"
```

This command does not produce any output.

For more information, see [Tagging Resources in Amazon Redshift](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [CreateTags](#) in *AWS CLI Command Reference*.

delete-cluster-parameter-group

The following code example shows how to use `delete-cluster-parameter-group`.

AWS CLI

Delete a Cluster Parameter Group This example deletes a cluster parameter group. Command:

```
aws redshift delete-cluster-parameter-group --parameter-group-name
myclusterparametergroup
```

- For API details, see [DeleteClusterParameterGroup](#) in *AWS CLI Command Reference*.

delete-cluster-security-group

The following code example shows how to use `delete-cluster-security-group`.

AWS CLI

Delete a Cluster Security Group This example deletes a cluster security group. Command:

```
aws redshift delete-cluster-security-group --cluster-security-group-name
mysecuritygroup
```

- For API details, see [DeleteClusterSecurityGroup](#) in *AWS CLI Command Reference*.

delete-cluster-snapshot

The following code example shows how to use `delete-cluster-snapshot`.

AWS CLI

Delete a Cluster Snapshot This example deletes a cluster snapshot. Command:

```
aws redshift delete-cluster-snapshot --snapshot-identifier my-snapshot-id
```

- For API details, see [DeleteClusterSnapshot](#) in *AWS CLI Command Reference*.

delete-cluster-subnet-group

The following code example shows how to use `delete-cluster-subnet-group`.

AWS CLI

Delete a Cluster subnet Group This example deletes a cluster subnet group. Command:

```
aws redshift delete-cluster-subnet-group --cluster-subnet-group-name mysubnetgroup
```

Result:

```
{
  "ResponseMetadata": {
    "RequestId": "253fbffd-6993-11e2-bc3a-47431073908a"
  }
}
```

- For API details, see [DeleteClusterSubnetGroup](#) in *AWS CLI Command Reference*.

delete-cluster

The following code example shows how to use `delete-cluster`.

AWS CLI

Delete a Cluster with No Final Cluster SnapshotThis example deletes a cluster, forcing data deletion so no final cluster snapshot is created.**Command:**

```
aws redshift delete-cluster --cluster-identifier mycluster --skip-final-cluster-snapshot
```

Delete a Cluster, Allowing a Final Cluster SnapshotThis example deletes a cluster, but specifies a final cluster snapshot.**Command:**

```
aws redshift delete-cluster --cluster-identifier mycluster --final-cluster-snapshot-identifier myfinalsnapshot
```

- For API details, see [DeleteCluster](#) in *AWS CLI Command Reference*.

delete-event-subscription

The following code example shows how to use `delete-event-subscription`.

AWS CLI

To delete event subscription

The following `delete-event-subscription` example deletes the specified event notification subscription.

```
aws redshift delete-event-subscription \  
  --subscription-name mysubscription
```

This command does not produce any output.

For more information, see [Subscribing to Amazon Redshift Event Notifications](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [DeleteEventSubscription](#) in *AWS CLI Command Reference*.

delete-hsm-client-certificate

The following code example shows how to use `delete-hsm-client-certificate`.

AWS CLI

To delete HSM client certificate

The following `delete-hsm-client-certificate` example deletes an HSM client certificate.

```
aws redshift delete-hsm-client-certificate \  
  --hsm-client-certificate-identifier myhsmclientcert
```

This command does not produce any output.

For more information, see [Amazon Redshift API Permissions Reference](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [DeleteHsmClientCertificate](#) in *AWS CLI Command Reference*.

delete-hsm-configuration

The following code example shows how to use `delete-hsm-configuration`.

AWS CLI

To delete an HSM configuration

The following `delete-hsm-configuration` example deletes the specified HSM configuration from the current AWS account.

```
aws redshift delete-hsm-configuration /  
  --hsm-configuration-identifier myhsmconnection
```

This command does not produce any output.

- For API details, see [DeleteHsmConfiguration](#) in *AWS CLI Command Reference*.

delete-scheduled-action

The following code example shows how to use `delete-scheduled-action`.

AWS CLI

To delete scheduled action

The following `delete-scheduled-action` example deletes the specified scheduled action.

```
aws redshift delete-scheduled-action \  
  --scheduled-action-name myscheduledaction
```

This command does not produce any output.

- For API details, see [DeleteScheduledAction](#) in *AWS CLI Command Reference*.

delete-snapshot-copy-grant

The following code example shows how to use `delete-snapshot-copy-grant`.

AWS CLI

To delete snapshot copy grant

The following `delete-snapshot-copy-grant` example deletes the specified snapshot copy grant.

```
aws redshift delete-snapshot-copy-grant \  
  --snapshot-copy-grant-name mysnapshotcopygrantname
```

This command does not produce any output.

For more information, see [Amazon Redshift Database Encryption](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [DeleteSnapshotCopyGrant](#) in *AWS CLI Command Reference*.

delete-snapshot-schedule

The following code example shows how to use `delete-snapshot-schedule`.

AWS CLI

To delete snapshot schedule

The following `delete-snapshot-schedule` example deletes the specified snapshot schedule. You must disassociate clusters before deleting the schedule.

```
aws redshift delete-snapshot-schedule \  
  --schedule-identifier mysnapshotschedule
```

This command does not produce any output.

For more information, see [Automated Snapshot Schedules](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [DeleteSnapshotSchedule](#) in *AWS CLI Command Reference*.

delete-tags

The following code example shows how to use `delete-tags`.

AWS CLI

To delete tags from a cluster

The following `delete-tags` example deletes the tags with the specified key names from the specified cluster.

```
aws redshift delete-tags \  
  --resource-name arn:aws:redshift:us-west-2:123456789012:cluster:mycluster \  
  --tag-keys "clustertagkey" "clustertagvalue"
```

This command does not produce any output.

For more information, see [Tagging Resources in Amazon Redshift](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [DeleteTags](#) in *AWS CLI Command Reference*.

describe-account-attributes

The following code example shows how to use `describe-account-attributes`.

AWS CLI

To describe attributes of an AWS account

The following `describe-account-attributes` example displays the attributes attached to the calling AWS account.

```
aws redshift describe-account-attributes
```

Output:

```
{
  "AccountAttributes": [
    {
      "AttributeName": "max-defer-maintenance-duration",
      "AttributeValues": [
        {
          "AttributeValue": "45"
        }
      ]
    }
  ]
}
```

- For API details, see [DescribeAccountAttributes](#) in *AWS CLI Command Reference*.

describe-cluster-db-revisions

The following code example shows how to use `describe-cluster-db-revisions`.

AWS CLI

To describe DB revisions for a cluster

The following `describe-cluster-db-revisions` example displays the details of an array of `ClusterDbRevision` objects for the specified cluster.

```
aws redshift describe-cluster-db-revisions \  
  --cluster-identifier mycluster
```

Output:

```
{  
  "ClusterDbRevisions": [  
    {  
      "ClusterIdentifier": "mycluster",  
      "CurrentDatabaseRevision": "11420",  
      "DatabaseRevisionReleaseDate": "2019-11-22T16:43:49.597Z",  
      "RevisionTargets": []  
    }  
  ]  
}
```

- For API details, see [DescribeClusterDbRevisions](#) in *AWS CLI Command Reference*.

`describe-cluster-parameter-groups`

The following code example shows how to use `describe-cluster-parameter-groups`.

AWS CLI

Get a Description of All Cluster Parameter Groups This example returns a description of all cluster parameter groups for the account, with column headers. By default, the output is in JSON format.
Command:

```
aws redshift describe-cluster-parameter-groups
```

Result:

```
{  
  "ParameterGroups": [  
    {  
      "ParameterGroupFamily": "redshift-1.0",  
      "Description": "My first cluster parameter group",  
      "ParameterGroupIdentifier": "pg1",  
      "ParameterGroupStatus": "available",  
      "ParameterGroupType": "default",  
      "CreationTime": "2019-11-22T16:43:49.597Z",  
      "DeletionProtection": false,  
      "AllowMajorVersionUpgrade": true,  
      "AllowMinorVersionUpgrade": true,  
      "AssociatedRoles": [],  
      "ClusterDbRevisions": []  
    }  
  ]  
}
```

```
    "ParameterGroupName": "myclusterparametergroup"
  } ],
  "ResponseMetadata": {
    "RequestId": "8ceb8f6f-64cc-11e2-bea9-49e0ce183f07"
  }
}
```

You can also obtain the same information in text format using the `--output text` option.Command:

`--output text` option.Command:

option.Command:

```
aws redshift describe-cluster-parameter-groups --output text
```

Result:

```
redshift-1.0      My first cluster parameter group      myclusterparametergroup
RESPONSEMETADATA 9e665a36-64cc-11e2-8f7d-3b939af52818
```

- For API details, see [DescribeClusterParameterGroups](#) in *AWS CLI Command Reference*.

describe-cluster-parameters

The following code example shows how to use `describe-cluster-parameters`.

AWS CLI

Retrieve the Parameters for a Specified Cluster Parameter GroupThis example retrieves the parameters for the named parameter group. By default, the output is in JSON format.Command:

```
aws redshift describe-cluster-parameters --parameter-group-name
myclusterparametergroup
```

Result:

```
{
  "Parameters": [
```

```

    {
      "Description": "Sets the display format for date and time values.",
      "DataType": "string",
      "IsModifiable": true,
      "Source": "engine-default",
      "ParameterValue": "ISO, MDY",
      "ParameterName": "datestyle"
    },
    {
      "Description": "Sets the number of digits displayed for floating-point
values",
      "DataType": "integer",
      "IsModifiable": true,
      "AllowedValues": "-15-2",
      "Source": "engine-default",
      "ParameterValue": "0",
      "ParameterName": "extra_float_digits"
    },
    (...remaining output omitted...)
  ]
}

```

You can also obtain the same information in text format using the `--output text` option.Command:

`--output text` option.Command:

option.Command:

```
aws redshift describe-cluster-parameters --parameter-group-name
myclusterparametergroup --output text
```

Result:

```

RESPONSEMETADATA    cdac40aa-64cc-11e2-9e70-918437dd236d
Sets the display format for date and time values.  string True  engine-default
ISO, MDY      datestyle
Sets the number of digits displayed for floating-point values  integer True
-15-2  engine-default  0      extra_float_digits
This parameter applies a user-defined label to a group of queries that are run
during the same session..  string True  engine-default  default query_group
require ssl for all databaseconnections  boolean True  true,false  engine-
default  false  require_ssl

```

```

Sets the schema search order for names that are not schema-qualified.      string
True    engine-default $user, public  search_path
Aborts any statement that takes over the specified number of milliseconds. integer
True    engine-default 0              statement_timeout
wlm json configuration              string True    engine-default
\["query_concurrency":5}]          wlm_json_configuration

```

- For API details, see [DescribeClusterParameters](#) in *AWS CLI Command Reference*.

describe-cluster-security-groups

The following code example shows how to use `describe-cluster-security-groups`.

AWS CLI

Get a Description of All Cluster Security Groups This example returns a description of all cluster security groups for the account. By default, the output is in JSON format. Command:

```
aws redshift describe-cluster-security-groups
```

Result:

```

{
  "ClusterSecurityGroups": [
    {
      "OwnerId": "100447751468",
      "Description": "default",
      "ClusterSecurityGroupName": "default",
      "EC2SecurityGroups": [],
      "IPRanges": [
        {
          "Status": "authorized",
          "CIDRIP": "0.0.0.0/0"
        }
      ]
    },
    {
      "OwnerId": "100447751468",
      "Description": "This is my cluster security group",
      "ClusterSecurityGroupName": "mysecuritygroup",
      "EC2SecurityGroups": [],
      "IPRanges": []
    }
  ]
}

```

```

    },
    (...remaining output omitted...)
  ]
}

```

- For API details, see [DescribeClusterSecurityGroups](#) in *AWS CLI Command Reference*.

describe-cluster-snapshots

The following code example shows how to use `describe-cluster-snapshots`.

AWS CLI

Get a Description of All Cluster Snapshots This example returns a description of all cluster snapshots for the account. By default, the output is in JSON format. Command:

```
aws redshift describe-cluster-snapshots
```

Result:

```

{
  "Snapshots": [
    {
      "Status": "available",
      "SnapshotCreateTime": "2013-07-17T22:02:22.852Z",
      "EstimatedSecondsToCompletion": -1,
      "AvailabilityZone": "us-east-1a",
      "ClusterVersion": "1.0",
      "MasterUsername": "adminuser",
      "Encrypted": false,
      "OwnerAccount": "111122223333",
      "BackupProgressInMegabytes": 20.0,
      "ElapsedTimeInSeconds": 0,
      "DBName": "dev",
      "CurrentBackupRateInMegabytesPerSecond": 0.0,
      "ClusterCreateTime": "2013-01-22T21:59:29.559Z",
      "ActualIncrementalBackupSizeInMegabytes": 20.0,
      "SnapshotType": "automated",
      "NodeType": "dw.hs1.xlarge",
      "ClusterIdentifier": "mycluster",
      "Port": 5439,
      "TotalBackupSizeInMegabytes": 20.0,
    }
  ]
}

```



```

    "NumberOfNodes": "2",
    "SnapshotIdentifier": "cm:mycluster-2013-01-22-22-04-18"
  },
  {
    "EstimatedSecondsToCompletion": 0,
    "OwnerAccount": "111122223333",
    "CurrentBackupRateInMegabytesPerSecond": 0.1534,
    "ActualIncrementalBackupSizeInMegabytes": 11.0,
    "NumberOfNodes": "2",
    "Status": "available",
    "ClusterVersion": "1.0",
    "MasterUsername": "adminuser",
    "AccountsWithRestoreAccess": [
      {
        "AccountID": "444455556666"
      }
    ],
    "TotalBackupSizeInMegabytes": 20.0,
    "DBName": "dev",
    "BackupProgressInMegabytes": 11.0,
    "ClusterCreateTime": "2013-01-22T21:59:29.559Z",
    "ElapsedTimeInSeconds": 0,
    "ClusterIdentifier": "mycluster",
    "SnapshotCreateTime": "2013-07-17T22:04:18.947Z",
    "AvailabilityZone": "us-east-1a",
    "NodeType": "dw.hs1.xlarge",
    "Encrypted": false,
    "SnapshotType": "manual",
    "Port": 5439,
    "SnapshotIdentifier": "my-snapshot-id"
  }
]
}
(...remaining output omitted...)

```

- For API details, see [DescribeClusterSnapshots](#) in *AWS CLI Command Reference*.

describe-cluster-subnet-groups

The following code example shows how to use `describe-cluster-subnet-groups`.

AWS CLI

Get a Description of All Cluster Subnet Groups This example returns a description of all cluster subnet groups. By default, the output is in JSON format. Command:

```
aws redshift describe-cluster-subnet-groups
```

Result:

```
{
  "ClusterSubnetGroups": [
    {
      "Subnets": [
        {
          "SubnetStatus": "Active",
          "SubnetIdentifier": "subnet-763fdd1c",
          "SubnetAvailabilityZone": {
            "Name": "us-east-1a"
          }
        }
      ],
      "VpcId": "vpc-7e3fdd14",
      "SubnetGroupStatus": "Complete",
      "Description": "My subnet group",
      "ClusterSubnetGroupName": "mysubnetgroup"
    }
  ],
  "ResponseMetadata": {
    "RequestId": "37fa8c89-6990-11e2-8f75-ab4018764c77"
  }
}
```

- For API details, see [DescribeClusterSubnetGroups](#) in *AWS CLI Command Reference*.

describe-cluster-tracks

The following code example shows how to use `describe-cluster-tracks`.

AWS CLI**To describe cluster tracks**

The following `describe-cluster-tracks` example displays details of the available maintenance tracks.

```
aws redshift describe-cluster-tracks \
```

```
--maintenance-track-name current
```

Output:

```
{
  "MaintenanceTracks": [
    {
      "MaintenanceTrackName": "current",
      "DatabaseVersion": "1.0.11420",
      "UpdateTargets": [
        {
          "MaintenanceTrackName": "preview_features",
          "DatabaseVersion": "1.0.11746",
          "SupportedOperations": [
            {
              "OperationName": "restore-from-cluster-snapshot"
            }
          ]
        },
        {
          "MaintenanceTrackName": "trailing",
          "DatabaseVersion": "1.0.11116",
          "SupportedOperations": [
            {
              "OperationName": "restore-from-cluster-snapshot"
            },
            {
              "OperationName": "modify-cluster"
            }
          ]
        }
      ]
    }
  ]
}
```

For more information, see [Choosing Cluster Maintenance Tracks](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [DescribeClusterTracks](#) in *AWS CLI Command Reference*.

describe-cluster-versions

The following code example shows how to use `describe-cluster-versions`.

AWS CLI

Get a Description of All Cluster VersionsThis example returns a description of all cluster versions. By default, the output is in JSON format.**Command:**

```
aws redshift describe-cluster-versions
```

Result:

```
{
  "ClusterVersions": [
    {
      "ClusterVersion": "1.0",
      "Description": "Initial release",
      "ClusterParameterGroupFamily": "redshift-1.0"
    } ],
  "ResponseMetadata": {
    "RequestId": "16a53de3-64cc-11e2-bec0-17624ad140dd"
  }
}
```

- For API details, see [DescribeClusterVersions](#) in *AWS CLI Command Reference*.

describe-clusters

The following code example shows how to use `describe-clusters`.

AWS CLI

Get a Description of All ClustersThis example returns a description of all clusters for the account. By default, the output is in JSON format.**Command:**

```
aws redshift describe-clusters
```

Result:

```
{
```

```
"Clusters": [
  {
    "NodeType": "dw.hs1.xlarge",
    "Endpoint": {
      "Port": 5439,
      "Address": "mycluster.coqoarplqhsn.us-east-1.redshift.amazonaws.com"
    },
    "ClusterVersion": "1.0",
    "PubliclyAccessible": "true",
    "MasterUsername": "adminuser",
    "ClusterParameterGroups": [
      {
        "ParameterApplyStatus": "in-sync",
        "ParameterGroupName": "default.redshift-1.0"
      }
    ],
    "ClusterSecurityGroups": [
      {
        "Status": "active",
        "ClusterSecurityGroupName": "default"
      }
    ],
    "AllowVersionUpgrade": true,
    "VpcSecurityGroups": [],
    "AvailabilityZone": "us-east-1a",
    "ClusterCreateTime": "2013-01-22T21:59:29.559Z",
    "PreferredMaintenanceWindow": "sat:03:30-sat:04:00",
    "AutomatedSnapshotRetentionPeriod": 1,
    "ClusterStatus": "available",
    "ClusterIdentifier": "mycluster",
    "DBName": "dev",
    "NumberOfNodes": 2,
    "PendingModifiedValues": {}
  }
],
"ResponseMetadata": {
  "RequestId": "65b71cac-64df-11e2-8f5b-e90bd6c77476"
}
}
```

You can also obtain the same information in text format using the `--output text` option.Command:

`--output text option.Command:`

`option.Command:`

```
aws redshift describe-clusters --output text
```

Result:

```
dw.hs1.xlarge      1.0      true      adminuser      True      us-east-1a
2013-01-22T21:59:29.559Z      sat:03:30-sat:04:00      1      available
mycluster      dev      2
ENDPOINT      5439      mycluster.coqoarplqhsn.us-east-1.redshift.amazonaws.com
in-sync      default.redshift-1.0
active      default
PENDINGMODIFIEDVALUES
RESPONSEMETADATA      934281a8-64df-11e2-b07c-f7fbdd006c67
```

- For API details, see [DescribeClusters](#) in *AWS CLI Command Reference*.

describe-default-cluster-parameters

The following code example shows how to use describe-default-cluster-parameters.

AWS CLI

Get a Description of Default Cluster Parameters This example returns a description of the default cluster parameters for the redshift-1.0 family. By default, the output is in JSON format. Command:

```
aws redshift describe-default-cluster-parameters --parameter-group-family
redshift-1.0
```

Result:

```
{
  "DefaultClusterParameters": {
    "ParameterGroupFamily": "redshift-1.0",
    "Parameters": [
      {
        "Description": "Sets the display format for date and time values.",
        "DataType": "string",
        "IsModifiable": true,
        "Source": "engine-default",
        "ParameterValue": "ISO, MDY",
        "ParameterName": "datestyle"
      }
    ]
  }
}
```

```
    },
    {
      "Description": "Sets the number of digits displayed for floating-point
values",
      "DataType": "integer",
      "IsModifiable": true,
      "AllowedValues": "-15-2",
      "Source": "engine-default",
      "ParameterValue": "0",
      "ParameterName": "extra_float_digits"
    },
    (...remaining output omitted...)
  ]
}
```

To see a list of valid parameter group families, use the `describe-cluster-parameter-groups` command.

`describe-cluster-parameter-groups` command.

command.

- For API details, see [DescribeDefaultClusterParameters](#) in *AWS CLI Command Reference*.

describe-event-categories

The following code example shows how to use `describe-event-categories`.

AWS CLI

To describe event categories for a cluster

The following `describe-event-categories` example displays details for the event categories for a cluster.

```
aws redshift describe-event-categories \
  --source-type cluster
```

Output:

```
{
```

```

    "EventCategoriesMapList": [
      {
        "SourceType": "cluster",
        "Events": [
          {
            "EventId": "REDSHIFT-EVENT-2000",
            "EventCategories": [
              "management"
            ],
            "EventDescription": "Cluster <cluster name> created at <time in
UTC>.",
            "Severity": "INFO"
          },
          {
            "EventId": "REDSHIFT-EVENT-2001",
            "EventCategories": [
              "management"
            ],
            "EventDescription": "Cluster <cluster name> deleted at <time in
UTC>.",
            "Severity": "INFO"
          },
          {
            "EventId": "REDSHIFT-EVENT-3625",
            "EventCategories": [
              "monitoring"
            ],
            "EventDescription": "The cluster <cluster name> can't be resumed
with its previous elastic network interface <ENI id>. We will allocate a new
elastic network interface and associate it with the cluster node.",
            "Severity": "INFO"
          }
        ]
      }
    ]
  }
}

```

- For API details, see [DescribeEventCategories](#) in *AWS CLI Command Reference*.

describe-event-subscriptions

The following code example shows how to use describe-event-subscriptions.

AWS CLI

To describe event subscriptions

The following `describe-event-subscriptions` example displays event notification subscriptions for the specified subscription.

```
aws redshift describe-event-subscriptions \  
  --subscription-name mysubscription
```

Output:

```
{  
  "EventSubscriptionsList": [  
    {  
      "CustomerAwsId": "123456789012",  
      "CustSubscriptionId": "mysubscription",  
      "SnsTopicArn": "arn:aws:sns:us-west-2:123456789012:MySNSStopic",  
      "Status": "active",  
      "SubscriptionCreationTime": "2019-12-09T21:50:21.332Z",  
      "SourceIdsList": [],  
      "EventCategoriesList": [  
        "management"  
      ],  
      "Severity": "ERROR",  
      "Enabled": true,  
      "Tags": []  
    }  
  ]  
}
```

For more information, see [Subscribing to Amazon Redshift Event Notifications](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [DescribeEventSubscriptions](#) in *AWS CLI Command Reference*.

`describe-events`

The following code example shows how to use `describe-events`.

AWS CLI

Describe All Events this example returns all events. By default, the output is in JSON format. Command:

```
aws redshift describe-events
```

Result:

```
{
  "Events": [
    {
      "Date": "2013-01-22T19:17:03.640Z",
      "SourceIdentifier": "myclusterparametergroup",
      "Message": "Cluster parameter group myclusterparametergroup has been
created.",
      "SourceType": "cluster-parameter-group"
    } ],
  "ResponseMetadata": {
    "RequestId": "9f056111-64c9-11e2-9390-ff04f2c1e638"
  }
}
```

You can also obtain the same information in text format using the `--output text` option. Command:

`--output text` option. Command:

option. Command:

```
aws redshift describe-events --output text
```

Result:

```
2013-01-22T19:17:03.640Z    myclusterparametergroup Cluster parameter group
myclusterparametergroup has been created.    cluster-parameter-group
RESPONSEMETADATA    8e5fe765-64c9-11e2-bce3-e56f52c50e17
```

- For API details, see [DescribeEvents](#) in *AWS CLI Command Reference*.

describe-hsm-client-certificates

The following code example shows how to use `describe-hsm-client-certificates`.

AWS CLI

To describe HSM client certificates

The following `describe-hsm-client-certificates` example displays details for the specified HSM client certificate.

```
aws redshift describe-hsm-client-certificates \
  --hsm-client-certificate-identifier myhsmclientcert
```

Output:

```
{
  "HsmClientCertificates": [
    {
      "HsmClientCertificateIdentifier": "myhsmclientcert",
      "HsmClientCertificatePublicKey": "-----BEGIN CERTIFICATE-----\
EXAMPLECAfICCQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAEXAMPLERAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAAsTC01BTSBDb25zEXAMPLEwEAYDVQQDEwLUZXN0Q21sYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb25lQGftYXpvbi5jb20wHhEXAMPLEDI1MjA0EXAMPLEN
EXAMPLE0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
VQQHEwdTZWF0dGEXAMPLEQYDVQQKEwZBbWF6b24xFDASBgNVBAAsTC01BTSBDb25z
b2x1MRIwEAYDVQQDEwLUZXN0Q21sEXAMPLEdBgkqhkiG9w0BCQEWEG5vb25lQGft
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIEXAMPLEMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLygVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY67EXAMPLEE
EXAMPLEZnzcvcQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9EXAMPLE6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJI1J00zbhNYS5f6GuoEEXAMPLEBHjJnyp3780D8uTs7fLvJx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rEXAMPLE=-----END CERTIFICATE-----\n",
      "Tags": []
    }
  ]
}
```

For more information, see [Amazon Redshift API Permissions Reference](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [DescribeHsmClientCertificates](#) in *AWS CLI Command Reference*.

describe-hsm-configurations

The following code example shows how to use describe-hsm-configurations.

AWS CLI

To describe HSM configurations

The following describe-hsm-configurations example displays details for the available HSM configurations for the calling AWS account.

```
aws redshift describe-hsm-configurations /  
  --hsm-configuration-identifier myhsmconnection
```

Output:

```
{  
  "HsmConfigurations": [  
    {  
      "HsmConfigurationIdentifier": "myhsmconnection",  
      "Description": "My HSM connection",  
      "HsmIpAddress": "192.0.2.09",  
      "HsmPartitionName": "myhsmpartition",  
      "Tags": []  
    }  
  ]  
}
```

- For API details, see [DescribeHsmConfigurations](#) in *AWS CLI Command Reference*.

describe-logging-status

The following code example shows how to use describe-logging-status.

AWS CLI

To describe logging status for a cluster

The following describe-logging-status example displays whether information, such as queries and connection attempts, is being logged for a cluster.

```
aws redshift describe-logging-status \  
  --cluster-identifier mycluster
```

Output:

```
{  
  "LoggingEnabled": false  
}
```

For more information, see [Database Audit Logging](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [DescribeLoggingStatus](#) in *AWS CLI Command Reference*.

describe-node-configuration-options

The following code example shows how to use `describe-node-configuration-options`.

AWS CLI

To describe node configuration options

The following `describe-node-configuration-options` example displays the properties of possible node configurations such as node type, number of nodes, and disk usage for the specified cluster snapshot.

```
aws redshift describe-node-configuration-options \  
  --action-type restore-cluster \  
  --snapshot-identifier rs:mycluster-2019-12-09-16-42-43
```

Output:

```
{  
  "NodeConfigurationOptionList": [  
    {  
      "NodeType": "dc2.large",  
      "NumberOfNodes": 2,  
      "EstimatedDiskUtilizationPercent": 19.61  
    },  
    {  
      "NodeType": "dc2.large",
```

```

        "NumberOfNodes": 4,
        "EstimatedDiskUtilizationPercent": 9.96
    },
    {
        "NodeType": "ds2.xlarge",
        "NumberOfNodes": 2,
        "EstimatedDiskUtilizationPercent": 1.53
    },
    {
        "NodeType": "ds2.xlarge",
        "NumberOfNodes": 4,
        "EstimatedDiskUtilizationPercent": 0.78
    }
]
}

```

For more information, see [Purchasing Amazon Redshift Reserved Nodes](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [DescribeNodeConfigurationOptions](#) in *AWS CLI Command Reference*.

describe-orderable-cluster-options

The following code example shows how to use `describe-orderable-cluster-options`.

AWS CLI

Describing All Orderable Cluster Options This example returns descriptions of all orderable cluster options. By default, the output is in JSON format. **Command:**

```
aws redshift describe-orderable-cluster-options
```

Result:

```

{
  "OrderableClusterOptions": [
    {
      "NodeType": "dw.hs1.8xlarge",
      "AvailabilityZones": [
        { "Name": "us-east-1a" },
        { "Name": "us-east-1b" },
        { "Name": "us-east-1c" } ],
    }
  ]
}

```

```

    "ClusterVersion": "1.0",
    "ClusterType": "multi-node"
  },
  {
    "NodeType": "dw.hs1.xlarge",
    "AvailabilityZones": [
      { "Name": "us-east-1a" },
      { "Name": "us-east-1b" },
      { "Name": "us-east-1c" } ],
    "ClusterVersion": "1.0",
    "ClusterType": "multi-node"
  },
  {
    "NodeType": "dw.hs1.xlarge",
    "AvailabilityZones": [
      { "Name": "us-east-1a" },
      { "Name": "us-east-1b" },
      { "Name": "us-east-1c" } ],
    "ClusterVersion": "1.0",
    "ClusterType": "single-node"
  } ],
  "ResponseMetadata": {
    "RequestId": "f6000035-64cb-11e2-9135-ff82df53a51a"
  }
}

```

You can also obtain the same information in text format using the `--output text` option.Command:

`--output text` option.Command:

option.Command:

```
aws redshift describe-orderable-cluster-options --output text
```

Result:

```

dw.hs1.8xlarge      1.0      multi-node
us-east-1a
us-east-1b
us-east-1c
dw.hs1.xlarge      1.0      multi-node
us-east-1a

```

```

us-east-1b
us-east-1c
dw.hs1.xlarge      1.0      single-node
us-east-1a
us-east-1b
us-east-1c
RESPONSEMETADATA  e648696b-64cb-11e2-bec0-17624ad140dd

```

- For API details, see [DescribeOrderableClusterOptions](#) in *AWS CLI Command Reference*.

describe-reserved-node-offerings

The following code example shows how to use `describe-reserved-node-offerings`.

AWS CLI

Describe Reserved Node Offerings This example shows all of the reserved node offerings that are available for purchase.
Command:

```
aws redshift describe-reserved-node-offerings
```

Result:

```

{
  "ReservedNodeOfferings": [
    {
      "OfferingType": "Heavy Utilization",
      "FixedPrice": "",
      "NodeType": "dw.hs1.xlarge",
      "UsagePrice": "",
      "RecurringCharges": [
        {
          "RecurringChargeAmount": "",
          "RecurringChargeFrequency": "Hourly"
        }
      ],
      "Duration": 31536000,
      "ReservedNodeOfferingId": "ceb6a579-cf4c-4343-be8b-d832c45ab51c"
    },
    {
      "OfferingType": "Heavy Utilization",
      "FixedPrice": "",
      "NodeType": "dw.hs1.8xlarge",

```



```

    "UsagePrice": "",
    "RecurringCharges": [
      {
        "RecurringChargeAmount": "",
        "RecurringChargeFrequency": "Hourly"
      } ],
    "Duration": 31536000,
    "ReservedNodeOfferingId": "e5a2ff3b-352d-4a9c-ad7d-373c4cab5dd2"
  },
  ...remaining output omitted...
],
"ResponseMetadata": {
  "RequestId": "8b1a1a43-75ff-11e2-9666-e142fe91ddd1"
}
}

```

If you want to purchase a reserved node offering, you can call `purchase-reserved-node-offering` using a valid *ReservedNodeOfferingId*.

`purchase-reserved-node-offering` using a valid *ReservedNodeOfferingId*.

using a valid *ReservedNodeOfferingId*.

ReservedNodeOfferingId.

.

- For API details, see [DescribeReservedNodeOfferings](#) in *AWS CLI Command Reference*.

describe-reserved-nodes

The following code example shows how to use `describe-reserved-nodes`.

AWS CLI

Describe Reserved Nodes This example shows a reserved node offering that has been purchased. Command:

```
aws redshift describe-reserved-nodes
```

Result:

```
{
```

```

"ResponseMetadata": {
  "RequestId": "bc29ce2e-7600-11e2-9949-4b361e7420b7"
},
"ReservedNodes": [
  {
    "OfferingType": "Heavy Utilization",
    "FixedPrice": "",
    "NodeType": "dw.hs1.xlarge",
    "ReservedNodeId": "1ba8e2e3-bc01-4d65-b35d-a4a3e931547e",
    "UsagePrice": "",
    "RecurringCharges": [
      {
        "RecurringChargeAmount": "",
        "RecurringChargeFrequency": "Hourly"
      } ],
    "NodeCount": 1,
    "State": "payment-pending",
    "StartTime": "2013-02-13T17:08:39.051Z",
    "Duration": 31536000,
    "ReservedNodeOfferingId": "ceb6a579-cf4c-4343-be8b-d832c45ab51c"
  }
]
}

```

- For API details, see [DescribeReservedNodes](#) in *AWS CLI Command Reference*.

describe-resize

The following code example shows how to use `describe-resize`.

AWS CLI

Describe ResizeThis example describes the latest resize of a cluster. The request was for 3 nodes of type `dw.hs1.8xlarge`. Command:

```
aws redshift describe-resize --cluster-identifier mycluster
```

Result:

```

{
  "Status": "NONE",
  "TargetClusterType": "multi-node",

```

```
"TargetNodeType": "dw.hs1.8xlarge",
"ResponseMetadata": {
  "RequestId": "9f52b0b4-7733-11e2-aa9b-318b2909bd27"
},
"TargetNumberOfNodes": "3"
}
```

- For API details, see [DescribeResize](#) in *AWS CLI Command Reference*.

describe-scheduled-actions

The following code example shows how to use describe-scheduled-actions.

AWS CLI

To describe scheduled actions

The following describe-scheduled-actions example displays details for any currently scheduled actions.

```
aws redshift describe-scheduled-actions
```

Output:

```
{
  "ScheduledActions": [
    {
      "ScheduledActionName": "resizecluster",
      "TargetAction": {
        "ResizeCluster": {
          "ClusterIdentifier": "mycluster",
          "NumberOfNodes": 4,
          "Classic": false
        }
      },
      "Schedule": "at(2019-12-10T00:07:00)",
      "IamRole": "arn:aws:iam::123456789012:role/myRedshiftRole",
      "State": "ACTIVE",
      "NextInvocations": [
        "2019-12-10T00:07:00Z"
      ]
    }
  ]
}
```

```
]
}
```

- For API details, see [DescribeScheduledActions](#) in *AWS CLI Command Reference*.

describe-snapshot-copy-grants

The following code example shows how to use `describe-snapshot-copy-grants`.

AWS CLI

To describe snapshot copy grants

The following `describe-snapshot-copy-grants` example displays details for the specified cluster snapshot copy grant.

```
aws redshift describe-snapshot-copy-grants \
  --snapshot-copy-grant-name mysnapshotcopygrantname
```

Output:

```
{
  "SnapshotCopyGrants": [
    {
      "SnapshotCopyGrantName": "mysnapshotcopygrantname",
      "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/
bPxRfih3yCo8nvbEXAMPLEKEY",
      "Tags": []
    }
  ]
}
```

For more information, see [Amazon Redshift Database Encryption](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [DescribeSnapshotCopyGrants](#) in *AWS CLI Command Reference*.

describe-snapshot-schedules

The following code example shows how to use `describe-snapshot-schedules`.

AWS CLI

To describe snapshot schedules

The following `describe-snapshot-schedules` example displays details for the specified cluster snapshot schedule.

```
aws redshift describe-snapshot-schedules \  
  --cluster-identifier mycluster \  
  --schedule-identifier mysnapshotschedule
```

Output:

```
{  
  "SnapshotSchedules": [  
    {  
      "ScheduleDefinitions": [  
        "rate(12 hours)"  
      ],  
      "ScheduleIdentifier": "mysnapshotschedule",  
      "ScheduleDescription": "My schedule description",  
      "Tags": [],  
      "AssociatedClusterCount": 1,  
      "AssociatedClusters": [  
        {  
          "ClusterIdentifier": "mycluster",  
          "ScheduleAssociationState": "ACTIVE"  
        }  
      ]  
    }  
  ]  
}
```

For more information, see [Automated Snapshot Schedules](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [DescribeSnapshotSchedules](#) in *AWS CLI Command Reference*.

describe-storage

The following code example shows how to use `describe-storage`.

AWS CLI

To describe storage

The following `describe-storage` example displays details about the backup storage and provisional storage sizes for the account.

```
aws redshift describe-storage
```

Output:

```
{
  "TotalBackupSizeInMegaBytes": 193149.0,
  "TotalProvisionedStorageInMegaBytes": 655360.0
}
```

For more information, see [Managing Snapshot Storage](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [DescribeStorage](#) in *AWS CLI Command Reference*.

`describe-table-restore-status`

The following code example shows how to use `describe-table-restore-status`.

AWS CLI

To describe status of table restore requests from a cluster snapshot

The following `describe-table-restore-status` example displays details for table restore requests made for the specified cluster.

```
aws redshift describe-table-restore-status /
  --cluster-identifier mycluster
```

Output:

```
{
  "TableRestoreStatusDetails": [
    {
      "TableRestoreRequestId": "z1116630-0e80-46f4-ba86-bd9670411ebd",
      "Status": "IN_PROGRESS",

```

```

    "RequestTime": "2019-12-27T18:22:12.257Z",
    "ClusterIdentifier": "mycluster",
    "SnapshotIdentifier": "mysnapshotid",
    "SourceDatabaseName": "dev",
    "SourceSchemaName": "public",
    "SourceTableName": "mytable",
    "TargetDatabaseName": "dev",
    "TargetSchemaName": "public",
    "NewTableName": "mytable-clone"
  }
]
}

```

For more information, see [Restoring a Table from a Snapshot](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [DescribeTableRestoreStatus](#) in *AWS CLI Command Reference*.

describe-tags

The following code example shows how to use `describe-tags`.

AWS CLI

To describe tags

The following `describe-tags` example displays the resources the specified cluster associated with the specified tag names and values.

```

aws redshift describe-tags \
  --resource-name arn:aws:redshift:us-west-2:123456789012:cluster:mycluster \
  --tag-keys clustertagkey \
  --tag-values clustertagvalue

```

Output:

```

{
  "TaggedResources": [
    {
      "Tag": {
        "Key": "clustertagkey",
        "Value": "clustertagvalue"
      },

```

```
        "ResourceName": "arn:aws:redshift:us-  
west-2:123456789012:cluster:mycluster",  
        "ResourceType": "cluster"  
    }  
]  
}
```

For more information, see [Tagging Resources in Amazon Redshift](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [DescribeTags](#) in *AWS CLI Command Reference*.

disable-snapshot-copy

The following code example shows how to use `disable-snapshot-copy`.

AWS CLI

To disable snapshot copy for a cluster

The following `disable-snapshot-copy` example disables the automatic copy of a snapshot for the specified cluster.

```
aws redshift disable-snapshot-copy \  
--cluster-identifier mycluster
```

Output:

```
{  
  "Cluster": {  
    "ClusterIdentifier": "mycluster",  
    "NodeType": "dc2.large",  
    "ClusterStatus": "available",  
    "ClusterAvailabilityStatus": "Available",  
    "MasterUsername": "adminuser",  
    "DBName": "dev",  
    "Endpoint": {  
      "Address": "mycluster.cmeaswqeuae.us-west-2.redshift.amazonaws.com",  
      "Port": 5439  
    },  
    "ClusterCreateTime": "2019-12-05T18:44:36.991Z",  
    "AutomatedSnapshotRetentionPeriod": 3,  
    "ManualSnapshotRetentionPeriod": -1,  
  },  
}
```



```
"ClusterSecurityGroups": [],
"VpcSecurityGroups": [
  {
    "VpcSecurityGroupId": "sh-i9b431cd",
    "Status": "active"
  }
],
"ClusterParameterGroups": [
  {
    "ParameterGroupName": "default.redshift-1.0",
    "ParameterApplyStatus": "in-sync"
  }
],
"ClusterSubnetGroupName": "default",
"VpcId": "vpc-b1fel7t9",
"AvailabilityZone": "us-west-2f",
"PreferredMaintenanceWindow": "sat:16:00-sat:16:30",
"PendingModifiedValues": {
  "NodeType": "dc2.large",
  "NumberOfNodes": 2,
  "ClusterType": "multi-node"
},
"ClusterVersion": "1.0",
"AllowVersionUpgrade": true,
"NumberOfNodes": 4,
"PubliclyAccessible": false,
"Encrypted": false,
"Tags": [
  {
    "Key": "mytags",
    "Value": "tag1"
  }
],
"EnhancedVpcRouting": false,
"IamRoles": [
  {
    "IamRoleArn": "arn:aws:iam::123456789012:role/myRedshiftRole",
    "ApplyStatus": "in-sync"
  }
],
"MaintenanceTrackName": "current",
"DeferredMaintenanceWindows": [],
"ExpectedNextSnapshotScheduleTime": "2019-12-10T04:42:43.390Z",
"ExpectedNextSnapshotScheduleTimeStatus": "OnTrack",
```

```
    "NextMaintenanceWindowStartTime": "2019-12-14T16:00:00Z"
  }
}
```

For more information, see [Copying Snapshots to Another AWS Region](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [DisableSnapshotCopy](#) in *AWS CLI Command Reference*.

enable-snapshot-copy

The following code example shows how to use `enable-snapshot-copy`.

AWS CLI

To enable snapshot copy for a cluster

The following `enable-snapshot-copy` example enables the automatic copy of a snapshot for the specified cluster.

```
aws redshift enable-snapshot-copy \
  --cluster-identifier mycluster \
  --destination-region us-west-1
```

Output:

```
{
  "Cluster": {
    "ClusterIdentifier": "mycluster",
    "NodeType": "dc2.large",
    "ClusterStatus": "available",
    "ClusterAvailabilityStatus": "Available",
    "MasterUsername": "adminuser",
    "DBName": "dev",
    "Endpoint": {
      "Address": "mycluster.cmeaswqeuae.us-west-2.redshift.amazonaws.com",
      "Port": 5439
    },
    "ClusterCreateTime": "2019-12-05T18:44:36.991Z",
    "AutomatedSnapshotRetentionPeriod": 3,
    "ManualSnapshotRetentionPeriod": -1,
    "ClusterSecurityGroups": [],
    "VpcSecurityGroups": [
```

```
    {
      "VpcSecurityGroupId": "sh-f4c731cd",
      "Status": "active"
    }
  ],
  "ClusterParameterGroups": [
    {
      "ParameterGroupName": "default.redshift-1.0",
      "ParameterApplyStatus": "in-sync"
    }
  ],
  "ClusterSubnetGroupName": "default",
  "VpcId": "vpc-b1ael7t9",
  "AvailabilityZone": "us-west-2f",
  "PreferredMaintenanceWindow": "sat:16:00-sat:16:30",
  "PendingModifiedValues": {
    "NodeType": "dc2.large",
    "NumberOfNodes": 2,
    "ClusterType": "multi-node"
  },
  "ClusterVersion": "1.0",
  "AllowVersionUpgrade": true,
  "NumberOfNodes": 4,
  "PubliclyAccessible": false,
  "Encrypted": false,
  "ClusterSnapshotCopyStatus": {
    "DestinationRegion": "us-west-1",
    "RetentionPeriod": 7,
    "ManualSnapshotRetentionPeriod": -1
  },
  "Tags": [
    {
      "Key": "mytags",
      "Value": "tag1"
    }
  ],
  "EnhancedVpcRouting": false,
  "IamRoles": [
    {
      "IamRoleArn": "arn:aws:iam::123456789012:role/myRedshiftRole",
      "ApplyStatus": "in-sync"
    }
  ],
  "MaintenanceTrackName": "current",
```

```
"DeferredMaintenanceWindows": [],
"ExpectedNextSnapshotScheduleTime": "2019-12-10T04:42:43.390Z",
"ExpectedNextSnapshotScheduleTimeStatus": "OnTrack",
"NextMaintenanceWindowStartTime": "2019-12-14T16:00:00Z"
}
}
```

For more information, see [Copying Snapshots to Another AWS Region](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [EnableSnapshotCopy](#) in *AWS CLI Command Reference*.

get-cluster-credentials

The following code example shows how to use `get-cluster-credentials`.

AWS CLI

To get cluster credentials for an AWS account

The following `get-cluster-credentials` example retrieves temporary credentials that enable access to an Amazon Redshift database.

```
aws redshift get-cluster-credentials \
  --db-user adminuser --db-name dev \
  --cluster-identifier mycluster
```

Output:

```
{
  "DbUser": "IAM:adminuser",
  "DbPassword": "AMAFUyyuros/QjxPTtgzcsuQsqzIasdzJEN04aCtWDzXx109d6UmpkBtvEqFly/
EXAMPLE==",
  "Expiration": "2019-12-10T17:25:05.770Z"
}
```

For more information, see [Generating IAM Database Credentials Using the Amazon Redshift CLI or API](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [GetClusterCredentials](#) in *AWS CLI Command Reference*.

get-reserved-node-exchange-offerings

The following code example shows how to use `get-reserved-node-exchange-offerings`.

AWS CLI

To get reserved node exchange offerings

The following `get-reserved-node-exchange-offerings` example retrieves an array of DC2 `ReservedNodeOfferings` that match the specified DC1 reserved node.

```
aws redshift get-reserved-node-exchange-offerings \  
  --reserved-node-id 12345678-12ab-12a1-1a2a-12ab-12a12EXAMPLE
```

Output:

```
{  
  "ReservedNodeOfferings": [  
    {  
      "ReservedNodeOfferingId": "12345678-12ab-12a1-1a2a-12ab-12a12EXAMPLE",  
      "NodeType": "dc2.large",  
      "Duration": 31536000,  
      "FixedPrice": 0.0,  
      "UsagePrice": 0.0,  
      "CurrencyCode": "USD",  
      "OfferingType": "All Upfront",  
      "RecurringCharges": [  
        {  
          "RecurringChargeAmount": 0.0,  
          "RecurringChargeFrequency": "Hourly"  
        }  
      ],  
      "ReservedNodeOfferingType": "Regular"  
    }  
  ]  
}
```

For more information, see [Upgrading Reserved Nodes With the AWS CLI](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [GetReservedNodeExchangeOfferings](#) in *AWS CLI Command Reference*.

modify-cluster-iam-roles

The following code example shows how to use `modify-cluster-iam-roles`.

AWS CLI

To modify the IAM role for a cluster

The following `modify-cluster-iam-roles` example removes the specified AWS IAM role from the specified cluster.

```
aws redshift modify-cluster-iam-roles \  
  --cluster-identifier mycluster \  
  --remove-iam-roles arn:aws:iam::123456789012:role/myRedshiftRole
```

Output:

```
{  
  "Cluster": {  
    "ClusterIdentifier": "mycluster",  
    "NodeType": "dc2.large",  
    "ClusterStatus": "available",  
    "ClusterAvailabilityStatus": "Available",  
    "MasterUsername": "adminuser",  
    "DBName": "dev",  
    "Endpoint": {  
      "Address": "mycluster.cmeaswqeuae.us-west-2.redshift.amazonaws.com",  
      "Port": 5439  
    },  
    "ClusterCreateTime": "2019-12-05T18:44:36.991Z",  
    "AutomatedSnapshotRetentionPeriod": 3,  
    "ManualSnapshotRetentionPeriod": -1,  
    "ClusterSecurityGroups": [],  
    "VpcSecurityGroups": [  
      {  
        "VpcSecurityGroupId": "sh-f9b731sd",  
        "Status": "active"  
      }  
    ],  
    "ClusterParameterGroups": [  
      {  
        "ParameterGroupName": "default.redshift-1.0",  
        "ParameterApplyStatus": "in-sync"  
      }  
    ]  
  }  
}
```

```
    }
  ],
  "ClusterSubnetGroupName": "default",
  "VpcId": "vpc-b2fal7t9",
  "AvailabilityZone": "us-west-2f",
  "PreferredMaintenanceWindow": "sat:16:00-sat:16:30",
  "PendingModifiedValues": {
    "NodeType": "dc2.large",
    "NumberOfNodes": 2,
    "ClusterType": "multi-node"
  },
  "ClusterVersion": "1.0",
  "AllowVersionUpgrade": true,
  "NumberOfNodes": 4,
  "PubliclyAccessible": false,
  "Encrypted": false,
  "ClusterSnapshotCopyStatus": {
    "DestinationRegion": "us-west-1",
    "RetentionPeriod": 7,
    "ManualSnapshotRetentionPeriod": -1
  },
  "Tags": [
    {
      "Key": "mytags",
      "Value": "tag1"
    }
  ],
  "EnhancedVpcRouting": false,
  "IamRoles": [],
  "MaintenanceTrackName": "current",
  "DeferredMaintenanceWindows": [],
  "ExpectedNextSnapshotScheduleTime": "2019-12-11T04:42:55.631Z",
  "ExpectedNextSnapshotScheduleTimeStatus": "OnTrack",
  "NextMaintenanceWindowStartTime": "2019-12-14T16:00:00Z"
}
}
```

For more information, see [Using Identity-Based Policies \(IAM Policies\) for Amazon Redshift](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [ModifyClusterIamRoles](#) in *AWS CLI Command Reference*.

modify-cluster-maintenance

The following code example shows how to use `modify-cluster-maintenance`.

AWS CLI

To modify cluster maintenance

The following `modify-cluster-maintenance` example defers the maintenance of the specified cluster by 30 days.

```
aws redshift modify-cluster-maintenance \  
  --cluster-identifier mycluster \  
  --defer-maintenance \  
  --defer-maintenance-duration 30
```

Output:

```
{  
  "Cluster": {  
    "ClusterIdentifier": "mycluster",  
    "NodeType": "dc2.large",  
    "ClusterStatus": "available",  
    "ClusterAvailabilityStatus": "Available",  
    "MasterUsername": "adminuser",  
    "DBName": "dev",  
    "Endpoint": {  
      "Address": "mycluster.cmeaswquae.us-west-2.redshift.amazonaws.com",  
      "Port": 5439  
    },  
    "ClusterCreateTime": "2019-12-05T18:44:36.991Z",  
    "AutomatedSnapshotRetentionPeriod": 3,  
    "ManualSnapshotRetentionPeriod": -1,  
    "ClusterSecurityGroups": [],  
    "VpcSecurityGroups": [  
      {  
        "VpcSecurityGroupId": "sh-a1a123ab",  
        "Status": "active"  
      }  
    ],  
    "ClusterParameterGroups": [  
      {  
        "ParameterGroupName": "default.redshift-1.0",
```



```
        "ParameterApplyStatus": "in-sync"
    }
],
"ClusterSubnetGroupName": "default",
"VpcId": "vpc-b1ael7t9",
"AvailabilityZone": "us-west-2f",
"PreferredMaintenanceWindow": "sat:16:00-sat:16:30",
"PendingModifiedValues": {
    "NodeType": "dc2.large",
    "NumberOfNodes": 2,
    "ClusterType": "multi-node"
},
"ClusterVersion": "1.0",
"AllowVersionUpgrade": true,
"NumberOfNodes": 4,
"PubliclyAccessible": false,
"Encrypted": false,
"ClusterSnapshotCopyStatus": {
    "DestinationRegion": "us-west-1",
    "RetentionPeriod": 7,
    "ManualSnapshotRetentionPeriod": -1
},
"Tags": [
    {
        "Key": "mytags",
        "Value": "tag1"
    }
],
"EnhancedVpcRouting": false,
"IamRoles": [],
"MaintenanceTrackName": "current",
"DeferredMaintenanceWindows": [
    {
        "DeferMaintenanceIdentifier": "dfm-mUdVIffFcT1B4SGhw6fyF",
        "DeferMaintenanceStartTime": "2019-12-10T18:18:39.354Z",
        "DeferMaintenanceEndTime": "2020-01-09T18:18:39.354Z"
    }
],
"ExpectedNextSnapshotScheduleTime": "2019-12-11T04:42:55.631Z",
"ExpectedNextSnapshotScheduleTimeStatus": "OnTrack",
"NextMaintenanceWindowStartTime": "2020-01-11T16:00:00Z"
}
}
```

For more information, see [Cluster Maintenance](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [ModifyClusterMaintenance](#) in *AWS CLI Command Reference*.

modify-cluster-parameter-group

The following code example shows how to use `modify-cluster-parameter-group`.

AWS CLI

Modify a parameter in a parameter group

The following `modify-cluster-parameter-group` example modifies the `wlm_json_configuration` parameter for workload management. It accepts the parameters from a file that contains the JSON contents shown below.

```
aws redshift modify-cluster-parameter-group \  
  --parameter-group-name myclusterparametergroup \  
  --parameters file://modify_pg.json
```

Contents of `modify_pg.json`:

```
[  
  {  
    "ParameterName": "wlm_json_configuration",  
    "ParameterValue": "[{\\"user_group\\":\\"example_user_group1\\",\\"query_group\\":  
  \\"example_query_group1\\", \\"query_concurrency\\":7},{\\"query_concurrency\\":5}]"  
  }  
]
```

Output:

```
{  
  "ParameterGroupStatus": "Your parameter group has been updated but changes won't  
  get applied until you reboot the associated Clusters.",  
  "ParameterGroupName": "myclusterparametergroup",  
  "ResponseMetadata": {  
    "RequestId": "09974cc0-64cd-11e2-bea9-49e0ce183f07"  
  }  
}
```

- For API details, see [ModifyClusterParameterGroup](#) in *AWS CLI Command Reference*.

modify-cluster-snapshot-schedule

The following code example shows how to use `modify-cluster-snapshot-schedule`.

AWS CLI

To modify cluster snapshot schedule

The following `modify-cluster-snapshot-schedule` example removes the specified snapshot schedule from the specified cluster.

```
aws redshift modify-cluster-snapshot-schedule \  
  --cluster-identifier mycluster \  
  --schedule-identifier mysnapshotschedule \  
  --disassociate-schedule
```

This command does not produce any output.

For more information, see [Automated Snapshot Schedules](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [ModifyClusterSnapshotSchedule](#) in *AWS CLI Command Reference*.

modify-cluster-snapshot

The following code example shows how to use `modify-cluster-snapshot`.

AWS CLI

To modify cluster snapshot

The following `modify-cluster-snapshot` example sets the manual retention period setting for the specified cluster snapshot to value of 10 days.

```
aws redshift modify-cluster-snapshot \  
  --snapshot-identifier mycluster-2019-11-06-16-32 \  
  --manual-snapshot-retention-period 10
```

Output:

```
{
  "Snapshot": {
    "SnapshotIdentifier": "mycluster-2019-11-06-16-32",
    "ClusterIdentifier": "mycluster",
    "SnapshotCreateTime": "2019-12-07T00:34:05.633Z",
    "Status": "available",
    "Port": 5439,
    "AvailabilityZone": "us-west-2f",
    "ClusterCreateTime": "2019-12-05T18:44:36.991Z",
    "MasterUsername": "adminuser",
    "ClusterVersion": "1.0",
    "SnapshotType": "manual",
    "NodeType": "dc2.large",
    "NumberOfNodes": 2,
    "DBName": "dev",
    "VpcId": "vpc-b1cel7t9",
    "Encrypted": false,
    "EncryptedWithHSM": false,
    "OwnerAccount": "123456789012",
    "TotalBackupSizeInMegaBytes": 64384.0,
    "ActualIncrementalBackupSizeInMegaBytes": 24.0,
    "BackupProgressInMegaBytes": 24.0,
    "CurrentBackupRateInMegaBytesPerSecond": 13.0011,
    "EstimatedSecondsToCompletion": 0,
    "ElapsedTimeInSeconds": 1,
    "Tags": [
      {
        "Key": "mytagkey",
        "Value": "mytagvalue"
      }
    ],
    "EnhancedVpcRouting": false,
    "MaintenanceTrackName": "current",
    "ManualSnapshotRetentionPeriod": 10,
    "ManualSnapshotRemainingDays": 6,
    "SnapshotRetentionStartTime": "2019-12-07T00:34:07.479Z"
  }
}
```

For more information, see [Amazon Redshift Snapshots](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [ModifyClusterSnapshot](#) in *AWS CLI Command Reference*.

modify-cluster-subnet-group

The following code example shows how to use `modify-cluster-subnet-group`.

AWS CLI

Modify the Subnets in a Cluster Subnet Group This example shows how to modify the list of subnets in a cache subnet group. By default, the output is in JSON format.**Command:**

```
aws redshift modify-cluster-subnet-group --cluster-subnet-group-name mysubnetgroup
--subnet-ids subnet-763fdd1 subnet-ac830e9
```

Result:

```
{
  "ClusterSubnetGroup": {
    "Subnets": [
      {
        "SubnetStatus": "Active",
        "SubnetIdentifier": "subnet-763fdd1c",
        "SubnetAvailabilityZone": {
          "Name": "us-east-1a"
        }
      },
      {
        "SubnetStatus": "Active",
        "SubnetIdentifier": "subnet-ac830e9",
        "SubnetAvailabilityZone": {
          "Name": "us-east-1b"
        }
      }
    ],
    "VpcId": "vpc-7e3fdd14",
    "SubnetGroupStatus": "Complete",
    "Description": "My subnet group",
    "ClusterSubnetGroupName": "mysubnetgroup"
  },
  "ResponseMetadata": {
    "RequestId": "8da93e89-8372-f936-93a8-873918938197a"
  }
}
```

- For API details, see [ModifyClusterSubnetGroup](#) in *AWS CLI Command Reference*.

modify-cluster

The following code example shows how to use `modify-cluster`.

AWS CLI

Associate a Security Group with a Cluster This example shows how to associate a cluster security group with the specified cluster. Command:

```
aws redshift modify-cluster --cluster-identifier mycluster --cluster-security-groups mysecuritygroup
```

Modify the Maintenance Window for a Cluster This shows how to change the weekly preferred maintenance window for a cluster to be the minimum four hour window starting Sundays at 11:15 PM, and ending Mondays at 3:15 AM. Command:

```
aws redshift modify-cluster --cluster-identifier mycluster --preferred-maintenance-window Sun:23:15-Mon:03:15
```

Change the Master Password for the Cluster This example shows how to change the master password for a cluster. Command:

```
aws redshift modify-cluster --cluster-identifier mycluster --master-user-password A1b2c3d4
```

- For API details, see [ModifyCluster](#) in *AWS CLI Command Reference*.

modify-event-subscription

The following code example shows how to use `modify-event-subscription`.

AWS CLI

To modify event subscription

The following `modify-event-subscription` example disables the specified event notification subscription.

```
aws redshift modify-event-subscription \  
  --subscription-name mysubscription \  
  --no-enabled
```

Output:

```
{
  "EventSubscription": {
    "CustomerAwsId": "123456789012",
    "CustSubscriptionId": "mysubscription",
    "SnsTopicArn": "arn:aws:sns:us-west-2:123456789012:MySNStopic",
    "Status": "active",
    "SubscriptionCreationTime": "2019-12-09T21:50:21.332Z",
    "SourceIdsList": [],
    "EventCategoriesList": [
      "management"
    ],
    "Severity": "ERROR",
    "Enabled": false,
    "Tags": []
  }
}
```

For more information, see [Subscribing to Amazon Redshift Event Notifications](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [ModifyEventSubscription](#) in *AWS CLI Command Reference*.

modify-scheduled-action

The following code example shows how to use `modify-scheduled-action`.

AWS CLI**To modify scheduled action**

The following `modify-scheduled-action` example adds a description to the specified existing scheduled action.

```
aws redshift modify-scheduled-action \
  --scheduled-action-name myscheduledaction \
  --scheduled-action-description "My scheduled action"
```

Output:

```
{
```

```

    "ScheduledActionName": "myscheduledaction",
    "TargetAction": {
      "ResizeCluster": {
        "ClusterIdentifier": "mycluster",
        "NumberOfNodes": 2,
        "Classic": false
      }
    },
    "Schedule": "at(2019-12-25T00:00:00)",
    "IamRole": "arn:aws:iam::123456789012:role/myRedshiftRole",
    "ScheduledActionDescription": "My scheduled action",
    "State": "ACTIVE",
    "NextInvocations": [
      "2019-12-25T00:00:00Z"
    ]
  }
}

```

- For API details, see [ModifyScheduledAction](#) in *AWS CLI Command Reference*.

modify-snapshot-copy-retention-period

The following code example shows how to use `modify-snapshot-copy-retention-period`.

AWS CLI

To modify snapshot copy retention period

The following `modify-snapshot-copy-retention-period` example modifies the number of days to retain snapshots for the specified cluster in the destination AWS Region after they are copied from the source AWS Region.

```

aws redshift modify-snapshot-copy-retention-period \
  --cluster-identifier mycluster \
  --retention-period 15

```

Output:

```

{
  "Cluster": {
    "ClusterIdentifier": "mycluster",
    "NodeType": "dc2.large",
    "ClusterStatus": "available",

```



```
"ClusterAvailabilityStatus": "Available",
"MasterUsername": "adminuser",
"DBName": "dev",
"Endpoint": {
  "Address": "mycluster.cmeaswqeuae.us-west-2.redshift.amazonaws.com",
  "Port": 5439
},
"ClusterCreateTime": "2019-12-05T18:44:36.991Z",
"AutomatedSnapshotRetentionPeriod": 3,
"ManualSnapshotRetentionPeriod": -1,
"ClusterSecurityGroups": [],
"VpcSecurityGroups": [
  {
    "VpcSecurityGroupId": "sh-a1a123ab",
    "Status": "active"
  }
],
"ClusterParameterGroups": [
  {
    "ParameterGroupName": "default.redshift-1.0",
    "ParameterApplyStatus": "in-sync"
  }
],
"ClusterSubnetGroupName": "default",
"VpcId": "vpc-b1fet7t9",
"AvailabilityZone": "us-west-2f",
"PreferredMaintenanceWindow": "sat:16:00-sat:16:30",
"PendingModifiedValues": {
  "NodeType": "dc2.large",
  "NumberOfNodes": 2,
  "ClusterType": "multi-node"
},
"ClusterVersion": "1.0",
"AllowVersionUpgrade": true,
"NumberOfNodes": 4,
"PubliclyAccessible": false,
"Encrypted": false,
"ClusterSnapshotCopyStatus": {
  "DestinationRegion": "us-west-1",
  "RetentionPeriod": 15,
  "ManualSnapshotRetentionPeriod": -1
},
"Tags": [
  {
```

```

        "Key": "mytags",
        "Value": "tag1"
    }
],
"EnhancedVpcRouting": false,
"IamRoles": [],
"MaintenanceTrackName": "current",
"DeferredMaintenanceWindows": [
    {
        "DeferMaintenanceIdentifier": "dfm-mUdVSfDcT1F4SGhw6fyF",
        "DeferMaintenanceStartTime": "2019-12-10T18:18:39.354Z",
        "DeferMaintenanceEndTime": "2020-01-09T18:18:39.354Z"
    }
],
"NextMaintenanceWindowStartTime": "2020-01-11T16:00:00Z"
}
}

```

For more information, see [Snapshot Schedule Format](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [ModifySnapshotCopyRetentionPeriod](#) in *AWS CLI Command Reference*.

modify-snapshot-schedule

The following code example shows how to use `modify-snapshot-schedule`.

AWS CLI

To modify snapshot schedule

The following `modify-snapshot-schedule` example modifies the rate of the specified snapshot schedule to every 10 hours.

```

aws redshift modify-snapshot-schedule \
  --schedule-identifier mysnapshotschedule \
  --schedule-definitions "rate(10 hours)"

```

Output:

```

{
  "ScheduleDefinitions": [

```

```

    "rate(10 hours)"
  ],
  "ScheduleIdentifier": "mysnapshotschedule",
  "ScheduleDescription": "My schedule description",
  "Tags": []
}

```

For more information, see [Snapshot Schedule Format](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [ModifySnapshotSchedule](#) in *AWS CLI Command Reference*.

purchase-reserved-node-offering

The following code example shows how to use `purchase-reserved-node-offering`.

AWS CLI

Purchase a Reserved Node This example shows how to purchase a reserved node offering. The `reserved-node-offering-id` is obtained by calling `describe-reserved-node-offerings`. Command:

```
aws redshift purchase-reserved-node-offering --reserved-node-offering-id ceb6a579-cf4c-4343-be8b-d832c45ab51c
```

Result:

```

{
  "ReservedNode": {
    "OfferingType": "Heavy Utilization",
    "FixedPrice": "",
    "NodeType": "dw.hs1.xlarge",
    "ReservedNodeId": "1ba8e2e3-bc01-4d65-b35d-a4a3e931547e",
    "UsagePrice": "",
    "RecurringCharges": [
      {
        "RecurringChargeAmount": "",
        "RecurringChargeFrequency": "Hourly"
      }
    ],
    "NodeCount": 1,
  }
}

```

```
"State": "payment-pending",
"StartTime": "2013-02-13T17:08:39.051Z",
"Duration": 31536000,
"ReservedNodeOfferingId": "ceb6a579-cf4c-4343-be8b-d832c45ab51c"
},
"ResponseMetadata": {
  "RequestId": "01bda7bf-7600-11e2-b605-2568d7396e7f"
}
}
```

- For API details, see [PurchaseReservedNodeOffering](#) in *AWS CLI Command Reference*.

reboot-cluster

The following code example shows how to use `reboot-cluster`.

AWS CLI

Reboot a Cluster This example reboots a cluster. By default, the output is in JSON format.
Command:

```
aws redshift reboot-cluster --cluster-identifier mycluster
```

Result:

```
{
  "Cluster": {
    "NodeType": "dw.hs1.xlarge",
    "Endpoint": {
      "Port": 5439,
      "Address": "mycluster.coqoarplqhsn.us-east-1.redshift.amazonaws.com"
    },
    "ClusterVersion": "1.0",
    "PubliclyAccessible": "true",
    "MasterUsername": "adminuser",
    "ClusterParameterGroups": [
      {
        "ParameterApplyStatus": "in-sync",
        "ParameterGroupName": "default.redshift-1.0"
      }
    ],
    "ClusterSecurityGroups": [
```

```

    {
      "Status": "active",
      "ClusterSecurityGroupName": "default"
    }
  ],
  "AllowVersionUpgrade": true,
  "VpcSecurityGroups": [],
  "AvailabilityZone": "us-east-1a",
  "ClusterCreateTime": "2013-01-22T21:59:29.559Z",
  "PreferredMaintenanceWindow": "sun:23:15-mon:03:15",
  "AutomatedSnapshotRetentionPeriod": 1,
  "ClusterStatus": "rebooting",
  "ClusterIdentifier": "mycluster",
  "DBName": "dev",
  "NumberOfNodes": 2,
  "PendingModifiedValues": {}
},
"ResponseMetadata": {
  "RequestId": "61c8b564-64e8-11e2-8f7d-3b939af52818"
}
}

```

- For API details, see [RebootCluster](#) in *AWS CLI Command Reference*.

reset-cluster-parameter-group

The following code example shows how to use `reset-cluster-parameter-group`.

AWS CLI

Reset Parameters in a Parameter Group This example shows how to reset all of the parameters in a parameter group. Command:

```
aws redshift reset-cluster-parameter-group --parameter-group-name
myclusterparametergroup --reset-all-parameters
```

- For API details, see [ResetClusterParameterGroup](#) in *AWS CLI Command Reference*.

resize-cluster

The following code example shows how to use `resize-cluster`.

AWS CLI

To resize cluster

The following `resize-cluster` example resizes the specified cluster.

```
aws redshift resize-cluster \  
  --cluster-identifier mycluster \  
  --cluster-type multi-node \  
  --node-type dc2.large \  
  --number-of-nodes 6 \  
  --classic
```

Output:

```
{  
  "Cluster": {  
    "ClusterIdentifier": "mycluster",  
    "NodeType": "dc2.large",  
    "ClusterStatus": "resizing",  
    "ClusterAvailabilityStatus": "Modifying",  
    "MasterUsername": "adminuser",  
    "DBName": "dev",  
    "Endpoint": {  
      "Address": "mycluster.cmeaswqeuae.us-west-2.redshift.amazonaws.com",  
      "Port": 5439  
    },  
    "ClusterCreateTime": "2019-12-05T18:44:36.991Z",  
    "AutomatedSnapshotRetentionPeriod": 3,  
    "ManualSnapshotRetentionPeriod": -1,  
    "ClusterSecurityGroups": [],  
    "VpcSecurityGroups": [  
      {  
        "VpcSecurityGroupId": "sh-a1a123ab",  
        "Status": "active"  
      }  
    ],  
    "ClusterParameterGroups": [  
      {  
        "ParameterGroupName": "default.redshift-1.0",  
        "ParameterApplyStatus": "in-sync"  
      }  
    ],  
  },  
}
```

```
"ClusterSubnetGroupName": "default",
"VpcId": "vpc-a1abc1a1",
"AvailabilityZone": "us-west-2f",
"PreferredMaintenanceWindow": "sat:16:00-sat:16:30",
"PendingModifiedValues": {
  "NodeType": "dc2.large",
  "NumberOfNodes": 6,
  "ClusterType": "multi-node"
},
"ClusterVersion": "1.0",
"AllowVersionUpgrade": true,
"NumberOfNodes": 4,
"PubliclyAccessible": false,
"Encrypted": false,
"ClusterSnapshotCopyStatus": {
  "DestinationRegion": "us-west-1",
  "RetentionPeriod": 15,
  "ManualSnapshotRetentionPeriod": -1
},
"Tags": [
  {
    "Key": "mytags",
    "Value": "tag1"
  }
],
"EnhancedVpcRouting": false,
"IamRoles": [],
"MaintenanceTrackName": "current",
"DeferredMaintenanceWindows": [
  {
    "DeferMaintenanceIdentifier": "dfm-mUdVCfDcT1B4SGhw6fyF",
    "DeferMaintenanceStartTime": "2019-12-10T18:18:39.354Z",
    "DeferMaintenanceEndTime": "2020-01-09T18:18:39.354Z"
  }
],
"NextMaintenanceWindowStartTime": "2020-01-11T16:00:00Z",
"ResizeInfo": {
  "ResizeType": "ClassicResize",
  "AllowCancelResize": true
}
}
```

For more information, see [Resizing a Cluster](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [ResizeCluster](#) in *AWS CLI Command Reference*.

restore-from-cluster-snapshot

The following code example shows how to use `restore-from-cluster-snapshot`.

AWS CLI

Restore a Cluster From a Snapshot This example restores a cluster from a snapshot. Command:

```
aws redshift restore-from-cluster-snapshot --cluster-identifier mycluster-clone --
snapshot-identifier my-snapshot-id
```

Result:

```
{
  "Cluster": {
    "NodeType": "dw.hs1.xlarge",
    "ClusterVersion": "1.0",
    "PubliclyAccessible": "true",
    "MasterUsername": "adminuser",
    "ClusterParameterGroups": [
      {
        "ParameterApplyStatus": "in-sync",
        "ParameterGroupName": "default.redshift-1.0"
      }
    ],
    "ClusterSecurityGroups": [
      {
        "Status": "active",
        "ClusterSecurityGroupName": "default"
      }
    ],
    "AllowVersionUpgrade": true,
    "VpcSecurityGroups": [],
    "PreferredMaintenanceWindow": "sun:23:15-mon:03:15",
    "AutomatedSnapshotRetentionPeriod": 1,
    "ClusterStatus": "creating",
    "ClusterIdentifier": "mycluster-clone",
    "DBName": "dev",
    "NumberOfNodes": 2,
```



```
    "PendingModifiedValues": {}
  },
  "ResponseMetadata": {
    "RequestId": "77fd512b-64e3-11e2-8f5b-e90bd6c77476"
  }
}
```

- For API details, see [RestoreFromClusterSnapshot](#) in *AWS CLI Command Reference*.

restore-table-from-cluster-snapshot

The following code example shows how to use `restore-table-from-cluster-snapshot`.

AWS CLI

To restore table from a cluster snapshot

The following `restore-table-from-cluster-snapshot` example creates a new table from the specified table in the specified cluster snapshot.

```
aws redshift restore-table-from-cluster-snapshot /
  --cluster-identifier mycluster /
  --snapshot-identifier mycluster-2019-11-19-16-17 /
  --source-database-name dev /
  --source-schema-name public /
  --source-table-name mytable /
  --target-database-name dev /
  --target-schema-name public /
  --new-table-name mytable-clone
```

Output:

```
{
  "TableRestoreStatus": {
    "TableRestoreRequestId": "a123a12b-abc1-1a1a-a123-a1234ab12345",
    "Status": "PENDING",
    "RequestTime": "2019-12-20T00:20:16.402Z",
    "ClusterIdentifier": "mycluster",
    "SnapshotIdentifier": "mycluster-2019-11-19-16-17",
    "SourceDatabaseName": "dev",
    "SourceSchemaName": "public",
    "SourceTableName": "mytable",
```

```
    "TargetDatabaseName": "dev",
    "TargetSchemaName": "public",
    "NewTableName": "mytable-clone"
  }
}
```

For more information, see [Restoring a Table from a Snapshot](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [RestoreTableFromClusterSnapshot](#) in *AWS CLI Command Reference*.

revoke-cluster-security-group-ingress

The following code example shows how to use `revoke-cluster-security-group-ingress`.

AWS CLI

Revoke Access from an EC2 Security Group This example revokes access to a named Amazon EC2 security group. Command:

```
aws redshift revoke-cluster-security-group-ingress --cluster-security-group-name
mysecuritygroup --ec2-security-group-name myec2securitygroup --ec2-security-group-
owner-id 123445677890
```

Revoking Access to a CIDR range This example revokes access to a CIDR range. Command:

```
aws redshift revoke-cluster-security-group-ingress --cluster-security-group-name
mysecuritygroup --cidrip 192.168.100.100/32
```

- For API details, see [RevokeClusterSecurityGroupIngress](#) in *AWS CLI Command Reference*.

revoke-snapshot-access

The following code example shows how to use `revoke-snapshot-access`.

AWS CLI

Revoke the Authorization of an AWS Account to Restore a Snapshot This example revokes the authorization of the AWS account 444455556666 to restore the snapshot `my-snapshot-id`. By default, the output is in JSON format. Command:

```
aws redshift revoke-snapshot-access --snapshot-id my-snapshot-id --account-with-restore-access 444455556666
```

Result:

```
{
  "Snapshot": {
    "Status": "available",
    "SnapshotCreateTime": "2013-07-17T22:04:18.947Z",
    "EstimatedSecondsToCompletion": 0,
    "AvailabilityZone": "us-east-1a",
    "ClusterVersion": "1.0",
    "MasterUsername": "adminuser",
    "Encrypted": false,
    "OwnerAccount": "111122223333",
    "BackupProgressInMegabytes": 11.0,
    "ElapsedTimeInSeconds": 0,
    "DBName": "dev",
    "CurrentBackupRateInMegabytesPerSecond": 0.1534,
    "ClusterCreateTime": "2013-01-22T21:59:29.559Z",
    "ActualIncrementalBackupSizeInMegabytes": 11.0,
    "SnapshotType": "manual",
    "NodeType": "dw.hs1.xlarge",
    "ClusterIdentifier": "mycluster",
    "TotalBackupSizeInMegabytes": 20.0,
    "Port": 5439,
    "NumberOfNodes": 2,
    "SnapshotIdentifier": "my-snapshot-id"
  }
}
```

- For API details, see [RevokeSnapshotAccess](#) in *AWS CLI Command Reference*.

rotate-encryption-key

The following code example shows how to use rotate-encryption-key.

AWS CLI

To rotate encryption key for a cluster

The following rotate-encryption-key example rotates the encryption key for the specified cluster.

```
aws redshift rotate-encryption-key \  
  --cluster-identifier mycluster
```

Output:

```
{  
  "Cluster": {  
    "ClusterIdentifier": "mycluster",  
    "NodeType": "dc2.large",  
    "ClusterStatus": "rotating-keys",  
    "ClusterAvailabilityStatus": "Modifying",  
    "MasterUsername": "adminuser",  
    "DBName": "dev",  
    "Endpoint": {  
      "Address": "mycluster.cmeaswqeuae.us-west-2.redshift.amazonaws.com",  
      "Port": 5439  
    },  
    "ClusterCreateTime": "2019-12-10T19:25:45.886Z",  
    "AutomatedSnapshotRetentionPeriod": 30,  
    "ManualSnapshotRetentionPeriod": -1,  
    "ClusterSecurityGroups": [],  
    "VpcSecurityGroups": [  
      {  
        "VpcSecurityGroupId": "sh-a1a123ab",  
        "Status": "active"  
      }  
    ],  
    "ClusterParameterGroups": [  
      {  
        "ParameterGroupName": "default.redshift-1.0",  
        "ParameterApplyStatus": "in-sync"  
      }  
    ],  
    "ClusterSubnetGroupName": "default",  
    "VpcId": "vpc-a1abc1a1",  
    "AvailabilityZone": "us-west-2a",  
    "PreferredMaintenanceWindow": "sat:16:00-sat:16:30",  
    "PendingModifiedValues": {},  
    "ClusterVersion": "1.0",  
    "AllowVersionUpgrade": true,  
  }  
}
```

```
    "NumberOfNodes": 2,
    "PubliclyAccessible": false,
    "Encrypted": true,
    "Tags": [],
    "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/
bPxRfih3yCo8nvbEXAMPLEKEY",
    "EnhancedVpcRouting": false,
    "IamRoles": [
      {
        "IamRoleArn": "arn:aws:iam::123456789012:role/myRedshiftRole",
        "ApplyStatus": "in-sync"
      }
    ],
    "MaintenanceTrackName": "current",
    "DeferredMaintenanceWindows": [],
    "NextMaintenanceWindowStartTime": "2019-12-14T16:00:00Z"
  }
}
```

For more information, see [Amazon Redshift Database Encryption](#) in the *Amazon Redshift Cluster Management Guide*.

- For API details, see [RotateEncryptionKey](#) in *AWS CLI Command Reference*.

Amazon Rekognition examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon Rekognition.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

compare-faces

The following code example shows how to use compare-faces.

For more information, see [Comparing faces in images](#).

AWS CLI

To compare faces in two images

The following compare-faces command compares faces in two images stored in an Amazon S3 bucket.

```
aws rekognition compare-faces \  
  --source-image '{"S3object":{"Bucket":"MyImageS3Bucket","Name":"source.jpg"}}' \  
  --target-image '{"S3object":{"Bucket":"MyImageS3Bucket","Name":"target.jpg"}}'
```

Output:

```
{  
  "UnmatchedFaces": [],  
  "FaceMatches": [  
    {  
      "Face": {  
        "BoundingBox": {  
          "Width": 0.12368916720151901,  
          "Top": 0.16007372736930847,  
          "Left": 0.5901257991790771,  
          "Height": 0.25140416622161865  
        },  
        "Confidence": 100.0,  
        "Pose": {  
          "Yaw": -3.7351467609405518,  
          "Roll": -0.10309021919965744,  
          "Pitch": 0.8637830018997192  
        },  
        "Quality": {  
          "Sharpness": 95.51618957519531,  
          "Brightness": 65.29893493652344  
        },  
        "Landmarks": [  
          {
```

```

        "Y": 0.26721030473709106,
        "X": 0.6204193830490112,
        "Type": "eyeLeft"
    },
    {
        "Y": 0.26831310987472534,
        "X": 0.6776827573776245,
        "Type": "eyeRight"
    },
    {
        "Y": 0.3514654338359833,
        "X": 0.6241428852081299,
        "Type": "mouthLeft"
    },
    {
        "Y": 0.35258132219314575,
        "X": 0.6713621020317078,
        "Type": "mouthRight"
    },
    {
        "Y": 0.3140771687030792,
        "X": 0.6428444981575012,
        "Type": "nose"
    }
]
},
"Similarity": 100.0
}
],
"SourceImageFace": {
    "BoundingBox": {
        "Width": 0.12368916720151901,
        "Top": 0.16007372736930847,
        "Left": 0.5901257991790771,
        "Height": 0.25140416622161865
    },
    "Confidence": 100.0
}
}

```

For more information, see [Comparing Faces in Images](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [CompareFaces](#) in *AWS CLI Command Reference*.

create-collection

The following code example shows how to use `create-collection`.

For more information, see [Creating a collection](#).

AWS CLI

To create a collection

The following `create-collection` command creates a collection with the specified name.

```
aws rekognition create-collection \  
  --collection-id "MyCollection"
```

Output:

```
{  
  "CollectionArn": "aws:rekognition:us-west-2:123456789012:collection/  
MyCollection",  
  "FaceModelVersion": "4.0",  
  "StatusCode": 200  
}
```

For more information, see [Creating a Collection](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [CreateCollection](#) in *AWS CLI Command Reference*.

create-stream-processor

The following code example shows how to use `create-stream-processor`.

AWS CLI

To create a new stream processor

The following `create-stream-processor` example creates a new stream processor with the specified configuration.

```
aws rekognition create-stream-processor --name my-stream-processor\  
  --input '{"KinesisVideoStream":{"Arn":"arn:aws:kinesisvideo:us-  
west-2:123456789012:stream/macwebcam/1530559711205"}}'\
```



```
--stream-processor-output '{"KinesisDataStream":{"Arn":"arn:aws:kinesis:us-  
west-2:123456789012:stream/AmazonRekognitionRekStream"}}'\  
--role-arn arn:aws:iam::123456789012:role/AmazonRekognitionDetect\  
--settings '{"FaceSearch":  
{"CollectionId":"MyCollection","FaceMatchThreshold":85.5}}'
```

Output:

```
{  
  "StreamProcessorArn": "arn:aws:rekognition:us-  
west-2:123456789012:streamprocessor/my-stream-processor"  
}
```

For more information, see [Working with Streaming Videos](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [CreateStreamProcessor](#) in *AWS CLI Command Reference*.

delete-collection

The following code example shows how to use delete-collection.

For more information, see [Deleting a collection](#).

AWS CLI

To delete a collection

The following delete-collection command deletes the specified collection.

```
aws rekognition delete-collection \  
  --collection-id MyCollection
```

Output:

```
{  
  "StatusCode": 200  
}
```

For more information, see [Deleting a Collection](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [DeleteCollection](#) in *AWS CLI Command Reference*.

delete-faces

The following code example shows how to use delete-faces.

For more information, see [Deleting faces from a collection](#).

AWS CLI

To delete faces from a collection

The following delete-faces command deletes the specified face from a collection.

```
aws rekognition delete-faces \  
  --collection-id MyCollection \  
  --face-ids '["0040279c-0178-436e-b70a-e61b074e96b0"]'
```

Output:

```
{  
  "DeletedFaces": [  
    "0040279c-0178-436e-b70a-e61b074e96b0"  
  ]  
}
```

For more information, see [Deleting Faces from a Collection](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [DeleteFaces](#) in *AWS CLI Command Reference*.

delete-stream-processor

The following code example shows how to use delete-stream-processor.

AWS CLI

To delete a stream processor

The following delete-stream-processor command deletes the specified stream processor.

```
aws rekognition delete-stream-processor \  
  --stream-processor-name MyStreamProcessor
```

```
--name my-stream-processor
```

This command produces no output.

For more information, see [Working with Streaming Videos](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [DeleteStreamProcessor](#) in *AWS CLI Command Reference*.

describe-collection

The following code example shows how to use describe-collection.

For more information, see [Describing a collection](#).

AWS CLI

To describe a collection

The following describe-collection example displays the details about the specified collection.

```
aws rekognition describe-collection \  
  --collection-id MyCollection
```

Output:

```
{  
  "FaceCount": 200,  
  "CreationTimestamp": 1569444828.274,  
  "CollectionARN": "arn:aws:rekognition:us-west-2:123456789012:collection/  
MyCollection",  
  "FaceModelVersion": "4.0"  
}
```

For more information, see [Describing a Collection](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [DescribeCollection](#) in *AWS CLI Command Reference*.

describe-stream-processor

The following code example shows how to use describe-stream-processor.

AWS CLI

To get information about a stream processor

The following `describe-stream-processor` command displays details about the specified stream processor.

```
aws rekognition describe-stream-processor \  
  --name my-stream-processor
```

Output:

```
{  
  "Status": "STOPPED",  
  "Name": "my-stream-processor",  
  "LastUpdateTimestamp": 1532449292.712,  
  "Settings": {  
    "FaceSearch": {  
      "FaceMatchThreshold": 80.0,  
      "CollectionId": "my-collection"  
    }  
  },  
  "RoleArn": "arn:aws:iam::123456789012:role/AmazonRekognitionDetectStream",  
  "StreamProcessorArn": "arn:aws:rekognition:us-west-2:123456789012:streamprocessor/my-stream-processor",  
  "Output": {  
    "KinesisDataStream": {  
      "Arn": "arn:aws:kinesis:us-west-2:123456789012:stream/AmazonRekognitionRekStream"  
    }  
  },  
  "Input": {  
    "KinesisVideoStream": {  
      "Arn": "arn:aws:kinesisvideo:us-west-2:123456789012:stream/macwebcam/123456789012"  
    }  
  },  
  "CreationTimestamp": 1532449292.712  
}
```

For more information, see [Working with Streaming Videos](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [DescribeStreamProcessor](#) in *AWS CLI Command Reference*.

detect-faces

The following code example shows how to use detect-faces.

For more information, see [Detecting faces in an image](#).

AWS CLI

To detect faces in an image

The following detect-faces command detects faces in the specified image stored in an Amazon S3 bucket.

```
aws rekognition detect-faces \  
  --image '{"S3Object":{"Bucket":"MyImageS3Bucket","Name":"MyFriend.jpg"}}' \  
  --attributes "ALL"
```

Output:

```
{  
  "FaceDetails": [  
    {  
      "Confidence": 100.0,  
      "Eyeglasses": {  
        "Confidence": 98.91107940673828,  
        "Value": false  
      },  
      "Sunglasses": {  
        "Confidence": 99.7966537475586,  
        "Value": false  
      },  
      "Gender": {  
        "Confidence": 99.56611633300781,  
        "Value": "Male"  
      },  
      "Landmarks": [  
        {  
          "Y": 0.26721030473709106,  
          "X": 0.6204193830490112,  
          "Type": "eyeLeft"  
        },  
      ],  
    },  
  ],  
}
```

```
{
  "Y": 0.26831310987472534,
  "X": 0.6776827573776245,
  "Type": "eyeRight"
},
{
  "Y": 0.3514654338359833,
  "X": 0.6241428852081299,
  "Type": "mouthLeft"
},
{
  "Y": 0.35258132219314575,
  "X": 0.6713621020317078,
  "Type": "mouthRight"
},
{
  "Y": 0.3140771687030792,
  "X": 0.6428444981575012,
  "Type": "nose"
},
{
  "Y": 0.24662546813488007,
  "X": 0.6001564860343933,
  "Type": "leftEyeBrowLeft"
},
{
  "Y": 0.24326619505882263,
  "X": 0.6303644776344299,
  "Type": "leftEyeBrowRight"
},
{
  "Y": 0.23818562924861908,
  "X": 0.6146903038024902,
  "Type": "leftEyeBrowUp"
},
{
  "Y": 0.24373626708984375,
  "X": 0.6640064716339111,
  "Type": "rightEyeBrowLeft"
},
{
  "Y": 0.24877218902111053,
  "X": 0.7025929093360901,
  "Type": "rightEyeBrowRight"
}
```

```
    },  
    {  
      "Y": 0.23938551545143127,  
      "X": 0.6823262572288513,  
      "Type": "rightEyeBrowUp"  
    },  
    {  
      "Y": 0.265746533870697,  
      "X": 0.6112898588180542,  
      "Type": "leftEyeLeft"  
    },  
    {  
      "Y": 0.2676128149032593,  
      "X": 0.6317071914672852,  
      "Type": "leftEyeRight"  
    },  
    {  
      "Y": 0.262735515832901,  
      "X": 0.6201658248901367,  
      "Type": "leftEyeUp"  
    },  
    {  
      "Y": 0.27025148272514343,  
      "X": 0.6206279993057251,  
      "Type": "leftEyeDown"  
    },  
    {  
      "Y": 0.268223375082016,  
      "X": 0.6658390760421753,  
      "Type": "rightEyeLeft"  
    },  
    {  
      "Y": 0.2672517001628876,  
      "X": 0.687832236289978,  
      "Type": "rightEyeRight"  
    },  
    {  
      "Y": 0.26383838057518005,  
      "X": 0.6769183874130249,  
      "Type": "rightEyeUp"  
    },  
    {  
      "Y": 0.27138751745224,  
      "X": 0.676596462726593,
```

```
    "Type": "rightEyeDown"
  },
  {
    "Y": 0.32283174991607666,
    "X": 0.6350004076957703,
    "Type": "noseLeft"
  },
  {
    "Y": 0.3219289481639862,
    "X": 0.6567046642303467,
    "Type": "noseRight"
  },
  {
    "Y": 0.3420318365097046,
    "X": 0.6450609564781189,
    "Type": "mouthUp"
  },
  {
    "Y": 0.3664324879646301,
    "X": 0.6455618143081665,
    "Type": "mouthDown"
  },
  {
    "Y": 0.26721030473709106,
    "X": 0.6204193830490112,
    "Type": "leftPupil"
  },
  {
    "Y": 0.26831310987472534,
    "X": 0.6776827573776245,
    "Type": "rightPupil"
  },
  {
    "Y": 0.26343393325805664,
    "X": 0.5946047306060791,
    "Type": "upperJawlineLeft"
  },
  {
    "Y": 0.3543180525302887,
    "X": 0.6044883728027344,
    "Type": "midJawlineLeft"
  },
  {
    "Y": 0.4084877669811249,
```



```
        "X": 0.6477024555206299,
        "Type": "chinBottom"
    },
    {
        "Y": 0.3562754988670349,
        "X": 0.707981526851654,
        "Type": "midJawlineRight"
    },
    {
        "Y": 0.26580461859703064,
        "X": 0.7234612107276917,
        "Type": "upperJawlineRight"
    }
],
"Pose": {
    "Yaw": -3.7351467609405518,
    "Roll": -0.10309021919965744,
    "Pitch": 0.8637830018997192
},
"Emotions": [
    {
        "Confidence": 8.74203109741211,
        "Type": "SURPRISED"
    },
    {
        "Confidence": 2.501944065093994,
        "Type": "ANGRY"
    },
    {
        "Confidence": 0.7378743290901184,
        "Type": "DISGUSTED"
    },
    {
        "Confidence": 3.5296201705932617,
        "Type": "HAPPY"
    },
    {
        "Confidence": 1.7162904739379883,
        "Type": "SAD"
    },
    {
        "Confidence": 9.518536567687988,
        "Type": "CONFUSED"
    }
],
```

```
    {
      "Confidence": 0.45474427938461304,
      "Type": "FEAR"
    },
    {
      "Confidence": 72.79895782470703,
      "Type": "CALM"
    }
  ],
  "AgeRange": {
    "High": 48,
    "Low": 32
  },
  "EyesOpen": {
    "Confidence": 98.93987274169922,
    "Value": true
  },
  "BoundingBox": {
    "Width": 0.12368916720151901,
    "Top": 0.16007372736930847,
    "Left": 0.5901257991790771,
    "Height": 0.25140416622161865
  },
  "Smile": {
    "Confidence": 93.4493179321289,
    "Value": false
  },
  "MouthOpen": {
    "Confidence": 90.53053283691406,
    "Value": false
  },
  "Quality": {
    "Sharpness": 95.51618957519531,
    "Brightness": 65.29893493652344
  },
  "Mustache": {
    "Confidence": 89.85221099853516,
    "Value": false
  },
  "Beard": {
    "Confidence": 86.1991195678711,
    "Value": true
  }
}
```

```
]
}
```

For more information, see [Detecting Faces in an Image](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [DetectFaces](#) in *AWS CLI Command Reference*.

detect-labels

The following code example shows how to use `detect-labels`.

For more information, see [Detecting labels in an image](#).

AWS CLI

To detect a label in an image

The following `detect-labels` example detects scenes and objects in an image stored in an Amazon S3 bucket.

```
aws rekognition detect-labels \  
  --image '{"S3Object":{"Bucket":"bucket","Name":"image"}}'
```

Output:

```
{  
  "Labels": [  
    {  
      "Instances": [],  
      "Confidence": 99.15271759033203,  
      "Parents": [  
        {  
          "Name": "Vehicle"  
        },  
        {  
          "Name": "Transportation"  
        }  
      ],  
      "Name": "Automobile"  
    },  
    {  
      "Instances": [],
```

```
"Confidence": 99.15271759033203,
"Parents": [
  {
    "Name": "Transportation"
  }
],
"Name": "Vehicle"
},
{
  "Instances": [],
  "Confidence": 99.15271759033203,
  "Parents": [],
  "Name": "Transportation"
},
{
  "Instances": [
    {
      "BoundingBox": {
        "Width": 0.10616336017847061,
        "Top": 0.5039216876029968,
        "Left": 0.0037978808395564556,
        "Height": 0.18528179824352264
      },
      "Confidence": 99.15271759033203
    },
    {
      "BoundingBox": {
        "Width": 0.2429988533258438,
        "Top": 0.5251884460449219,
        "Left": 0.7309805154800415,
        "Height": 0.21577216684818268
      },
      "Confidence": 99.1286392211914
    },
    {
      "BoundingBox": {
        "Width": 0.14233611524105072,
        "Top": 0.5333095788955688,
        "Left": 0.6494812965393066,
        "Height": 0.15528248250484467
      },
      "Confidence": 98.48368072509766
    }
  ]
}
```

```
    "BoundingBox": {
      "Width": 0.11086395382881165,
      "Top": 0.5354844927787781,
      "Left": 0.10355594009160995,
      "Height": 0.10271988064050674
    },
    "Confidence": 96.45606231689453
  },
  {
    "BoundingBox": {
      "Width": 0.06254628300666809,
      "Top": 0.5573825240135193,
      "Left": 0.46083059906959534,
      "Height": 0.053911514580249786
    },
    "Confidence": 93.65448760986328
  },
  {
    "BoundingBox": {
      "Width": 0.10105438530445099,
      "Top": 0.534368634223938,
      "Left": 0.5743985772132874,
      "Height": 0.12226245552301407
    },
    "Confidence": 93.06217193603516
  },
  {
    "BoundingBox": {
      "Width": 0.056389667093753815,
      "Top": 0.5235804319381714,
      "Left": 0.9427769780158997,
      "Height": 0.17163699865341187
    },
    "Confidence": 92.6864013671875
  },
  {
    "BoundingBox": {
      "Width": 0.06003860384225845,
      "Top": 0.5441341400146484,
      "Left": 0.22409997880458832,
      "Height": 0.06737709045410156
    },
    "Confidence": 90.4227066040039
  },
}
```

```
{
  "BoundingBox": {
    "Width": 0.02848697081208229,
    "Top": 0.5107086896896362,
    "Left": 0,
    "Height": 0.19150497019290924
  },
  "Confidence": 86.65286254882812
},
{
  "BoundingBox": {
    "Width": 0.04067881405353546,
    "Top": 0.5566273927688599,
    "Left": 0.316415935754776,
    "Height": 0.03428703173995018
  },
  "Confidence": 85.36471557617188
},
{
  "BoundingBox": {
    "Width": 0.043411049991846085,
    "Top": 0.5394920110702515,
    "Left": 0.18293385207653046,
    "Height": 0.0893595889210701
  },
  "Confidence": 82.21705627441406
},
{
  "BoundingBox": {
    "Width": 0.031183116137981415,
    "Top": 0.5579366683959961,
    "Left": 0.2853088080883026,
    "Height": 0.03989990055561066
  },
  "Confidence": 81.0157470703125
},
{
  "BoundingBox": {
    "Width": 0.031113790348172188,
    "Top": 0.5504819750785828,
    "Left": 0.2580395042896271,
    "Height": 0.056484755128622055
  },
  "Confidence": 56.13441467285156
}
```

```
    },
    {
      "BoundingBox": {
        "Width": 0.08586374670267105,
        "Top": 0.5438792705535889,
        "Left": 0.5128012895584106,
        "Height": 0.08550430089235306
      },
      "Confidence": 52.37760925292969
    }
  ],
  "Confidence": 99.15271759033203,
  "Parents": [
    {
      "Name": "Vehicle"
    },
    {
      "Name": "Transportation"
    }
  ],
  "Name": "Car"
},
{
  "Instances": [],
  "Confidence": 98.9914321899414,
  "Parents": [],
  "Name": "Human"
},
{
  "Instances": [
    {
      "BoundingBox": {
        "Width": 0.19360728561878204,
        "Top": 0.35072067379951477,
        "Left": 0.43734854459762573,
        "Height": 0.2742200493812561
      },
      "Confidence": 98.9914321899414
    },
    {
      "BoundingBox": {
        "Width": 0.03801717236638069,
        "Top": 0.5010883808135986,
        "Left": 0.9155802130699158,
```

```
        "Height": 0.06597328186035156
      },
      "Confidence": 85.02790832519531
    }
  ],
  "Confidence": 98.9914321899414,
  "Parents": [],
  "Name": "Person"
},
{
  "Instances": [],
  "Confidence": 93.24951934814453,
  "Parents": [],
  "Name": "Machine"
},
{
  "Instances": [
    {
      "BoundingBox": {
        "Width": 0.03561960905790329,
        "Top": 0.6468243598937988,
        "Left": 0.7850857377052307,
        "Height": 0.08878646790981293
      },
      "Confidence": 93.24951934814453
    },
    {
      "BoundingBox": {
        "Width": 0.02217046171426773,
        "Top": 0.6149078607559204,
        "Left": 0.04757237061858177,
        "Height": 0.07136218994855881
      },
      "Confidence": 91.5025863647461
    },
    {
      "BoundingBox": {
        "Width": 0.016197510063648224,
        "Top": 0.6274210214614868,
        "Left": 0.6472989320755005,
        "Height": 0.04955997318029404
      },
      "Confidence": 85.14686584472656
    }
  ],
}
```



```
{
  "BoundingBox": {
    "Width": 0.020207518711686134,
    "Top": 0.6348286867141724,
    "Left": 0.7295016646385193,
    "Height": 0.07059963047504425
  },
  "Confidence": 83.34547424316406
},
{
  "BoundingBox": {
    "Width": 0.020280985161662102,
    "Top": 0.6171894669532776,
    "Left": 0.08744934946298599,
    "Height": 0.05297485366463661
  },
  "Confidence": 79.9981460571289
},
{
  "BoundingBox": {
    "Width": 0.018318990245461464,
    "Top": 0.623889148235321,
    "Left": 0.6836880445480347,
    "Height": 0.06730121374130249
  },
  "Confidence": 78.87144470214844
},
{
  "BoundingBox": {
    "Width": 0.021310249343514442,
    "Top": 0.6167286038398743,
    "Left": 0.004064912907779217,
    "Height": 0.08317798376083374
  },
  "Confidence": 75.89361572265625
},
{
  "BoundingBox": {
    "Width": 0.03604431077837944,
    "Top": 0.7030032277107239,
    "Left": 0.9254803657531738,
    "Height": 0.04569442570209503
  },
  "Confidence": 64.402587890625
}
```

```
    },
    {
      "BoundingBox": {
        "Width": 0.009834849275648594,
        "Top": 0.5821820497512817,
        "Left": 0.28094568848609924,
        "Height": 0.01964157074689865
      },
      "Confidence": 62.79907989501953
    },
    {
      "BoundingBox": {
        "Width": 0.01475677452981472,
        "Top": 0.6137543320655823,
        "Left": 0.5950819253921509,
        "Height": 0.039063986390829086
      },
      "Confidence": 59.40483474731445
    }
  ],
  "Confidence": 93.24951934814453,
  "Parents": [
    {
      "Name": "Machine"
    }
  ],
  "Name": "Wheel"
},
{
  "Instances": [],
  "Confidence": 92.61514282226562,
  "Parents": [],
  "Name": "Road"
},
{
  "Instances": [],
  "Confidence": 92.37877655029297,
  "Parents": [
    {
      "Name": "Person"
    }
  ],
  "Name": "Sport"
},
```

```
{
  "Instances": [],
  "Confidence": 92.37877655029297,
  "Parents": [
    {
      "Name": "Person"
    }
  ],
  "Name": "Sports"
},
{
  "Instances": [
    {
      "BoundingBox": {
        "Width": 0.12326609343290329,
        "Top": 0.6332163214683533,
        "Left": 0.44815489649772644,
        "Height": 0.058117982000112534
      },
      "Confidence": 92.37877655029297
    }
  ],
  "Confidence": 92.37877655029297,
  "Parents": [
    {
      "Name": "Person"
    },
    {
      "Name": "Sport"
    }
  ],
  "Name": "Skateboard"
},
{
  "Instances": [],
  "Confidence": 90.62931060791016,
  "Parents": [
    {
      "Name": "Person"
    }
  ],
  "Name": "Pedestrian"
},
{
```

```
    "Instances": [],
    "Confidence": 88.81334686279297,
    "Parents": [],
    "Name": "Asphalt"
  },
  {
    "Instances": [],
    "Confidence": 88.81334686279297,
    "Parents": [],
    "Name": "Tarmac"
  },
  {
    "Instances": [],
    "Confidence": 88.23201751708984,
    "Parents": [],
    "Name": "Path"
  },
  {
    "Instances": [],
    "Confidence": 80.26520538330078,
    "Parents": [],
    "Name": "Urban"
  },
  {
    "Instances": [],
    "Confidence": 80.26520538330078,
    "Parents": [
      {
        "Name": "Building"
      },
      {
        "Name": "Urban"
      }
    ],
    "Name": "Town"
  },
  {
    "Instances": [],
    "Confidence": 80.26520538330078,
    "Parents": [],
    "Name": "Building"
  },
  {
    "Instances": [],
```

```
"Confidence": 80.26520538330078,
"Parents": [
  {
    "Name": "Building"
  },
  {
    "Name": "Urban"
  }
],
"Name": "City"
},
{
  "Instances": [],
  "Confidence": 78.37934875488281,
  "Parents": [
    {
      "Name": "Car"
    },
    {
      "Name": "Vehicle"
    },
    {
      "Name": "Transportation"
    }
  ],
  "Name": "Parking Lot"
},
{
  "Instances": [],
  "Confidence": 78.37934875488281,
  "Parents": [
    {
      "Name": "Car"
    },
    {
      "Name": "Vehicle"
    },
    {
      "Name": "Transportation"
    }
  ],
  "Name": "Parking"
},
{
```

```
"Instances": [],
"Confidence": 74.37590026855469,
"Parents": [
  {
    "Name": "Building"
  },
  {
    "Name": "Urban"
  },
  {
    "Name": "City"
  }
],
"Name": "Downtown"
},
{
  "Instances": [],
  "Confidence": 69.84622955322266,
  "Parents": [
    {
      "Name": "Road"
    }
  ],
  "Name": "Intersection"
},
{
  "Instances": [],
  "Confidence": 57.68518829345703,
  "Parents": [
    {
      "Name": "Sports Car"
    },
    {
      "Name": "Car"
    },
    {
      "Name": "Vehicle"
    },
    {
      "Name": "Transportation"
    }
  ],
  "Name": "Coupe"
},
```

```
{
  "Instances": [],
  "Confidence": 57.68518829345703,
  "Parents": [
    {
      "Name": "Car"
    },
    {
      "Name": "Vehicle"
    },
    {
      "Name": "Transportation"
    }
  ],
  "Name": "Sports Car"
},
{
  "Instances": [],
  "Confidence": 56.59492111206055,
  "Parents": [
    {
      "Name": "Path"
    }
  ],
  "Name": "Sidewalk"
},
{
  "Instances": [],
  "Confidence": 56.59492111206055,
  "Parents": [
    {
      "Name": "Path"
    }
  ],
  "Name": "Pavement"
},
{
  "Instances": [],
  "Confidence": 55.58770751953125,
  "Parents": [
    {
      "Name": "Building"
    },
    {
```

```
        "Name": "Urban"
      }
    ],
    "Name": "Neighborhood"
  }
],
"LabelModelVersion": "2.0"
}
```

For more information, see [Detecting Labels in an Image](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [DetectLabels](#) in *AWS CLI Command Reference*.

detect-moderation-labels

The following code example shows how to use `detect-moderation-labels`.

For more information, see [Detecting inappropriate images](#).

AWS CLI

To detect unsafe content in an image

The following `detect-moderation-labels` command detects unsafe content in the specified image stored in an Amazon S3 bucket.

```
aws rekognition detect-moderation-labels \
  --image "S3Object={Bucket=MyImageS3Bucket,Name=gun.jpg}"
```

Output:

```
{
  "ModerationModelVersion": "3.0",
  "ModerationLabels": [
    {
      "Confidence": 97.29618072509766,
      "ParentName": "Violence",
      "Name": "Weapon Violence"
    },
    {
      "Confidence": 97.29618072509766,
      "ParentName": "",

```



```

        "Name": "Violence"
      }
    ]
  }

```

For more information, see [Detecting Unsafe Images](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [DetectModerationLabels](#) in *AWS CLI Command Reference*.

detect-text

The following code example shows how to use `detect-text`.

For more information, see [Detecting text in an image](#).

AWS CLI

To detect text in an image

The following `detect-text` command detects text in the specified image.

```

aws rekognition detect-text \
  --image '{"S3Object":{"Bucket":"MyImageS3Bucket","Name":"ExamplePicture.jpg"}}'

```

Output:

```

{
  "TextDetections": [
    {
      "Geometry": {
        "BoundingBox": {
          "Width": 0.24624845385551453,
          "Top": 0.28288066387176514,
          "Left": 0.391388863325119,
          "Height": 0.022687450051307678
        },
        "Polygon": [
          {
            "Y": 0.28288066387176514,
            "X": 0.391388863325119
          },
          {
            "Y": 0.2826388478279114,

```

```
        "X": 0.6376373171806335
      },
      {
        "Y": 0.30532628297805786,
        "X": 0.637677013874054
      },
      {
        "Y": 0.305568128824234,
        "X": 0.39142853021621704
      }
    ]
  },
  "Confidence": 94.35709381103516,
  "DetectedText": "ESTD 1882",
  "Type": "LINE",
  "Id": 0
},
{
  "Geometry": {
    "BoundingBox": {
      "Width": 0.33933889865875244,
      "Top": 0.32603850960731506,
      "Left": 0.34534579515457153,
      "Height": 0.07126858830451965
    },
    "Polygon": [
      {
        "Y": 0.32603850960731506,
        "X": 0.34534579515457153
      },
      {
        "Y": 0.32633158564567566,
        "X": 0.684684693813324
      },
      {
        "Y": 0.3976001739501953,
        "X": 0.684575080871582
      },
      {
        "Y": 0.3973070979118347,
        "X": 0.345236212015152
      }
    ]
  }
},
```

```
"Confidence": 99.95779418945312,
"DetectedText": "BRAINS",
"Type": "LINE",
"Id": 1
},
{
"Confidence": 97.22098541259766,
"Geometry": {
  "BoundingBox": {
    "Width": 0.061079490929841995,
    "Top": 0.2843210697174072,
    "Left": 0.391391396522522,
    "Height": 0.021029088646173477
  },
  "Polygon": [
    {
      "Y": 0.2843210697174072,
      "X": 0.391391396522522
    },
    {
      "Y": 0.2828207015991211,
      "X": 0.4524524509906769
    },
    {
      "Y": 0.3038259446620941,
      "X": 0.4534534513950348
    },
    {
      "Y": 0.30532634258270264,
      "X": 0.3923923969268799
    }
  ]
},
"DetectedText": "ESTD",
"ParentId": 0,
"Type": "WORD",
"Id": 2
},
{
"Confidence": 91.49320983886719,
"Geometry": {
  "BoundingBox": {
    "Width": 0.07007007300853729,
    "Top": 0.2828207015991211,
```

```
        "Left": 0.5675675868988037,
        "Height": 0.02250562608242035
    },
    "Polygon": [
        {
            "Y": 0.2828207015991211,
            "X": 0.5675675868988037
        },
        {
            "Y": 0.2828207015991211,
            "X": 0.6376376152038574
        },
        {
            "Y": 0.30532634258270264,
            "X": 0.6376376152038574
        },
        {
            "Y": 0.30532634258270264,
            "X": 0.5675675868988037
        }
    ]
},
"DetectedText": "1882",
"ParentId": 0,
"Type": "WORD",
"Id": 3
},
{
    "Confidence": 99.95779418945312,
    "Geometry": {
        "BoundingBox": {
            "Width": 0.33933934569358826,
            "Top": 0.32633158564567566,
            "Left": 0.3453453481197357,
            "Height": 0.07127484679222107
        },
        "Polygon": [
            {
                "Y": 0.32633158564567566,
                "X": 0.3453453481197357
            },
            {
                "Y": 0.32633158564567566,
                "X": 0.684684693813324
            }
        ]
    }
}
```

```
    },
    {
      "Y": 0.39759939908981323,
      "X": 0.6836836934089661
    },
    {
      "Y": 0.39684921503067017,
      "X": 0.3453453481197357
    }
  ]
},
"DetectedText": "BRAINS",
"ParentId": 1,
"Type": "WORD",
"Id": 4
}
]
```

- For API details, see [DetectText](#) in *AWS CLI Command Reference*.

disassociate-faces

The following code example shows how to use `disassociate-faces`.

AWS CLI

```
aws rekognition disassociate-faces --face-ids list-of-face-ids
  --user-id user-id --collection-id collection-name --region region-name
```

- For API details, see [DisassociateFaces](#) in *AWS CLI Command Reference*.

get-celebrity-info

The following code example shows how to use `get-celebrity-info`.

AWS CLI

To get information about a celebrity

The following `get-celebrity-info` command displays information about the specified celebrity. The `id` parameter comes from a previous call to `recognize-celebrities`.

```
aws rekognition get-celebrity-info --id nnnnnnn
```

Output:

```
{
  "Name": "Celeb A",
  "Urls": [
    "www.imdb.com/name/aaaaaaaaa"
  ]
}
```

For more information, see [Getting Information About a Celebrity](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [GetCelebrityInfo](#) in *AWS CLI Command Reference*.

get-celebrity-recognition

The following code example shows how to use `get-celebrity-recognition`.

AWS CLI

To get the results of a celebrity recognition operation

The following `get-celebrity-recognition` command displays the results of a celebrity recognition operation that you started previously by calling `start-celebrity-recognition`.

```
aws rekognition get-celebrity-recognition \
  --job-id 1234567890abcdef1234567890abcdef1234567890abcdef
```

Output:

```
{
  "NextToken": "3D01Clx1CiT31VsRDkA03IybLb/h5AtDWSGuhYi
+N1FIJwwPtAkuKzDhL2rV3GcwmNt77+12",
  "Celebrities": [
    {
```

```
"Timestamp": 0,
"Celebrity": {
  "Confidence": 96.0,
  "Face": {
    "BoundingBox": {
      "Width": 0.70333331823349,
      "Top": 0.16750000417232513,
      "Left": 0.19555555284023285,
      "Height": 0.3956249952316284
    },
    "Landmarks": [
      {
        "Y": 0.31031012535095215,
        "X": 0.441436767578125,
        "Type": "eyeLeft"
      },
      {
        "Y": 0.3081788718700409,
        "X": 0.6437258720397949,
        "Type": "eyeRight"
      },
      {
        "Y": 0.39542075991630554,
        "X": 0.5572493076324463,
        "Type": "nose"
      },
      {
        "Y": 0.4597957134246826,
        "X": 0.4579732120037079,
        "Type": "mouthLeft"
      },
      {
        "Y": 0.45688048005104065,
        "X": 0.6349081993103027,
        "Type": "mouthRight"
      }
    ],
    "Pose": {
      "Yaw": 8.943398475646973,
      "Roll": -2.0309247970581055,
      "Pitch": -0.5674862861633301
    },
    "Quality": {
      "Sharpness": 99.40211486816406,
```

```
        "Brightness": 89.47132110595703
      },
      "Confidence": 99.99861145019531
    },
    "Name": "CelebrityA",
    "Urls": [
      "www.imdb.com/name/1111111111"
    ],
    "Id": "nnnnnn"
  }
},
{
  "Timestamp": 467,
  "Celebrity": {
    "Confidence": 99.0,
    "Face": {
      "BoundingBox": {
        "Width": 0.6877777576446533,
        "Top": 0.18437500298023224,
        "Left": 0.20555555820465088,
        "Height": 0.3868750035762787
      },
      "Landmarks": [
        {
          "Y": 0.31895750761032104,
          "X": 0.4411413371562958,
          "Type": "eyeLeft"
        },
        {
          "Y": 0.3140959143638611,
          "X": 0.6523157954216003,
          "Type": "eyeRight"
        },
        {
          "Y": 0.4016456604003906,
          "X": 0.5682755708694458,
          "Type": "nose"
        },
        {
          "Y": 0.46894142031669617,
          "X": 0.4597797095775604,
          "Type": "mouthLeft"
        }
      ]
    }
  }
}
```



```

        "Y": 0.46971091628074646,
        "X": 0.6286435127258301,
        "Type": "mouthRight"
      }
    ],
    "Pose": {
      "Yaw": 10.433465957641602,
      "Roll": -3.347442388534546,
      "Pitch": 1.3709543943405151
    },
    "Quality": {
      "Sharpness": 99.5531005859375,
      "Brightness": 88.5764389038086
    },
    "Confidence": 99.99148559570312
  },
  "Name": "Jane Celebrity",
  "Urls": [
    "www.imdb.com/name/1111111111"
  ],
  "Id": "nnnnnn"
}
}
],
"JobStatus": "SUCCEEDED",
"VideoMetadata": {
  "Format": "QuickTime / MOV",
  "FrameRate": 29.978118896484375,
  "Codec": "h264",
  "DurationMillis": 4570,
  "FrameHeight": 1920,
  "FrameWidth": 1080
}
}
}

```

For more information, see [Recognizing Celebrities in a Stored Video](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [GetCelebrityRecognition](#) in *AWS CLI Command Reference*.

get-content-moderation

The following code example shows how to use `get-content-moderation`.

AWS CLI

To get the results of an unsafe content operation

The following `get-content-moderation` command displays the results of an unsafe content operation that you started previously by calling `start-content-moderation`.

```
aws rekognition get-content-moderation \  
  --job-id 1234567890abcdef1234567890abcdef1234567890abcdef
```

Output:

```
{  
  "NextToken": "dlhcKMHMzpCBGFukz6I03JMcWiJAamCVhXHt3r6b4b5Tfbyw3q7o+Jeezt  
+Zpgf0nW9FCCgQ",  
  "ModerationLabels": [  
    {  
      "Timestamp": 0,  
      "ModerationLabel": {  
        "Confidence": 97.39583587646484,  
        "ParentName": "",  
        "Name": "Violence"  
      }  
    },  
    {  
      "Timestamp": 0,  
      "ModerationLabel": {  
        "Confidence": 97.39583587646484,  
        "ParentName": "Violence",  
        "Name": "Weapon Violence"  
      }  
    }  
  ],  
  "JobStatus": "SUCCEEDED",  
  "VideoMetadata": {  
    "Format": "QuickTime / MOV",  
    "FrameRate": 29.97515869140625,  
    "Codec": "h264",  
    "DurationMillis": 6039,  
    "FrameHeight": 1920,  
    "FrameWidth": 1080  
  }  
}
```

For more information, see [Detecting Unsafe Stored Videos](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [GetContentModeration](#) in *AWS CLI Command Reference*.

get-face-detection

The following code example shows how to use `get-face-detection`.

AWS CLI

To get the results of a face detection operation

The following `get-face-detection` command displays the results of a face detection operation that you started previously by calling `start-face-detection`.

```
aws rekognition get-face-detection \  
  --job-id 1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef
```

Output:

```
{  
  "Faces": [  
    {  
      "Timestamp": 467,  
      "Face": {  
        "BoundingBox": {  
          "Width": 0.1560753583908081,  
          "Top": 0.13555361330509186,  
          "Left": -0.0952017530798912,  
          "Height": 0.6934483051300049  
        },  
        "Landmarks": [  
          {  
            "Y": 0.4013825058937073,  
            "X": -0.041750285774469376,  
            "Type": "eyeLeft"  
          },  
          {  
            "Y": 0.41695496439933777,  
            "X": 0.027979329228401184,  
            "Type": "eyeRight"  
          }  
        ]  
      }  
    ]  
  }
```

```
        {
            "Y": 0.6375303268432617,
            "X": -0.04034662991762161,
            "Type": "mouthLeft"
        },
        {
            "Y": 0.6497718691825867,
            "X": 0.013960429467260838,
            "Type": "mouthRight"
        },
        {
            "Y": 0.5238034129142761,
            "X": 0.008022055961191654,
            "Type": "nose"
        }
    ],
    "Pose": {
        "Yaw": -58.07863998413086,
        "Roll": 1.9384294748306274,
        "Pitch": -24.66305160522461
    },
    "Quality": {
        "Sharpness": 83.14741516113281,
        "Brightness": 25.75942611694336
    },
    "Confidence": 87.7622299194336
}
},
{
    "Timestamp": 967,
    "Face": {
        "BoundingBox": {
            "Width": 0.28559377789497375,
            "Top": 0.19436298310756683,
            "Left": 0.024553587660193443,
            "Height": 0.7216082215309143
        },
        "Landmarks": [
            {
                "Y": 0.4650231599807739,
                "X": 0.16269078850746155,
                "Type": "eyeLeft"
            },
            {
```

```
        "Y": 0.4843238294124603,
        "X": 0.2782580852508545,
        "Type": "eyeRight"
    },
    {
        "Y": 0.71530681848526,
        "X": 0.1741468608379364,
        "Type": "mouthLeft"
    },
    {
        "Y": 0.7310671210289001,
        "X": 0.26857468485832214,
        "Type": "mouthRight"
    },
    {
        "Y": 0.582602322101593,
        "X": 0.2566150426864624,
        "Type": "nose"
    }
],
"Pose": {
    "Yaw": 11.487052917480469,
    "Roll": 5.074230670928955,
    "Pitch": 15.396159172058105
},
"Quality": {
    "Sharpness": 73.32209777832031,
    "Brightness": 54.96497344970703
},
"Confidence": 99.99998474121094
}
],
"NextToken":
"OzL223pDKy91160/02KXRqFIEAwxy4PkgYcm3hSo0rdysbXg5Ex0eFgTGEj0ADEac6S037U",
"JobStatus": "SUCCEEDED",
"VideoMetadata": {
    "Format": "QuickTime / MOV",
    "FrameRate": 29.970617294311523,
    "Codec": "h264",
    "DurationMillis": 6806,
    "FrameHeight": 1080,
    "FrameWidth": 1920
}
```

```
}
```

For more information, see [Detecting Faces in a Stored Video](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [GetFaceDetection](#) in *AWS CLI Command Reference*.

get-face-search

The following code example shows how to use `get-face-search`.

AWS CLI

To get the results of a face search operation

The following `get-face-search` command displays the results of a face search operation that you started previously by calling `start-face-search`.

```
aws rekognition get-face-search \
  --job-id 1234567890abcdef1234567890abcdef1234567890abcdef
```

Output:

```
{
  "Persons": [
    {
      "Timestamp": 467,
      "FaceMatches": [],
      "Person": {
        "Index": 0,
        "Face": {
          "BoundingBox": {
            "Width": 0.1560753583908081,
            "Top": 0.13555361330509186,
            "Left": -0.0952017530798912,
            "Height": 0.6934483051300049
          },
          "Landmarks": [
            {
              "Y": 0.4013825058937073,
              "X": -0.041750285774469376,
              "Type": "eyeLeft"
            }
          ]
        }
      }
    }
  ]
}
```

```
        {
            "Y": 0.41695496439933777,
            "X": 0.027979329228401184,
            "Type": "eyeRight"
        },
        {
            "Y": 0.6375303268432617,
            "X": -0.04034662991762161,
            "Type": "mouthLeft"
        },
        {
            "Y": 0.6497718691825867,
            "X": 0.013960429467260838,
            "Type": "mouthRight"
        },
        {
            "Y": 0.5238034129142761,
            "X": 0.008022055961191654,
            "Type": "nose"
        }
    ],
    "Pose": {
        "Yaw": -58.07863998413086,
        "Roll": 1.9384294748306274,
        "Pitch": -24.66305160522461
    },
    "Quality": {
        "Sharpness": 83.14741516113281,
        "Brightness": 25.75942611694336
    },
    "Confidence": 87.7622299194336
}
},
{
    "Timestamp": 967,
    "FaceMatches": [
        {
            "Face": {
                "BoundingBox": {
                    "Width": 0.12368900328874588,
                    "Top": 0.16007399559020996,
                    "Left": 0.5901259779930115,
                    "Height": 0.2514039874076843
```

```
    },
    "FaceId": "056a95fa-2060-4159-9cab-7ed4daa030fa",
    "ExternalImageId": "image3.jpg",
    "Confidence": 100.0,
    "ImageId": "08f8a078-8929-37fd-8e8f-aadf690e8232"
  },
  "Similarity": 98.44476318359375
}
],
"Person": {
  "Index": 1,
  "Face": {
    "BoundingBox": {
      "Width": 0.28559377789497375,
      "Top": 0.19436298310756683,
      "Left": 0.024553587660193443,
      "Height": 0.7216082215309143
    },
    "Landmarks": [
      {
        "Y": 0.4650231599807739,
        "X": 0.16269078850746155,
        "Type": "eyeLeft"
      },
      {
        "Y": 0.4843238294124603,
        "X": 0.2782580852508545,
        "Type": "eyeRight"
      },
      {
        "Y": 0.71530681848526,
        "X": 0.1741468608379364,
        "Type": "mouthLeft"
      },
      {
        "Y": 0.7310671210289001,
        "X": 0.26857468485832214,
        "Type": "mouthRight"
      },
      {
        "Y": 0.582602322101593,
        "X": 0.2566150426864624,
        "Type": "nose"
      }
    ]
  }
}
```



```

        ],
        "Pose": {
            "Yaw": 11.487052917480469,
            "Roll": 5.074230670928955,
            "Pitch": 15.396159172058105
        },
        "Quality": {
            "Sharpness": 73.32209777832031,
            "Brightness": 54.96497344970703
        },
        "Confidence": 99.99998474121094
    }
}
}
],
"NextToken": "5bkgcezyuaqhtWk3C80TW6cjRghrwV9XDMivm5B3MXm+Lv6G+L+GejyFHPhoNa/
ldXIC4c/d",
"JobStatus": "SUCCEEDED",
"VideoMetadata": {
    "Format": "QuickTime / MOV",
    "FrameRate": 29.970617294311523,
    "Codec": "h264",
    "DurationMillis": 6806,
    "FrameHeight": 1080,
    "FrameWidth": 1920
}
}
}

```

For more information, see [Searching Stored Videos for Faces](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [GetFaceSearch](#) in *AWS CLI Command Reference*.

get-label-detection

The following code example shows how to use `get-label-detection`.

AWS CLI

To get the results of an objects and scenes detection operation

The following `get-label-detection` command displays the results of an objects and scenes detection operation that you started previously by calling `start-label-detection`.

```
aws rekognition get-label-detection \  
  --job-id 1234567890abcdef1234567890abcdef1234567890abcdef
```

Output:

```
{  
  "Labels": [  
    {  
      "Timestamp": 0,  
      "Label": {  
        "Instances": [],  
        "Confidence": 50.19071578979492,  
        "Parents": [  
          {  
            "Name": "Person"  
          },  
          {  
            "Name": "Crowd"  
          }  
        ],  
        "Name": "Audience"  
      }  
    },  
    {  
      "Timestamp": 0,  
      "Label": {  
        "Instances": [],  
        "Confidence": 55.74115753173828,  
        "Parents": [  
          {  
            "Name": "Room"  
          },  
          {  
            "Name": "Indoors"  
          },  
          {  
            "Name": "School"  
          }  
        ],  
        "Name": "Classroom"  
      }  
    }  
  ],  
}
```

```

"JobStatus": "SUCCEEDED",
"LabelModelVersion": "2.0",
"VideoMetadata": {
  "Format": "QuickTime / MOV",
  "FrameRate": 29.970617294311523,
  "Codec": "h264",
  "DurationMillis": 6806,
  "FrameHeight": 1080,
  "FrameWidth": 1920
},
"NextToken": "BMugzAi4L72IERzQdbpyMQuEFBsjl05W0Yx3mfG+sR9mm98E1/
Cp0benspRfs/5FBQFs4X7G"
}

```

For more information, see [Detecting Labels in a Video](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [GetLabelDetection](#) in *AWS CLI Command Reference*.

get-person-tracking

The following code example shows how to use `get-person-tracking`.

AWS CLI

To get the results of a people pathing operation

The following `get-person-tracking` command displays the results of a people pathing operation that you started previously by calling `start-person-tracking`.

```

aws rekognition get-person-tracking \
  --job-id 1234567890abcdef1234567890abcdef1234567890abcdef

```

Output:

```

{
  "Persons": [
    {
      "Timestamp": 500,
      "Person": {
        "BoundingBox": {

```

```

        "Width": 0.4151041805744171,
        "Top": 0.07870370149612427,
        "Left": 0.0,
        "Height": 0.9212962985038757
      },
      "Index": 0
    }
  ],
  {
    "Timestamp": 567,
    "Person": {
      "BoundingBox": {
        "Width": 0.4755208194255829,
        "Top": 0.07777778059244156,
        "Left": 0.0,
        "Height": 0.9194444417953491
      },
      "Index": 0
    }
  }
],
"NextToken": "D/vRIYnyhG79ugdta3f+8cRg9oSro
+HigG0uxRiYpTn0ExnqTi1CJektVAc4HrAXDv25eHYk",
"JobStatus": "SUCCEEDED",
"VideoMetadata": {
  "Format": "QuickTime / MOV",
  "FrameRate": 29.970617294311523,
  "Codec": "h264",
  "DurationMillis": 6806,
  "FrameHeight": 1080,
  "FrameWidth": 1920
}
}

```

For more information, see [People Pathing](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [GetPersonTracking](#) in *AWS CLI Command Reference*.

index-faces

The following code example shows how to use `index-faces`.

For more information, see [Adding faces to a collection](#).

AWS CLI

To add faces to a collection

The following `index-faces` command adds the faces found in an image to the specified collection.

```
aws rekognition index-faces \  
  --image '{"S3Object":{"Bucket":"MyVideoS3Bucket","Name":"MyPicture.jpg"}}' \  
  --collection-id MyCollection \  
  --max-faces 1 \  
  --quality-filter "AUTO" \  
  --detection-attributes "ALL" \  
  --external-image-id "MyPicture.jpg"
```

Output:

```
{  
  "FaceRecords": [  
    {  
      "FaceDetail": {  
        "Confidence": 99.993408203125,  
        "Eyeglasses": {  
          "Confidence": 99.11750030517578,  
          "Value": false  
        },  
        "Sunglasses": {  
          "Confidence": 99.98249053955078,  
          "Value": false  
        },  
        "Gender": {  
          "Confidence": 99.92769622802734,  
          "Value": "Male"  
        },  
        "Landmarks": [  
          {  
            "Y": 0.26750367879867554,  
            "X": 0.6202793717384338,  
            "Type": "eyeLeft"  
          },  
          {  
            "Y": 0.26642778515815735,  
            "X": 0.6787431836128235,  
            "Type": "eyeRight"  
          }  
        ]  
      }  
    }  
  ]  
}
```

```
    "Type": "eyeRight"
  },
  {
    "Y": 0.31361380219459534,
    "X": 0.6421601176261902,
    "Type": "nose"
  },
  {
    "Y": 0.3495299220085144,
    "X": 0.6216195225715637,
    "Type": "mouthLeft"
  },
  {
    "Y": 0.35194727778434753,
    "X": 0.669899046421051,
    "Type": "mouthRight"
  },
  {
    "Y": 0.26844894886016846,
    "X": 0.6210268139839172,
    "Type": "leftPupil"
  },
  {
    "Y": 0.26707562804222107,
    "X": 0.6817160844802856,
    "Type": "rightPupil"
  },
  {
    "Y": 0.24834522604942322,
    "X": 0.6018546223640442,
    "Type": "leftEyeBrowLeft"
  },
  {
    "Y": 0.24397172033786774,
    "X": 0.6172008514404297,
    "Type": "leftEyeBrowUp"
  },
  {
    "Y": 0.24677404761314392,
    "X": 0.6339119076728821,
    "Type": "leftEyeBrowRight"
  },
  {
    "Y": 0.24582654237747192,
```

```
        "X": 0.6619398593902588,  
        "Type": "rightEyeBrowLeft"  
    },  
    {  
        "Y": 0.23973053693771362,  
        "X": 0.6804757118225098,  
        "Type": "rightEyeBrowUp"  
    },  
    {  
        "Y": 0.24441994726657867,  
        "X": 0.6978968977928162,  
        "Type": "rightEyeBrowRight"  
    },  
    {  
        "Y": 0.2695908546447754,  
        "X": 0.6085202693939209,  
        "Type": "leftEyeLeft"  
    },  
    {  
        "Y": 0.26716896891593933,  
        "X": 0.6315826177597046,  
        "Type": "leftEyeRight"  
    },  
    {  
        "Y": 0.26289820671081543,  
        "X": 0.6202316880226135,  
        "Type": "leftEyeUp"  
    },  
    {  
        "Y": 0.27123287320137024,  
        "X": 0.6205548048019409,  
        "Type": "leftEyeDown"  
    },  
    {  
        "Y": 0.2668408751487732,  
        "X": 0.6663622260093689,  
        "Type": "rightEyeLeft"  
    },  
    {  
        "Y": 0.26741549372673035,  
        "X": 0.6910083889961243,  
        "Type": "rightEyeRight"  
    },  
    {
```

```
        "Y": 0.2614026665687561,  
        "X": 0.6785826086997986,  
        "Type": "rightEyeUp"  
    },  
    {  
        "Y": 0.27075251936912537,  
        "X": 0.6789616942405701,  
        "Type": "rightEyeDown"  
    },  
    {  
        "Y": 0.3211299479007721,  
        "X": 0.6324167847633362,  
        "Type": "noseLeft"  
    },  
    {  
        "Y": 0.32276326417922974,  
        "X": 0.6558475494384766,  
        "Type": "noseRight"  
    },  
    {  
        "Y": 0.34385165572166443,  
        "X": 0.6444970965385437,  
        "Type": "mouthUp"  
    },  
    {  
        "Y": 0.3671635091304779,  
        "X": 0.6459195017814636,  
        "Type": "mouthDown"  
    }  
],  
"Pose": {  
    "Yaw": -9.54541015625,  
    "Roll": -0.5709401965141296,  
    "Pitch": 0.6045494675636292  
},  
"Emotions": [  
    {  
        "Confidence": 39.90074157714844,  
        "Type": "HAPPY"  
    },  
    {  
        "Confidence": 23.38753890991211,  
        "Type": "CALM"  
    }  
],
```



```
        {
            "Confidence": 5.840933322906494,
            "Type": "CONFUSED"
        }
    ],
    "AgeRange": {
        "High": 63,
        "Low": 45
    },
    "EyesOpen": {
        "Confidence": 99.80887603759766,
        "Value": true
    },
    "BoundingBox": {
        "Width": 0.18562500178813934,
        "Top": 0.1618015021085739,
        "Left": 0.5575000047683716,
        "Height": 0.24770642817020416
    },
    "Smile": {
        "Confidence": 99.69740295410156,
        "Value": false
    },
    "MouthOpen": {
        "Confidence": 99.97393798828125,
        "Value": false
    },
    "Quality": {
        "Sharpness": 95.54405975341797,
        "Brightness": 63.867706298828125
    },
    "Mustache": {
        "Confidence": 97.05007934570312,
        "Value": false
    },
    "Beard": {
        "Confidence": 87.34505462646484,
        "Value": false
    }
},
"Face": {
    "BoundingBox": {
        "Width": 0.18562500178813934,
        "Top": 0.1618015021085739,
```

```
        "Left": 0.5575000047683716,  
        "Height": 0.24770642817020416  
    },  
    "FaceId": "ce7ed422-2132-4a11-ab14-06c5c410f29f",  
    "ExternalImageId": "example-image.jpg",  
    "Confidence": 99.993408203125,  
    "ImageId": "8d67061e-90d2-598f-9fbd-29c8497039c0"  
  }  
}  
],  
"UnindexedFaces": [],  
"FaceModelVersion": "3.0",  
"OrientationCorrection": "ROTATE_0"  
}
```

For more information, see [Adding Faces to a Collection](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [IndexFaces](#) in *AWS CLI Command Reference*.

list-collections

The following code example shows how to use `list-collections`.

For more information, see [Listing collections](#).

AWS CLI

To list the available collections

The following `list-collections` command lists the available collections in the AWS account.

```
aws rekognition list-collections
```

Output:

```
{  
  "FaceModelVersions": [  
    "2.0",  
    "3.0",  
    "3.0",  
    "3.0",  
    "4.0",
```

```
        "1.0",
        "3.0",
        "4.0",
        "4.0",
        "4.0"
    ],
    "CollectionIds": [
        "MyCollection1",
        "MyCollection2",
        "MyCollection3",
        "MyCollection4",
        "MyCollection5",
        "MyCollection6",
        "MyCollection7",
        "MyCollection8",
        "MyCollection9",
        "MyCollection10"
    ]
}
```

For more information, see [Listing Collections](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [ListCollections](#) in *AWS CLI Command Reference*.

list-faces

The following code example shows how to use `list-faces`.

For more information, see [Listing faces in a collection](#).

AWS CLI

To list the faces in a collection

The following `list-faces` command lists the faces in the specified collection.

```
aws rekognition list-faces \  
    --collection-id MyCollection
```

Output:

```
{  
    "FaceModelVersion": "3.0",
```

```
"Faces": [  
  {  
    "BoundingBox": {  
      "Width": 0.5216310024261475,  
      "Top": 0.3256250023841858,  
      "Left": 0.13394300639629364,  
      "Height": 0.3918749988079071  
    },  
    "FaceId": "0040279c-0178-436e-b70a-e61b074e96b0",  
    "ExternalImageId": "image1.jpg",  
    "Confidence": 100.0,  
    "ImageId": "f976e487-3719-5e2d-be8b-ea2724c26991"  
  },  
  {  
    "BoundingBox": {  
      "Width": 0.5074880123138428,  
      "Top": 0.3774999976158142,  
      "Left": 0.18302799761295319,  
      "Height": 0.3812499940395355  
    },  
    "FaceId": "086261e8-6deb-4bc0-ac73-ab22323cc38d",  
    "ExternalImageId": "image2.jpg",  
    "Confidence": 99.99930572509766,  
    "ImageId": "ae1593b0-a8f6-5e24-a306-abf529e276fa"  
  },  
  {  
    "BoundingBox": {  
      "Width": 0.5574039816856384,  
      "Top": 0.37187498807907104,  
      "Left": 0.14559100568294525,  
      "Height": 0.4181250035762787  
    },  
    "FaceId": "11c4bd3c-19c5-4eb8-aecc-24feb93a26e1",  
    "ExternalImageId": "image3.jpg",  
    "Confidence": 99.99960327148438,  
    "ImageId": "80739b4d-883f-5b78-97cf-5124038e26b9"  
  },  
  {  
    "BoundingBox": {  
      "Width": 0.18562500178813934,  
      "Top": 0.1618019938468933,  
      "Left": 0.5575000047683716,  
      "Height": 0.24770599603652954  
    },  
  },  
]
```

```
"FaceId": "13692fe4-990a-4679-b14a-5ac23d135eab",
"ExternalImageId": "image4.jpg",
"Confidence": 99.99340057373047,
"ImageId": "8df18239-9ad1-5acd-a46a-6581ff98f51b"
},
{
  "BoundingBox": {
    "Width": 0.5307819843292236,
    "Top": 0.2862499952316284,
    "Left": 0.1564060002565384,
    "Height": 0.3987500071525574
  },
  "FaceId": "2eb5f3fd-e2a9-4b1c-a89f-afa0a518fe06",
  "ExternalImageId": "image5.jpg",
  "Confidence": 99.99970245361328,
  "ImageId": "3c314792-197d-528d-bbb6-798ed012c150"
},
{
  "BoundingBox": {
    "Width": 0.5773710012435913,
    "Top": 0.34437501430511475,
    "Left": 0.12396000325679779,
    "Height": 0.4337500035762787
  },
  "FaceId": "57189455-42b0-4839-a86c-abda48b13174",
  "ExternalImageId": "image6.jpg",
  "Confidence": 100.0,
  "ImageId": "0aff2f37-e7a2-5dbc-a3a3-4ef6ec18eaa0"
},
{
  "BoundingBox": {
    "Width": 0.5349419713020325,
    "Top": 0.29124999046325684,
    "Left": 0.16389399766921997,
    "Height": 0.40187498927116394
  },
  "FaceId": "745f7509-b1fa-44e0-8b95-367b1359638a",
  "ExternalImageId": "image7.jpg",
  "Confidence": 99.99979400634766,
  "ImageId": "67a34327-48d1-5179-b042-01e52ccfeada"
},
{
  "BoundingBox": {
    "Width": 0.41499999165534973,
```

```
        "Top": 0.09187500178813934,  
        "Left": 0.28083300590515137,  
        "Height": 0.3112500011920929  
    },  
    "FaceId": "8d3cfc70-4ba8-4b36-9644-90fba29c2dac",  
    "ExternalImageId": "image8.jpg",  
    "Confidence": 99.99769592285156,  
    "ImageId": "a294da46-2cb1-5cc4-9045-61d7ca567662"  
},  
{  
    "BoundingBox": {  
        "Width": 0.48166701197624207,  
        "Top": 0.20999999344348907,  
        "Left": 0.21250000596046448,  
        "Height": 0.36125001311302185  
    },  
    "FaceId": "bd4ceb4d-9acc-4ab7-8ef8-1c2d2ba0a66a",  
    "ExternalImageId": "image9.jpg",  
    "Confidence": 99.99949645996094,  
    "ImageId": "5e1a7588-e5a0-5ee3-bd00-c642518dfe3a"  
},  
{  
    "BoundingBox": {  
        "Width": 0.18562500178813934,  
        "Top": 0.1618019938468933,  
        "Left": 0.5575000047683716,  
        "Height": 0.24770599603652954  
    },  
    "FaceId": "ce7ed422-2132-4a11-ab14-06c5c410f29f",  
    "ExternalImageId": "image10.jpg",  
    "Confidence": 99.99340057373047,  
    "ImageId": "8d67061e-90d2-598f-9fbd-29c8497039c0"  
    }  
    ]  
}
```

For more information, see [Listing Faces in a Collection](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [ListFaces](#) in *AWS CLI Command Reference*.

list-stream-processors

The following code example shows how to use `list-stream-processors`.

AWS CLI

To list the stream processors in your account

The following `list-stream-processors` command lists the stream processors in your account and the state of each.

```
aws rekognition list-stream-processors
```

Output:

```
{
  "StreamProcessors": [
    {
      "Status": "STOPPED",
      "Name": "my-stream-processor"
    }
  ]
}
```

For more information, see [Working with Streaming Videos](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [ListStreamProcessors](#) in *AWS CLI Command Reference*.

recognize-celebrities

The following code example shows how to use `recognize-celebrities`.

For more information, see [Recognizing celebrities in an image](#).

AWS CLI

To recognize celebrities in an image

The following `recognize-celebrities` command recognizes celebrities in the specified image stored in an Amazon S3 bucket.:

```
aws rekognition recognize-celebrities \  
  --image "S3Object={Bucket=MyImageS3Bucket,Name=moviestars.jpg}"
```

Output:

```
{  
  "UnrecognizedFaces": [  
    {  
      "BoundingBox": {  
        "Width": 0.14416666328907013,  
        "Top": 0.07777778059244156,  
        "Left": 0.625,  
        "Height": 0.2746031880378723  
      },  
      "Confidence": 99.9990234375,  
      "Pose": {  
        "Yaw": 10.80408763885498,  
        "Roll": -12.761146545410156,  
        "Pitch": 10.96889877319336  
      },  
      "Quality": {  
        "Sharpness": 94.1185531616211,  
        "Brightness": 79.18367004394531  
      },  
      "Landmarks": [  
        {  
          "Y": 0.18220913410186768,  
          "X": 0.6702951788902283,  
          "Type": "eyeLeft"  
        },  
        {  
          "Y": 0.16337193548679352,  
          "X": 0.7188183665275574,  
          "Type": "eyeRight"  
        },  
        {  
          "Y": 0.20739148557186127,  
          "X": 0.7055801749229431,  
          "Type": "nose"  
        },  
        {  
          "Y": 0.2889308035373688,  
          "X": 0.687512218952179,  
          "Type": "mouth"  
        }  
      ]  
    }  
  ]  
}
```



```
        "Type": "mouthLeft"
      },
      {
        "Y": 0.2706988751888275,
        "X": 0.7250053286552429,
        "Type": "mouthRight"
      }
    ]
  }
],
"CelebrityFaces": [
  {
    "MatchConfidence": 100.0,
    "Face": {
      "BoundingBox": {
        "Width": 0.14000000059604645,
        "Top": 0.1190476194024086,
        "Left": 0.82833331823349,
        "Height": 0.2666666805744171
      },
      "Confidence": 99.99359130859375,
      "Pose": {
        "Yaw": -10.509642601013184,
        "Roll": -14.51749324798584,
        "Pitch": 13.799399375915527
      },
      "Quality": {
        "Sharpness": 78.74752044677734,
        "Brightness": 42.201324462890625
      },
      "Landmarks": [
        {
          "Y": 0.2290833294391632,
          "X": 0.8709492087364197,
          "Type": "eyeLeft"
        },
        {
          "Y": 0.20639978349208832,
          "X": 0.9153988361358643,
          "Type": "eyeRight"
        },
        {
          "Y": 0.25417643785476685,
          "X": 0.8907724022865295,
```

```
        "Type": "nose"
      },
      {
        "Y": 0.32729196548461914,
        "X": 0.8876466155052185,
        "Type": "mouthLeft"
      },
      {
        "Y": 0.3115464746952057,
        "X": 0.9238573312759399,
        "Type": "mouthRight"
      }
    ]
  },
  "Name": "Celeb A",
  "Urls": [
    "www.imdb.com/name/aaaaaaaaa"
  ],
  "Id": "1111111"
},
{
  "MatchConfidence": 97.0,
  "Face": {
    "BoundingBox": {
      "Width": 0.13333334028720856,
      "Top": 0.24920634925365448,
      "Left": 0.4449999928474426,
      "Height": 0.2539682686328888
    },
    "Confidence": 99.99979400634766,
    "Pose": {
      "Yaw": 6.557040691375732,
      "Roll": -7.316643714904785,
      "Pitch": 9.272967338562012
    },
    "Quality": {
      "Sharpness": 83.23492431640625,
      "Brightness": 78.83267974853516
    },
    "Landmarks": [
      {
        "Y": 0.3625510632991791,
        "X": 0.48898839950561523,
        "Type": "eyeLeft"
      }
    ]
  }
}
```

```
    },
    {
      "Y": 0.35366007685661316,
      "X": 0.5313721299171448,
      "Type": "eyeRight"
    },
    {
      "Y": 0.3894785940647125,
      "X": 0.5173314809799194,
      "Type": "nose"
    },
    {
      "Y": 0.44889405369758606,
      "X": 0.5020005702972412,
      "Type": "mouthLeft"
    },
    {
      "Y": 0.4408611059188843,
      "X": 0.5351271629333496,
      "Type": "mouthRight"
    }
  ]
},
"Name": "Celeb B",
"Urls": [
  "www.imdb.com/name/bbbbbbbbbb"
],
"Id": "2222222"
},
{
  "MatchConfidence": 100.0,
  "Face": {
    "BoundingBox": {
      "Width": 0.12416666746139526,
      "Top": 0.2968254089355469,
      "Left": 0.2150000035762787,
      "Height": 0.23650793731212616
    },
    "Confidence": 99.99958801269531,
    "Pose": {
      "Yaw": 7.801797866821289,
      "Roll": -8.326810836791992,
      "Pitch": 7.844768047332764
    }
  },
}
```

```
    "Quality": {
      "Sharpness": 86.93206024169922,
      "Brightness": 79.81291198730469
    },
    "Landmarks": [
      {
        "Y": 0.4027804136276245,
        "X": 0.2575301229953766,
        "Type": "eyeLeft"
      },
      {
        "Y": 0.3934555947780609,
        "X": 0.2956969439983368,
        "Type": "eyeRight"
      },
      {
        "Y": 0.4309830069541931,
        "X": 0.2837020754814148,
        "Type": "nose"
      },
      {
        "Y": 0.48186683654785156,
        "X": 0.26812544465065,
        "Type": "mouthLeft"
      },
      {
        "Y": 0.47338807582855225,
        "X": 0.29905644059181213,
        "Type": "mouthRight"
      }
    ]
  },
  "Name": "Celeb C",
  "Urls": [
    "www.imdb.com/name/ccccccccc"
  ],
  "Id": "3333333"
},
{
  "MatchConfidence": 97.0,
  "Face": {
    "BoundingBox": {
      "Width": 0.11916666477918625,
      "Top": 0.3698412775993347,
```

```
    "Left": 0.008333333767950535,  
    "Height": 0.22698412835597992  
  },  
  "Confidence": 99.99999237060547,  
  "Pose": {  
    "Yaw": 16.38478660583496,  
    "Roll": -1.0260354280471802,  
    "Pitch": 5.975185394287109  
  },  
  "Quality": {  
    "Sharpness": 83.23492431640625,  
    "Brightness": 61.408443450927734  
  },  
  "Landmarks": [  
    {  
      "Y": 0.4632347822189331,  
      "X": 0.049406956881284714,  
      "Type": "eyeLeft"  
    },  
    {  
      "Y": 0.46388113498687744,  
      "X": 0.08722897619009018,  
      "Type": "eyeRight"  
    },  
    {  
      "Y": 0.5020678639411926,  
      "X": 0.0758260041475296,  
      "Type": "nose"  
    },  
    {  
      "Y": 0.544157862663269,  
      "X": 0.054029736667871475,  
      "Type": "mouthLeft"  
    },  
    {  
      "Y": 0.5463630557060242,  
      "X": 0.08464983850717545,  
      "Type": "mouthRight"  
    }  
  ]  
},  
"Name": "Celeb D",  
"Urls": [  
  "www.imdb.com/name/dddddddd"
```

```
    ],
    "Id": "44444444"
  }
]
}
```

For more information, see [Recognizing Celebrities in an Image](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [RecognizeCelebrities](#) in *AWS CLI Command Reference*.

search-faces-by-image

The following code example shows how to use `search-faces-by-image`.

For more information, see [Searching for a face \(image\)](#).

AWS CLI

To search for faces in a collection that match the largest face in an image.

The following `search-faces-by-image` command searches for faces in a collection that match the largest face in the specified image.:

```
aws rekognition search-faces-by-image \
  --image '{"S3Object":{"Bucket":"MyImageS3Bucket","Name":"ExamplePerson.jpg"}}' \
  --collection-id MyFaceImageCollection

{
  "SearchedFaceBoundingBox": {
    "Width": 0.18562500178813934,
    "Top": 0.1618015021085739,
    "Left": 0.5575000047683716,
    "Height": 0.24770642817020416
  },
  "SearchedFaceConfidence": 99.993408203125,
  "FaceMatches": [
    {
      "Face": {
        "BoundingBox": {
          "Width": 0.18562500178813934,
          "Top": 0.1618019938468933,
          "Left": 0.5575000047683716,
```

```
        "Height": 0.24770599603652954
      },
      "FaceId": "ce7ed422-2132-4a11-ab14-06c5c410f29f",
      "ExternalImageId": "example-image.jpg",
      "Confidence": 99.99340057373047,
      "ImageId": "8d67061e-90d2-598f-9fbd-29c8497039c0"
    },
    "Similarity": 99.97913360595703
  },
  {
    "Face": {
      "BoundingBox": {
        "Width": 0.18562500178813934,
        "Top": 0.1618019938468933,
        "Left": 0.5575000047683716,
        "Height": 0.24770599603652954
      },
      "FaceId": "13692fe4-990a-4679-b14a-5ac23d135eab",
      "ExternalImageId": "image3.jpg",
      "Confidence": 99.99340057373047,
      "ImageId": "8df18239-9ad1-5acd-a46a-6581ff98f51b"
    },
    "Similarity": 99.97913360595703
  },
  {
    "Face": {
      "BoundingBox": {
        "Width": 0.41499999165534973,
        "Top": 0.09187500178813934,
        "Left": 0.28083300590515137,
        "Height": 0.3112500011920929
      },
      "FaceId": "8d3cfc70-4ba8-4b36-9644-90fba29c2dac",
      "ExternalImageId": "image2.jpg",
      "Confidence": 99.99769592285156,
      "ImageId": "a294da46-2cb1-5cc4-9045-61d7ca567662"
    },
    "Similarity": 99.18069458007812
  },
  {
    "Face": {
      "BoundingBox": {
        "Width": 0.48166701197624207,
        "Top": 0.20999999344348907,
```

```
        "Left": 0.21250000596046448,
        "Height": 0.36125001311302185
    },
    "FaceId": "bd4ceb4d-9acc-4ab7-8ef8-1c2d2ba0a66a",
    "ExternalImageId": "image1.jpg",
    "Confidence": 99.99949645996094,
    "ImageId": "5e1a7588-e5a0-5ee3-bd00-c642518dfe3a"
},
"Similarity": 98.66607666015625
},
{
  "Face": {
    "BoundingBox": {
      "Width": 0.5349419713020325,
      "Top": 0.29124999046325684,
      "Left": 0.16389399766921997,
      "Height": 0.40187498927116394
    },
    "FaceId": "745f7509-b1fa-44e0-8b95-367b1359638a",
    "ExternalImageId": "image9.jpg",
    "Confidence": 99.99979400634766,
    "ImageId": "67a34327-48d1-5179-b042-01e52ccfeada"
  },
  "Similarity": 98.24278259277344
},
{
  "Face": {
    "BoundingBox": {
      "Width": 0.5307819843292236,
      "Top": 0.2862499952316284,
      "Left": 0.1564060002565384,
      "Height": 0.3987500071525574
    },
    "FaceId": "2eb5f3fd-e2a9-4b1c-a89f-afa0a518fe06",
    "ExternalImageId": "image10.jpg",
    "Confidence": 99.99970245361328,
    "ImageId": "3c314792-197d-528d-bbb6-798ed012c150"
  },
  "Similarity": 98.10665893554688
},
{
  "Face": {
    "BoundingBox": {
      "Width": 0.5074880123138428,
```



```
        "Top": 0.3774999976158142,
        "Left": 0.18302799761295319,
        "Height": 0.3812499940395355
    },
    "FaceId": "086261e8-6deb-4bc0-ac73-ab22323cc38d",
    "ExternalImageId": "image6.jpg",
    "Confidence": 99.99930572509766,
    "ImageId": "ae1593b0-a8f6-5e24-a306-abf529e276fa"
},
"Similarity": 98.10526275634766
},
{
  "Face": {
    "BoundingBox": {
      "Width": 0.5574039816856384,
      "Top": 0.37187498807907104,
      "Left": 0.14559100568294525,
      "Height": 0.4181250035762787
    },
    "FaceId": "11c4bd3c-19c5-4eb8-aecc-24feb93a26e1",
    "ExternalImageId": "image5.jpg",
    "Confidence": 99.99960327148438,
    "ImageId": "80739b4d-883f-5b78-97cf-5124038e26b9"
  },
  "Similarity": 97.94659423828125
},
{
  "Face": {
    "BoundingBox": {
      "Width": 0.5773710012435913,
      "Top": 0.34437501430511475,
      "Left": 0.12396000325679779,
      "Height": 0.4337500035762787
    },
    "FaceId": "57189455-42b0-4839-a86c-abda48b13174",
    "ExternalImageId": "image8.jpg",
    "Confidence": 100.0,
    "ImageId": "0aff2f37-e7a2-5dbc-a3a3-4ef6ec18eaa0"
  },
  "Similarity": 97.93476867675781
}
],
"FaceModelVersion": "3.0"
```

```
}
```

For more information, see [Searching for a Face Using an Image](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [SearchFacesByImage](#) in *AWS CLI Command Reference*.

search-faces

The following code example shows how to use search-faces.

For more information, see [Searching for a face \(face ID\)](#).

AWS CLI

To search for faces in a collection that match a face ID.

The following search-faces command searches for faces in a collection that match the specified face ID.

```
aws rekognition search-faces \  
  --face-id 8d3cfc70-4ba8-4b36-9644-90fba29c2dac \  
  --collection-id MyCollection
```

Output:

```
{  
  "SearchedFaceId": "8d3cfc70-4ba8-4b36-9644-90fba29c2dac",  
  "FaceModelVersion": "3.0",  
  "FaceMatches": [  
    {  
      "Face": {  
        "BoundingBox": {  
          "Width": 0.48166701197624207,  
          "Top": 0.20999999344348907,  
          "Left": 0.21250000596046448,  
          "Height": 0.36125001311302185  
        },  
        "FaceId": "bd4ceb4d-9acc-4ab7-8ef8-1c2d2ba0a66a",  
        "ExternalImageId": "image1.jpg",  
        "Confidence": 99.99949645996094,  
        "ImageId": "5e1a7588-e5a0-5ee3-bd00-c642518dfe3a"  
      },  
    },  
  ],  
}
```

```
    "Similarity": 99.30997467041016
  },
  {
    "Face": {
      "BoundingBox": {
        "Width": 0.18562500178813934,
        "Top": 0.1618019938468933,
        "Left": 0.5575000047683716,
        "Height": 0.24770599603652954
      },
      "FaceId": "ce7ed422-2132-4a11-ab14-06c5c410f29f",
      "ExternalImageId": "example-image.jpg",
      "Confidence": 99.99340057373047,
      "ImageId": "8d67061e-90d2-598f-9fbd-29c8497039c0"
    },
    "Similarity": 99.24862670898438
  },
  {
    "Face": {
      "BoundingBox": {
        "Width": 0.18562500178813934,
        "Top": 0.1618019938468933,
        "Left": 0.5575000047683716,
        "Height": 0.24770599603652954
      },
      "FaceId": "13692fe4-990a-4679-b14a-5ac23d135eab",
      "ExternalImageId": "image3.jpg",
      "Confidence": 99.99340057373047,
      "ImageId": "8df18239-9ad1-5acd-a46a-6581ff98f51b"
    },
    "Similarity": 99.24862670898438
  },
  {
    "Face": {
      "BoundingBox": {
        "Width": 0.5349419713020325,
        "Top": 0.29124999046325684,
        "Left": 0.16389399766921997,
        "Height": 0.40187498927116394
      },
      "FaceId": "745f7509-b1fa-44e0-8b95-367b1359638a",
      "ExternalImageId": "image9.jpg",
      "Confidence": 99.99979400634766,
      "ImageId": "67a34327-48d1-5179-b042-01e52ccfeada"
```

```
    },
    "Similarity": 96.73158264160156
  },
  {
    "Face": {
      "BoundingBox": {
        "Width": 0.5307819843292236,
        "Top": 0.2862499952316284,
        "Left": 0.1564060002565384,
        "Height": 0.3987500071525574
      },
      "FaceId": "2eb5f3fd-e2a9-4b1c-a89f-afa0a518fe06",
      "ExternalImageId": "image10.jpg",
      "Confidence": 99.99970245361328,
      "ImageId": "3c314792-197d-528d-bbb6-798ed012c150"
    },
    "Similarity": 96.48291015625
  },
  {
    "Face": {
      "BoundingBox": {
        "Width": 0.5074880123138428,
        "Top": 0.3774999976158142,
        "Left": 0.18302799761295319,
        "Height": 0.3812499940395355
      },
      "FaceId": "086261e8-6deb-4bc0-ac73-ab22323cc38d",
      "ExternalImageId": "image6.jpg",
      "Confidence": 99.99930572509766,
      "ImageId": "ae1593b0-a8f6-5e24-a306-abf529e276fa"
    },
    "Similarity": 96.43287658691406
  },
  {
    "Face": {
      "BoundingBox": {
        "Width": 0.5574039816856384,
        "Top": 0.37187498807907104,
        "Left": 0.14559100568294525,
        "Height": 0.4181250035762787
      },
      "FaceId": "11c4bd3c-19c5-4eb8-aecc-24feb93a26e1",
      "ExternalImageId": "image5.jpg",
      "Confidence": 99.99960327148438,
```

```
        "ImageId": "80739b4d-883f-5b78-97cf-5124038e26b9"
      },
      "Similarity": 95.25305938720703
    },
    {
      "Face": {
        "BoundingBox": {
          "Width": 0.5773710012435913,
          "Top": 0.34437501430511475,
          "Left": 0.12396000325679779,
          "Height": 0.4337500035762787
        },
        "FaceId": "57189455-42b0-4839-a86c-abda48b13174",
        "ExternalImageId": "image8.jpg",
        "Confidence": 100.0,
        "ImageId": "0aff2f37-e7a2-5dbc-a3a3-4ef6ec18eaa0"
      },
      "Similarity": 95.22837829589844
    }
  ]
}
```

For more information, see [Searching for a Face Using Its Face ID](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [SearchFaces](#) in *AWS CLI Command Reference*.

start-celebrity-recognition

The following code example shows how to use `start-celebrity-recognition`.

AWS CLI

To start the recognition of celebrities in a stored video

The following `start-celebrity-recognition` command starts a job to look for celebrities in the specified video file stored in an Amazon S3 bucket.

```
aws rekognition start-celebrity-recognition \
  --video "S3Object={Bucket=MyVideoS3Bucket,Name=MyVideoFile.mpg}"
```

Output:

```
{
  "JobId": "1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef"
}
```

For more information, see [Recognizing Celebrities in a Stored Video](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [StartCelebrityRecognition](#) in *AWS CLI Command Reference*.

start-content-moderation

The following code example shows how to use `start-content-moderation`.

AWS CLI

To start the recognition of unsafe content in a stored video

The following `start-content-moderation` command starts a job to detect unsafe content in the specified video file stored in an Amazon S3 bucket.

```
aws rekognition start-content-moderation \
  --video "S3Object={Bucket=MyVideoS3Bucket,Name=MyVideoFile.mpg}"
```

Output:

```
{
  "JobId": "1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef"
}
```

For more information, see [Detecting Unsafe Stored Videos](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [StartContentModeration](#) in *AWS CLI Command Reference*.

start-face-detection

The following code example shows how to use `start-face-detection`.

AWS CLI

To detect faces in a video

The following `start-face-detection` command starts a job to detect faces in the specified video file stored in an Amazon S3 bucket.

```
aws rekognition start-face-detection
  --video "S3Object={Bucket=MyVideoS3Bucket,Name=MyVideoFile.mpg}"
```

Output:

```
{
  "JobId": "1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef"
}
```

For more information, see [Detecting Faces in a Stored Video](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [StartFaceDetection](#) in *AWS CLI Command Reference*.

start-face-search

The following code example shows how to use `start-face-search`.

AWS CLI

To search for faces in a collection that match faces detected in a video

The following `start-face-search` command starts a job to search for faces in a collection that match faces detected in the specified video file in an Amazon S3 bucket.

```
aws rekognition start-face-search \
  --video "S3Object={Bucket=MyVideoS3Bucket,Name=MyVideoFile.mpg}" \
  --collection-id collection
```

Output:

```
{
  "JobId": "1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef"
}
```

For more information, see [Searching Stored Videos for Faces](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [StartFaceSearch](#) in *AWS CLI Command Reference*.

start-label-detection

The following code example shows how to use `start-label-detection`.

AWS CLI

To detect objects and scenes in a video

The following `start-label-detection` command starts a job to detect objects and scenes in the specified video file stored in an Amazon S3 bucket.

```
aws rekognition start-label-detection \  
  --video "S3object={Bucket=MyVideoS3Bucket,Name=MyVideoFile.mpg}"
```

Output:

```
{  
  "JobId": "1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef"  
}
```

For more information, see [Detecting Labels in a Video](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [StartLabelDetection](#) in *AWS CLI Command Reference*.

start-person-tracking

The following code example shows how to use `start-person-tracking`.

AWS CLI

To start the pathing of people in a stored video

The following `start-person-tracking` command starts a job to track the paths that people take in the specified video file stored in an Amazon S3 bucket.:

```
aws rekognition start-person-tracking \  
  --video "S3object={Bucket=MyVideoS3Bucket,Name=MyVideoFile.mpg}"
```


Output:

```
{
  "JobId": "1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef"
}
```

For more information, see [People Pathing](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [StartPersonTracking](#) in *AWS CLI Command Reference*.

start-stream-processor

The following code example shows how to use `start-stream-processor`.

AWS CLI**To start a stream processor**

The following `start-stream-processor` command starts the specified video stream processor.

```
aws rekognition start-stream-processor \
  --name my-stream-processor
```

This command produces no output.

For more information, see [Working with Streaming Videos](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [StartStreamProcessor](#) in *AWS CLI Command Reference*.

stop-stream-processor

The following code example shows how to use `stop-stream-processor`.

AWS CLI**To stop a running stream processor**

The following `stop-stream-processor` command stops the specified running stream processor.

```
aws rekognition stop-stream-processor \  
  --name my-stream-processor
```

This command produces no output.

For more information, see [Working with Streaming Videos](#) in the *Amazon Rekognition Developer Guide*.

- For API details, see [StopStreamProcessor](#) in *AWS CLI Command Reference*.

AWS RAM examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS RAM.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

accept-resource-share-invitation

The following code example shows how to use `accept-resource-share-invitation`.

AWS CLI

To accept a resource share invitation

The following `accept-resource-share-invitation` example accepts the specified resource share invitation. Principals in the invited account can immediately start using the resources in the share.

```
aws ram accept-resource-share-invitation \  
  --resource-share-invitation-arn arn:aws:ram:us-west-2:111111111111:resource-  
share-invitation/1e3477be-4a95-46b4-bbe0-c4001EXAMPLE
```

Output:

```
{  
  "resourceShareInvitation": {  
    "resourceShareInvitationArn": "arn:aws:ram:us-west-2:111111111111:resource-  
share-invitation/1e3477be-4a95-46b4-bbe0-c4001EXAMPLE",  
    "resourceShareName": "MyLicenseShare",  
    "resourceShareArn": "arn:aws:ram:us-west-2:111111111111:resource-  
share/27d09b4b-5e12-41d1-a4f2-19dedEXAMPLE",  
    "senderAccountId": "111111111111",  
    "receiverAccountId": "222222222222",  
    "invitationTimestamp": "2021-09-22T15:07:35.620000-07:00",  
    "status": "ACCEPTED"  
  }  
}
```

- For API details, see [AcceptResourceShareInvitation](#) in *AWS CLI Command Reference*.

associate-resource-share-permission

The following code example shows how to use `associate-resource-share-permission`.

AWS CLI

To associate a RAM managed permission with a resource share

The following `associate-resource-share-permission` example replaces the existing managed permission for the relevant resource type with the specified managed permission. Access to all resources of the relevant resource type is governed by the new permission.

```
aws ram associate-resource-share-permission \  
  --permission-arn arn:aws:ram::aws:permission/  
AWSRAMPermissionGlueDatabaseReadWrite \  
  --replace \  
  --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-  
share/27d09b4b-5e12-41d1-a4f2-19dedEXAMPLE
```

Output:

```
{
  "returnValue": true
}
```

- For API details, see [AssociateResourceSharePermission](#) in *AWS CLI Command Reference*.

associate-resource-share

The following code example shows how to use `associate-resource-share`.

AWS CLI**Example 1: To associate a resource with a resource share**

The following `associate-resource-share` example adds a license configuration to the specified resource share.

```
aws ram associate-resource-share \
  --resource-share arn:aws:ram:us-west-2:123456789012:resource-
share/27d09b4b-5e12-41d1-a4f2-19dedEXAMPLE \
  --resource-arns arn:aws:license-manager:us-west-2:123456789012:license-
configuration:lic-36be0485f5ae379cc74cf8e92EXAMPLE
```

Output:

```
{
  "resourceShareAssociations": [
    {
      "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/27d09b4b-5e12-41d1-a4f2-19dedEXAMPLE",
      "associatedEntity": "arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-36be0485f5ae379cc74cf8e92EXAMPLE",
      "associationType": "RESOURCE",
      "status": "ASSOCIATING",
      "external": false
    }
  ]
}
```

Example 2: To associate a principal with a resource share

The following `associate-resource-share` example grants access for the specified resource share to all accounts in the specified organizational unit.

```
aws ram associate-resource-share \  
  --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-  
share/27d09b4b-5e12-41d1-a4f2-19dedEXAMPLE \  
  --principals arn:aws:organizations::123456789012:ou/o-63bEXAMPLE/ou-46xi-  
rEXAMPLE
```

Output:

```
{  
  "resourceShareAssociations": [  
    {  
      "status": "ASSOCIATING",  
      "associationType": "PRINCIPAL",  
      "associatedEntity": "arn:aws:organizations::123456789012:ou/  
o-63bEXAMPLE/ou-46xi-rEXAMPLE",  
      "external": false,  
      "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-  
share/27d09b4b-5e12-41d1-a4f2-19dedEXAMPLE"  
    }  
  ]  
}
```

- For API details, see [AssociateResourceShare](#) in *AWS CLI Command Reference*.

create-resource-share

The following code example shows how to use `create-resource-share`.

AWS CLI

Example 1: To create a resource share

The following `create-resource-share` example creates an empty resource share with the specified name. You must separately add resources, principals, and permissions to the share.

```
aws ram create-resource-share \  
  --name my-share
```

```
--name MyNewResourceShare
```

Output:

```
{
  "resourceShare": {
    "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/4476c27d-8feb-4b21-afe9-7de23EXAMPLE",
    "name": "MyNewResourceShare",
    "owningAccountId": "123456789012",
    "allowExternalPrincipals": true,
    "status": "ACTIVE",
    "creationTime": 1634586271.302,
    "lastUpdatedTime": 1634586271.302
  }
}
```

Example 2: To create a resource share with AWS accounts as principals

The following `create-resource-share` example creates a resource share and grants access to the specified AWS account (222222222222). If the specified principals are not part of the same AWS Organization, then invitations are sent and must be accepted before access is granted.

```
aws ram create-resource-share \
  --name MyNewResourceShare \
  --principals 222222222222
```

Example 3: To create a resource share restricted to your AWS Organization

The following `create-resource-share` example creates a resource share that is restricted to accounts in the AWS Organization that your account is a member of, and adds the specified OU as a principal. All accounts in that OU can use the resources in the resource share.

```
aws ram create-resource-share \
  --name MyNewResourceShare \
  --no-allow-external-principals \
  --principals arn:aws:organizations::123456789012:ou/o-63bEXAMPLE/ou-46xi-
rEXAMPLE
```

Output:

```
{
  "resourceShare": {
    "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/7be8694e-095c-41ca-9ce8-7be4aEXAMPLE",
    "name": "MyNewResourceShare",
    "owningAccountId": "123456789012",
    "allowExternalPrincipals": false,
    "status": "ACTIVE",
    "creationTime": 1634587042.49,
    "lastUpdatedTime": 1634587042.49
  }
}
```

- For API details, see [CreateResourceShare](#) in *AWS CLI Command Reference*.

delete-resource-share

The following code example shows how to use `delete-resource-share`.

AWS CLI

To delete a resource share

The following `delete-resource-share` example deletes the specified resource share.

```
aws ram delete-resource-share \
  --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-share/7ab63972-
b505-7e2a-420d-6f5d3EXAMPLE
```

The following output indicates success:

```
{
  "returnValue": true
}
```

- For API details, see [DeleteResourceShare](#) in *AWS CLI Command Reference*.

disassociate-resource-share-permission

The following code example shows how to use `disassociate-resource-share-permission`.

AWS CLI

To remove a RAM managed permission for a resource type from a resource share

The following `disassociate-resource-share-permission` example removes the RAM managed permission for Glue databases from the specified resource share.

```
aws ram disassociate-resource-share-permission \  
  --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-  
share/27d09b4b-5e12-41d1-a4f2-19dedEXAMPLE \  
  --permission-arn arn:aws:ram::aws:permission/  
AWSRAMPermissionGlueDatabaseReadWrite
```

Output:

```
{  
  "returnValue": true  
}
```

- For API details, see [DisassociateResourceSharePermission](#) in *AWS CLI Command Reference*.

disassociate-resource-share

The following code example shows how to use `disassociate-resource-share`.

AWS CLI

To remove a resource from a resource share

The following `disassociate-resource-share` example removes the specified resource, in this case a VPC subnet, from the specified resource share. Any principals with access to the resource share can no longer perform operations on that resource.

```
aws ram disassociate-resource-share \  
  --resource-arns arn:aws:ec2:us-west-2:123456789012:subnet/  
subnet-0250c25a1fEXAMPLE \  
  --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-share/7ab63972-  
b505-7e2a-420d-6f5d3EXAMPLE
```

Output:

```
{
```



```
"resourceShareAssociations": [  
  "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-  
share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE",  
  "associatedEntity": "arn:aws:ec2:us-west-2:123456789012:subnet/  
subnet-0250c25a1fEXAMPLE",  
  "associationType": "RESOURCE",  
  "status": "DISASSOCIATING",  
  "external": false  
]  
}
```

- For API details, see [DisassociateResourceShare](#) in *AWS CLI Command Reference*.

enable-sharing-with-aws-organization

The following code example shows how to use `enable-sharing-with-aws-organization`.

AWS CLI

To enable resource sharing across AWS Organizations

The following `enable-sharing-with-aws-organization` example enables resource sharing across your organization and organizational units.

```
aws ram enable-sharing-with-aws-organization
```

The following output indicates success.

```
{  
  "returnValue": true  
}
```

- For API details, see [EnableSharingWithAwsOrganization](#) in *AWS CLI Command Reference*.

get-permission

The following code example shows how to use `get-permission`.

AWS CLI

To retrieve the details for a RAM managed permission

The following `get-permission` example displays the details for the default version of the specified RAM managed permission.

```
aws ram get-permission \
  --permission-arn arn:aws:ram::aws:permission/
  AWSRAMPermissionGlueTableReadWriteForDatabase
```

Output:

```
{
  "permission": {
    "arn": "arn:aws:ram::aws:permission/
  AWSRAMPermissionGlueTableReadWriteForDatabase",
    "version": "2",
    "defaultVersion": true,
    "name": "AWSRAMPermissionGlueTableReadWriteForDatabase",
    "resourceType": "glue:Database",
    "permission": "{\"Effect\":\"Allow\",\"Action\":[\"glue:GetTable
  \", \"glue:UpdateTable\", \"glue>DeleteTable\", \"glue:BatchDeleteTable\",
  \"glue:BatchDeleteTableVersion\", \"glue:GetTableVersion\", \"glue:GetTableVersions
  \", \"glue:GetPartition\", \"glue:GetPartitions\", \"glue:BatchGetPartition\",
  \"glue:BatchCreatePartition\", \"glue>CreatePartition\", \"glue:UpdatePartition
  \", \"glue:BatchDeletePartition\", \"glue>DeletePartition\", \"glue:GetTables\",
  \"glue:SearchTables\"]}",
    "creationTime": 1624912434.431,
    "lastUpdatedTime": 1624912434.431,
    "isResourceTypeDefault": false
  }
}
```

- For API details, see [GetPermission](#) in *AWS CLI Command Reference*.

get-resource-policies

The following code example shows how to use `get-resource-policies`.

AWS CLI

To get the policies for a resource

The following `get-resource-policies` example displays the resource-based permission policies for the specified resource associated with a resource share.

```
aws ram get-resource-policies \  
  --resource-arns arn:aws:ec2:us-west-2:123456789012:subnet/  
subnet-0250c25a1fEXAMPLE
```

Output:

```
{  
  "policies": [  
    "{ \"Version\": \"2008-10-17\", \"Statement\": [{ \"Sid\": \"RamStatement1\",  
\"Effect\": \"Allow\", \"Principal\": { \"AWS\": [] }, \"Action\": [ \"ec2:RunInstances  
\", \"ec2:CreateNetworkInterface\", \"ec2:DescribeSubnets\" ], \"Resource\":  
\"arn:aws:ec2:us-west-2:123456789012:subnet/subnet-0250c25a1fEXAMPLE\" ] } }"  
  ]  
}
```

- For API details, see [GetResourcePolicies](#) in *AWS CLI Command Reference*.

get-resource-share-associations

The following code example shows how to use `get-resource-share-associations`.

AWS CLI

Example 1: To list all resource associations for all resource types

The following `get-resource-share-associations` example lists the resource associations for all resource types across all of your resource shares.

```
aws ram get-resource-share-associations \  
  --association-type RESOURCE
```

Output:

```
{  
  "resourceShareAssociations": [  
    {  
      "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-  
share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE",  
      "associatedEntity": "arn:aws:ec2:us-west-2:123456789012:subnet/  
subnet-0250c25a1fEXAMPLE",  
    }  
  ]  
}
```

```

        "resourceShareName": "MySubnetShare",
        "associationType": "RESOURCE",
        "status": "ASSOCIATED",
        "creationTime": 1565303590.973,
        "lastUpdatedTime": 1565303591.695,
        "external": false
    },
    {
        "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/8167bdfc-4480-4a01-8632-315e0EXAMPLE",
        "associatedEntity": "arn:aws:license-manager:us-
west-2:123456789012:license-configuration:lic-36be0485f5ae379cc74cf8e92EXAMPLE",
        "resourceShareName": "MyLicenseShare",
        "associationType": "RESOURCE",
        "status": "ASSOCIATED",
        "creationTime": 1632342958.457,
        "lastUpdatedTime": 1632342958.907,
        "external": false
    }
]
}

```

Example 2: To list principal associations for a resource share

The following `get-resource-share-associations` example lists only the principal associations for only the specified resource share.

```

aws ram get-resource-share-associations \
  --resource-share-arns arn:aws:ram:us-west-2:123456789012:resource-
share/7be8694e-095c-41ca-9ce8-7be4aEXAMPLE \
  --association-type PRINCIPAL

```

Output:

```

{
  "resourceShareAssociations": [
    {
      "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/7be8694e-095c-41ca-9ce8-7be4aEXAMPLE",
      "resourceShareName": "MyNewResourceShare",
      "associatedEntity": "arn:aws:organizations::123456789012:ou/
o-63bEXAMPLE/ou-46xi-rEXAMPLE",

```

```

        "associationType": "PRINCIPAL",
        "status": "ASSOCIATED",
        "creationTime": 1634587042.49,
        "lastUpdatedTime": 1634587044.291,
        "external": false
    }
]
}

```

- For API details, see [GetResourceShareAssociations](#) in *AWS CLI Command Reference*.

get-resource-share-invitations

The following code example shows how to use `get-resource-share-invitations`.

AWS CLI

To list your resource share invitations

The following `get-resource-share-invitations` example lists your current resource share invitations.

```
aws ram get-resource-share-invitations
```

Output:

```

{
  "resourceShareInvitations": [
    {
      "resourceShareInvitationArn": "arn:aws:ram:us-west-2:1:111111111111:resource-share-invitation/32b639f0-14b8-7e8f-55ea-e6117EXAMPLE",
      "resourceShareName": "project-resource-share",
      "resourceShareArn": "arn:aws:ram:us-west-2:1:111111111111:resource-share/fcb639f0-1449-4744-35bc-a983fEXAMPLE",
      "senderAccountId": "111111111111",
      "receiverAccountId": "222222222222",
      "invitationTimestamp": 1565312166.258,
      "status": "PENDING"
    }
  ]
}

```

```
}
```

- For API details, see [GetResourceShareInvitations](#) in *AWS CLI Command Reference*.

get-resource-shares

The following code example shows how to use `get-resource-shares`.

AWS CLI

Example 1: To list resource shares you own and share with others

The following `get-resource-shares` example lists the resource shares that created and are sharing with others.

```
aws ram get-resource-shares \
  --resource-owner SELF
```

Output:

```
{
  "resourceShares": [
    {
      "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/3ab63985-99d9-1cd2-7d24-75e93EXAMPLE",
      "name": "my-resource-share",
      "owningAccountId": "123456789012",
      "allowExternalPrincipals": false,
      "status": "ACTIVE",
      "tags": [
        {
          "key": "project",
          "value": "lima"
        }
      ]
      "creationTime": 1565295733.282,
      "lastUpdatedTime": 1565295733.282
    },
    {
      "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE",
```

```
        "name": "my-resource-share",
        "owningAccountId": "123456789012",
        "allowExternalPrincipals": true,
        "status": "ACTIVE",
        "creationTime": 1565295733.282,
        "lastUpdatedTime": 1565295733.282
    }
]
```

Example 2: To list resource shares owned by others and shared with you

The following `get-resource-shares` example lists the resource shares that others created and shared with you. In this example, there are none.

```
aws ram get-resource-shares \
  --resource-owner OTHER-ACCOUNTS
```

Output:

```
{
  "resourceShares": []
}
```

- For API details, see [GetResourceShares](#) in *AWS CLI Command Reference*.

list-pending-invitation-resources

The following code example shows how to use `list-pending-invitation-resources`.

AWS CLI

To list the resources that are available in a pending resource share

The following `list-pending-invitation-resources` example lists all of the resources that are in the resource share associated with the specified invitation.

```
aws ram list-pending-invitation-resources \
  --resource-share-invitation-arn arn:aws:ram:us-west-2:123456789012:resource-
share-invitation/1e3477be-4a95-46b4-bbe0-c4001EXAMPLE
```

Output:

```
{
  "resources": [
    {
      "arn": "arn:aws:ec2:us-west-2:123456789012:subnet/
subnet-04a555b0e6EXAMPLE",
      "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/7be8694e-095c-41ca-9ce8-7be4aEXAMPLE",
      "creationTime": 1634676051.269,
      "lastUpdatedTime": 1634676052.07,
      "status": "AVAILABLE",
      "type": "ec2:Subnet"
    },
    {
      "arn": "arn:aws:license-manager:us-west-2:123456789012:license-
configuration:lic-36be0485f5ae379cc74cf8e92EXAMPLE",
      "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE",
      "creationTime": 1624912434.431,
      "lastUpdatedTime": 1624912434.431,
      "status": "AVAILABLE",
      "type": "license-manager:LicenseConfiguration"
    }
  ]
}
```

- For API details, see [ListPendingInvitationResources](#) in *AWS CLI Command Reference*.

list-permissions

The following code example shows how to use `list-permissions`.

AWS CLI**To list the available RAM managed permissions**

The following `list-permissions` example lists all of the RAM managed permissions available for only the AWS Glue database resource type.

```
aws ram list-permissions \
  --resource-type glue:Database
```


Output:

```
{
  "permissions": [
    {
      "arn": "arn:aws:ram::aws:permission/
AWSRAMDefaultPermissionGlueDatabase",
      "version": "1",
      "defaultVersion": true,
      "name": "AWSRAMDefaultPermissionGlueDatabase",
      "resourceType": "glue:Database",
      "creationTime": 1592007820.935,
      "lastUpdatedTime": 1592007820.935,
      "isResourceTypeDefault": true
    },
    {
      "arn": "arn:aws:ram::aws:permission/
AWSRAMPermissionGlueAllTablesReadWriteForDatabase",
      "version": "2",
      "defaultVersion": true,
      "name": "AWSRAMPermissionGlueAllTablesReadWriteForDatabase",
      "resourceType": "glue:Database",
      "creationTime": 1624912413.323,
      "lastUpdatedTime": 1624912413.323,
      "isResourceTypeDefault": false
    },
    {
      "arn": "arn:aws:ram::aws:permission/
AWSRAMPermissionGlueDatabaseReadWrite",
      "version": "2",
      "defaultVersion": true,
      "name": "AWSRAMPermissionGlueDatabaseReadWrite",
      "resourceType": "glue:Database",
      "creationTime": 1624912417.4,
      "lastUpdatedTime": 1624912417.4,
      "isResourceTypeDefault": false
    },
    {
      "arn": "arn:aws:ram::aws:permission/
AWSRAMPermissionGlueTableReadWriteForDatabase",
      "version": "2",
      "defaultVersion": true,
      "name": "AWSRAMPermissionGlueTableReadWriteForDatabase",
      "resourceType": "glue:Database",
```

```

        "creationTime": 1624912434.431,
        "lastUpdatedTime": 1624912434.431,
        "isResourceTypeDefault": false
    }
]
}

```

The following `list-permissions` example displays the available RAM managed permissions for all resource types.

```
aws ram list-permissions
```

Output:

```

{
  "permissions": [
    {
      "arn": "arn:aws:ram::aws:permission/
AWSRAMBlankEndEntityCertificateAPICSRPassthroughIssuanceCertificateAuthority",
      "version": "1",
      "defaultVersion": true,
      "name":
"AWSRAMBlankEndEntityCertificateAPICSRPassthroughIssuanceCertificateAuthority",
      "resourceType": "acm-pca:CertificateAuthority",
      "creationTime": 1623264861.085,
      "lastUpdatedTime": 1623264861.085,
      "isResourceTypeDefault": false
    },
    {
      "arn": "arn:aws:ram::aws:permission/AWSRAMDefaultPermissionAppMesh",
      "version": "1",
      "defaultVersion": true,
      "name": "AWSRAMDefaultPermissionAppMesh",
      "resourceType": "appmesh:Mesh",
      "creationTime": 1589307188.584,
      "lastUpdatedTime": 1589307188.584,
      "isResourceTypeDefault": true
    },
    ...TRUNCATED FOR BREVITY...
    {
      "arn": "arn:aws:ram::aws:permission/
AWSRAMSubordinateCACertificatePathLen0IssuanceCertificateAuthority",
      "version": "1",

```

```

        "defaultVersion": true,
        "name":
"AWSRAMSubordinateCACertificatePathLen0IssuanceCertificateAuthority",
        "resourceType": "acm-pca:CertificateAuthority",
        "creationTime": 1623264876.75,
        "lastUpdatedTime": 1623264876.75,
        "isResourceTypeDefault": false
    }
]
}

```

- For API details, see [ListPermissions](#) in *AWS CLI Command Reference*.

list-principals

The following code example shows how to use `list-principals`.

AWS CLI

To list principals with access to a resource

The following `list-principals` example displays a list of the principals that can access resources of the specified type through any resource shares.

```

aws ram list-principals \
  --resource-type ec2:Subnet

```

Output:

```

{
  "principals": [
    {
      "id": "arn:aws:organizations::123456789012:ou/o-gx7EXAMPLE/ou-29c5-
zEXAMPLE",
      "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE",
      "creationTime": 1565298209.737,
      "lastUpdatedTime": 1565298211.019,
      "external": false
    }
  ]
}

```

- For API details, see [ListPrincipals](#) in *AWS CLI Command Reference*.

list-resource-share-permissions

The following code example shows how to use `list-resource-share-permissions`.

AWS CLI

To list all of the RAM managed permissions currently attached to a resource share

The following `list-resource-share-permissions` example lists all of the RAM managed permissions that are attached to the specified resource share.

```
aws ram list-resource-share-permissions \
  --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-
share/27d09b4b-5e12-41d1-a4f2-19dedEXAMPLE
```

Output:

```
{
  "permissions": [
    {
      "arn": "arn:aws:ram::aws:permission/
AWSRAMDefaultPermissionLicenseConfiguration",
      "version": "1",
      "resourceType": "license-manager:LicenseConfiguration",
      "status": "ASSOCIATED",
      "lastUpdatedTime": 1632342984.234
    },
    {
      "arn": "arn:aws:ram::aws:permission/
AWSRAMPermissionGlueDatabaseReadWrite",
      "version": "2",
      "resourceType": "glue:Database",
      "status": "ASSOCIATED",
      "lastUpdatedTime": 1632512462.297
    }
  ]
}
```

- For API details, see [ListResourceSharePermissions](#) in *AWS CLI Command Reference*.

list-resource-types

The following code example shows how to use `list-resource-types`.

AWS CLI

To list the resource types that are supported by AWS RAM

The following `list-resource-types` example lists all of the resource types currently supported by AWS RAM.

```
aws ram list-resource-types
```

Output:

```
{
  "resourceTypes": [
    {
      "resourceType": "route53resolver:FirewallRuleGroup",
      "serviceName": "route53resolver"
    },
    {
      "resourceType": "ec2:LocalGatewayRouteTable",
      "serviceName": "ec2"
    },
    ...OUTPUT TRUNCATED FOR BREVITY...
    {
      "resourceType": "ec2:Subnet",
      "serviceName": "ec2"
    },
    {
      "resourceType": "ec2:TransitGatewayMulticastDomain",
      "serviceName": "ec2"
    }
  ]
}
```

- For API details, see [ListResourceTypes](#) in *AWS CLI Command Reference*.

list-resources

The following code example shows how to use `list-resources`.

AWS CLI

To list the resources associated with a resource share

The following `list-resources` example lists all resources in the specified resource share that are of the specified resource type.

```
aws ram list-resources \
  --resource-type ec2:Subnet \
  --resource-owner SELF \
  --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-share/7ab63972-
b505-7e2a-420d-6f5d3EXAMPLE
```

Output:

```
{
  "resources": [
    {
      "arn": "aarn:aws:ec2:us-west-2:123456789012:subnet/
subnet-0250c25a1f4e15235",
      "type": "ec2:Subnet",
      "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-
share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE",
      "creationTime": 1565301545.023,
      "lastUpdatedTime": 1565301545.947
    }
  ]
}
```

- For API details, see [ListResources](#) in *AWS CLI Command Reference*.

promote-resource-share-created-from-policy

The following code example shows how to use `promote-resource-share-created-from-policy`.

AWS CLI

To promote a resource-policy based resource share to full functionality in AWS RAM

The following `promote-resource-share-created-from-policy` example takes a resource share that you created implicitly by attaching a resource-based policy, and converts it to be fully functional with the AWS RAM console and its CLI and API operations.

```
aws ram promote-resource-share-created-from-policy \  
  --resource-share-arn arn:aws:ram:us-east-1:123456789012:resource-  
share/91fa8429-2d06-4032-909a-90909EXAMPLE
```

Output:

```
{  
  "returnValue": true  
}
```

- For API details, see [PromoteResourceShareCreatedFromPolicy](#) in *AWS CLI Command Reference*.

reject-resource-share-invitation

The following code example shows how to use `reject-resource-share-invitation`.

AWS CLI

To reject a resource share invitation

The following `reject-resource-share-invitation` example rejects the specified resource share invitation.

```
aws ram reject-resource-share-invitation \  
  --resource-share-invitation-arn arn:aws:ram:us-west-2:111111111111:resource-  
share-invitation/32b639f0-14b8-7e8f-55ea-e6117EXAMPLE
```

Output:

```
"resourceShareInvitations": [  
  {  
    "resourceShareInvitationArn": "arn:aws:ram:us-west2-1:111111111111:resource-  
share-invitation/32b639f0-14b8-7e8f-55ea-e6117EXAMPLE",  
    "resourceShareName": "project-resource-share",
```

```
    "resourceShareArn": "arn:aws:ram:us-west-2:111111111111:resource-share/
    fcb639f0-1449-4744-35bc-a983fEXAMPLE",
    "senderAccountId": "111111111111",
    "receiverAccountId": "222222222222",
    "invitationTimestamp": 1565319592.463,
    "status": "REJECTED"
  }
]
```

- For API details, see [RejectResourceShareInvitation](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add tags to a resource share

The following `tag-resource` example adds a tag key `project` and associated value `lima` to the specified resource share.

```
aws ram tag-resource \
  --tags key=project,value=lima \
  --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-share/7ab63972-
  b505-7e2a-420d-6f5d3EXAMPLE
```

This command produces no output.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from a resource share

The following `untag-resource` example removes the `project` tag key and associated value from the specified resource share.


```
aws ram untag-resource \  
  --tag-keys project \  
  --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-share/7ab63972-  
b505-7e2a-420d-6f5d3EXAMPLE
```

This command produces no output.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-resource-share

The following code example shows how to use `update-resource-share`.

AWS CLI

To update a resource share

The following `update-resource-share` example changes the specified resource share to allow external principals that are not in an AWS Organization.

```
aws ram update-resource-share \  
  --allow-external-principals \  
  --resource-share-arn arn:aws:ram:us-west-2:123456789012:resource-share/7ab63972-  
b505-7e2a-420d-6f5d3EXAMPLE
```

Output:

```
{  
  "resourceShare": {  
    "resourceShareArn": "arn:aws:ram:us-west-2:123456789012:resource-  
share/7ab63972-b505-7e2a-420d-6f5d3EXAMPLE",  
    "name": "my-resource-share",  
    "owningAccountId": "123456789012",  
    "allowExternalPrincipals": true,  
    "status": "ACTIVE",  
    "creationTime": 1565295733.282,  
    "lastUpdatedTime": 1565303080.023  
  }  
}
```

- For API details, see [UpdateResourceShare](#) in *AWS CLI Command Reference*.

Resource Explorer examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Resource Explorer.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-default-view

The following code example shows how to use `associate-default-view`.

AWS CLI

To set a Resource Explorer view as the default for its AWS Region

The following `associate-default-view` example sets a view, as specified by its ARN, to be the default view for the AWS Region in which you call the operation.

```
aws resource-explorer-2 associate-default-view \  
  --view-arn arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-Main-View/  
EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111
```

Output:

```
{  
  "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-Main-  
View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111"  
}
```

For more information, see [Setting a default view in an AWS Region](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [AssociateDefaultView](#) in *AWS CLI Command Reference*.

batch-get-view

The following code example shows how to use `batch-get-view`.

AWS CLI

To retrieve details about multiple Resource Explorer views

The following `batch-get-view` example displays the details about two views specified by their ARNs. Use spaces to separate the multiple ARNs in the `--view-arn` parameter.

```
aws resource-explorer-2 batch-get-view \
  --view-arns arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-EC2-Only-
View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222, \
              arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-Main-
View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111
```

Output:

```
{
  "Views": [
    {
      "Filters": {
        "FilterString": "service:ec2"
      },
      "IncludedProperties": [
        {
          "Name": "tags"
        }
      ],
      "LastUpdatedAt": "2022-07-13T21:33:45.249000+00:00",
      "Owner": "123456789012",
      "Scope": "arn:aws:iam::123456789012:root",
      "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-
EC2-Only-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222"
    },
    {
```

```
    "Filters": {
      "FilterString": ""
    },
    "IncludedProperties": [
      {
        "Name": "tags"
      }
    ],
    "LastUpdatedAt": "2022-07-13T20:34:11.314000+00:00",
    "Owner": "123456789012",
    "Scope": "arn:aws:iam::123456789012:root",
    "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-
Main-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111"
  }
]
"Errors": []
}
```

For more information about views, see [About Resource Explorer views](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [BatchGetView](#) in *AWS CLI Command Reference*.

create-index

The following code example shows how to use `create-index`.

AWS CLI

To turn on Resource Explorer in an AWS Region by creating an index

The following `create-index` example creates a local index in the AWS Region in which the operation is called. The AWS CLI automatically generates a random `client-token` parameter value and includes it in the call to AWS if you don't specify a value.

```
aws resource-explorer-2 create-index \
  --region us-east-1
```

Output:

```
{
```

```
"Arn": "arn:aws:resource-explorer-2:us-east-1:123456789012:index/EXAMPLE8-90ab-
cdef-fedc-EXAMPLE22222c",
"CreatedAt": "2022-11-01T20:00:59.149Z",
"State": "CREATING"
}
```

After you create a local index, you can convert it into the aggregator index for the account by running the [update-index-type](#) command.

For more information, see [Turning on Resource Explorer in an AWS Region to index your resources](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [CreateIndex](#) in *AWS CLI Command Reference*.

create-view

The following code example shows how to use `create-view`.

AWS CLI

Example 1: To create an unfiltered view for the index in an AWS Region

The following `create-view` example creates a view in the specified AWS Region that returns all results in the Region without any filtering. The view includes the optional `Tags` field on returned results. Because this view is created in the Region that contains the aggregator index, it can include results from all Regions in the account that contain a Resource Explorer index.

```
aws resource-explorer-2 create-view \
  --view-name My-Main-View \
  --included-properties Name=tags \
  --region us-east-1
```

Output:

```
{
  "View": {
    "Filters": {
      "FilterString": ""
    },
    "IncludedProperties": [
      {
        "Name": "tags"
      }
    ]
  }
}
```

```

    }
  ],
  "LastUpdatedAt": "2022-07-13T20:34:11.314000+00:00",
  "Owner": "123456789012",
  "Scope": "arn:aws:iam::123456789012:root",
  "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-Main-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111"
}
}

```

Example 2: To create a view that returns only resources associated with Amazon EC2

The following `create-view` creates a view in AWS Region `us-east-1` that returns only those resources in the Region that are associated with the Amazon EC2 service. The view includes the optional `Tags` field on returned results. Because this view is created in the Region that contains the aggregator index, it can include results from all Regions in the account that contain a Resource Explorer index.

```

aws resource-explorer-2 create-view \
  --view-name My-EC2-Only-View \
  --included-properties Name=tags \
  --filters FilterString="service:ec2" \
  --region us-east-1

```

Output:

```

{
  "View": {
    "Filters": {
      "FilterString": "service:ec2"
    },
    "IncludedProperties": [
      {
        "Name": "tags"
      }
    ],
    "LastUpdatedAt": "2022-07-13T21:35:09.059Z",
    "Owner": "123456789012",
    "Scope": "arn:aws:iam::123456789012:root",
    "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-EC2-Only-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222"
  }
}

```

```
}
```

For more information, see [Creating views for search](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [CreateView](#) in *AWS CLI Command Reference*.

delete-index

The following code example shows how to use `delete-index`.

AWS CLI

To turn off Resource Explorer in an AWS Region by deleting its index

The following `delete-index` example deletes the specified Resource Explorer index in the AWS Region in which you make the request.

```
aws resource-explorer-2 delete-index \  
  --arn arn:aws:resource-explorer-2:us-west-2:123456789012:index/EXAMPLE8-90ab-  
cdef-fedc-EXAMPLE22222 \  
  --region us-west-2
```

Output:

```
{  
  "Arn": "arn:aws:resource-explorer-2:us-west-2:123456789012:index/EXAMPLE8-90ab-  
cdef-fedc-EXAMPLE22222",  
  "State": "DELETING"  
}
```

For more information about deleting an index, see [Turning off AWS Resource Explorer in an AWS Region](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [DeleteIndex](#) in *AWS CLI Command Reference*.

delete-view

The following code example shows how to use `delete-view`.

AWS CLI

To delete a Resource Explorer view

The following `delete-view` example deletes a view specified by its ARN.

```
aws resource-explorer-2 delete-view \  
  --view-arn arn:aws:resource-explorer-2:us-east-1:123456789012:view/EC2-Only-  
View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111
```

Output:

```
{  
  "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/EC2-Only-  
View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111"  
}
```

For more information, see [Deleting views](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [DeleteView](#) in *AWS CLI Command Reference*.

disassociate-default-view

The following code example shows how to use `disassociate-default-view`.

AWS CLI

To remove the default Resource Explorer view for an AWS Region

The following `disassociate-default-view` removes the default Resource Explorer view for the AWS Region in which you call the operation. After performing this operation, all search operations in the Region must explicitly specify a view or the operation fails.

```
aws resource-explorer-2 disassociate-default-view
```

This command produces no output.

For more information, see [Setting a default view in an AWS Region](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [DisassociateDefaultView](#) in *AWS CLI Command Reference*.

get-default-view

The following code example shows how to use `get-default-view`.

AWS CLI

To retrieve the Resource Explorer view that is the default view for its AWS Region

The following `get-default-view` example retrieves the ARN of the view that is the default for the AWS Region in which you call the operation.

```
aws resource-explorer-2 get-default-view
```

Output:

```
{
  "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/default-
view/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111"
}
```

For more information, see [Setting a default view in an AWS Region](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [GetDefaultView](#) in *AWS CLI Command Reference*.

get-index

The following code example shows how to use `get-index`.

AWS CLI

Example 1: To retrieve the details for a Resource Explorer aggregator index

The following `get-index` example displays the details for the Resource Explorer index in the specified AWS Region. Because the specified Region contains the aggregator index for the account, the output lists the Regions that replicate data into this Region's index.

```
aws resource-explorer-2 get-index \
  --region us-east-1
```

Output:

```
{
  "Arn": "arn:aws:resource-explorer-2:us-east-1:123456789012:index/EXAMPLE8-90ab-
cdef-fedc-EXAMPLE11111",
  "CreatedAt": "2022-07-12T18:59:10.503000+00:00",
```

```

    "LastUpdatedAt": "2022-07-13T18:41:58.799000+00:00",
    "ReplicatingFrom": [
      "ap-south-1",
      "us-west-2"
    ],
    "State": "ACTIVE",
    "Tags": {},
    "Type": "AGGREGATOR"
  }

```

Example 2: To retrieve the details for a Resource Explorer local index

The following `get-index` example displays the details for the Resource Explorer index in the specified AWS Region. Because the specified Region contains a local index, the output lists the Region to which it replicates data from this Region's index.

```

aws resource-explorer-2 get-index \
  --region us-west-2

```

Output:

```

{
  "Arn": "arn:aws:resource-explorer-2:us-west-2:123456789012:index/EXAMPLE8-90ab-
  cdef-fedc-EXAMPLE22222",
  "CreatedAt": "2022-07-12T18:59:10.503000+00:00",
  "LastUpdatedAt": "2022-07-13T18:41:58.799000+00:00",
  "ReplicatingTo": [
    "us-west-2"
  ],
  "State": "ACTIVE",
  "Tags": {},
  "Type": "LOCAL"
}

```

For more information about indexes, see [Checking which AWS Regions have Resource Explorer turned on](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [GetIndex](#) in *AWS CLI Command Reference*.

get-view

The following code example shows how to use `get-view`.

AWS CLI

To retrieve details about a Resource Explorer view

The following `get-view` example displays the details about a view specified by its ARN.

```
aws resource-explorer-2 get-view \  
  --view-arn arn:aws:resource-explorer-2:us-east-1:123456789012:view/EC2-Only-  
View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111
```

Output:

```
{  
  "Tags" : {},  
  "View" : {  
    "Filters" : {  
      "FilterString" : "service:ec2"  
    },  
    "IncludedProperties" : [  
      {  
        "Name" : "tags"  
      }  
    ],  
    "LastUpdatedAt" : "2022-07-13T21:33:45.249Z",  
    "Owner" : "123456789012",  
    "Scope" : "arn:aws:iam::123456789012:root",  
    "ViewArn" : "arn:aws:resource-explorer-2:us-east-1:123456789012:view/EC2-  
Only-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111"  
  }  
}
```

For more information about views, see [About Resource Explorer views](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [GetView](#) in *AWS CLI Command Reference*.

list-indexes

The following code example shows how to use `list-indexes`.

AWS CLI

To list the AWS Regions where Resource Explorer has indexes

The following `list-indexes` example lists the indexes for all Regions where Resource Explorer has an index. The response specifies the type of each index, its AWS Region, and its ARN.

```
aws resource-explorer-2 list-indexes
```

Output:

```
{
  "Indexes": [
    {
      "Arn": "arn:aws:resource-explorer-2:us-west-2:123456789012:index/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111",
      "Region": "us-west-2",
      "Type": "AGGREGATOR"
    },
    {
      "Arn": "arn:aws:resource-explorer-2:us-east-1:123456789012:index/EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222",
      "Region": "us-east-1",
      "Type": "LOCAL"
    },
    {
      "Arn": "arn:aws:resource-explorer-2:us-east-2:123456789012:index/EXAMPLE8-90ab-cdef-fedc-EXAMPLE33333",
      "Region": "us-east-2",
      "Type": "LOCAL"
    },
    {
      "Arn": "arn:aws:resource-explorer-2:us-west-1:123456789012:index/EXAMPLE8-90ab-cdef-fedc-EXAMPLE44444",
      "Region": "us-west-1",
      "Type": "LOCAL"
    }
  ]
}
```

For more information about indexes, see [Checking which AWS Regions have Resource Explorer turned on](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [ListIndexes](#) in *AWS CLI Command Reference*.

list-supported-resource-types

The following code example shows how to use `list-supported-resource-types`.

AWS CLI

To list the AWS Regions where Resource Explorer has indexes

The following `list-supported-resource-types` example lists all of the resource types currently supported by &AREXlong;. The example response includes a `NextToken` value, which indicates that there is more output available to retrieve with additional calls.

```
aws resource-explorer-2 list-supported-resource-types \  
  --max-items 10
```

Output:

```
{  
  "ResourceTypes": [  
    {  
      "ResourceType": "cloudfront:cache-policy",  
      "Service": "cloudfront"  
    },  
    {  
      "ResourceType": "cloudfront:distribution",  
      "Service": "cloudfront"  
    },  
    {  
      "ResourceType": "cloudfront:function",  
      "Service": "cloudfront"  
    },  
    {  
      "ResourceType": "cloudfront:origin-access-identity",  
      "Service": "cloudfront"  
    },  
    {  
      "ResourceType": "cloudfront:origin-request-policy",  
      "Service": "cloudfront"  
    },  
    {  
      "ResourceType": "cloudfront:realtime-log-config",  
      "Service": "cloudfront"  
    },  
  ],  
}
```

```

    {
      "ResourceType": "cloudfront:response-headers-policy",
      "Service": "cloudfront"
    },
    {
      "ResourceType": "cloudwatch:alarm",
      "Service": "cloudwatch"
    },
    {
      "ResourceType": "cloudwatch:dashboard",
      "Service": "cloudwatch"
    },
    {
      "ResourceType": "cloudwatch:insight-rule",
      "Service": "cloudwatch"
    }
  ],
  "NextToken": "eyJ0ZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAxMH0="
}

```

To get the next part of the output, call the operation again, and pass the previous call's `NextToken` response value as the value for `--starting-token`. Repeat until `NextToken` is absent from the response.

```

aws resource-explorer-2 list-supported-resource-types \
  --max-items 10 \
  --starting-token
eyJ0ZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAxMH0=

```

Output:

```

{
  "ResourceTypes": [
    {
      "ResourceType": "cloudwatch:metric-stream",
      "Service": "cloudwatch"
    },
    {
      "ResourceType": "dynamodb:table",
      "Service": "dynamodb"
    },
    {

```

```

    "ResourceType": "ec2:capacity-reservation",
    "Service": "ec2"
  },
  {
    "ResourceType": "ec2:capacity-reservation-fleet",
    "Service": "ec2"
  },
  {
    "ResourceType": "ec2:client-vpn-endpoint",
    "Service": "ec2"
  },
  {
    "ResourceType": "ec2:customer-gateway",
    "Service": "ec2"
  },
  {
    "ResourceType": "ec2:dedicated-host",
    "Service": "ec2"
  },
  {
    "ResourceType": "ec2:dhcp-options",
    "Service": "ec2"
  },
  {
    "ResourceType": "ec2:egress-only-internet-gateway",
    "Service": "ec2"
  },
  {
    "ResourceType": "ec2:elastic-gpu",
    "Service": "ec2"
  }
],
"NextToken": "eyJ0ZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAyMH0="
}

```

For more information about indexes, see [Checking which AWS Regions have Resource Explorer turned on](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [ListSupportedResourceTypes](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags attached to a Resource Explorer view or index

The following `list-tags-for-resource` example lists the tag key and value pairs attached to view with the specified ARN. You must call the operation from the AWS Region that contains the resource.

```
aws resource-explorer-2 list-tags-for-resource \
  --resource-arn arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-View/
EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111
```

Output:

```
{
  "Tags": {
    "application": "MainCorpApp",
    "department": "1234"
  }
}
```

For more information about tagging views, see [Tagging views for access control](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

list-views

The following code example shows how to use `list-views`.

AWS CLI

To list the Resource Explorer views available in an AWS Region

The following `list-views` example lists all of the views available in the Region in which you invoke the operation.

```
aws resource-explorer-2 list-views
```

Output:


```
{
  "Views": [
    "arn:aws:resource-explorer-2:us-east-1:123456789012:view/EC2-Only-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111",
    "arn:aws:resource-explorer-2:us-east-1:123456789012:view/Default-All-Resources-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222",
    "arn:aws:resource-explorer-2:us-east-1:123456789012:view/Production-Only-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE33333"
  ]
}
```

For more information about views, see [About Resource Explorer views](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [ListViews](#) in *AWS CLI Command Reference*.

search

The following code example shows how to use search.

AWS CLI

Example 1: To search using the default view

The following search example displays all resources in the specified that are associated with the service. The search uses the default view for the Region. The example response includes a `NextToken` value, which indicates that there is more output available to retrieve with additional calls.

```
aws resource-explorer-2 search \
  --query-string "service:iam"
```

Output:

```
{
  "Count": {
    "Complete": true,
    "TotalResources": 55
  },
  "NextToken":
  "AG9V0EF1KLEXAMPLE0hJHVwo5chEXAMPLER5XiEpNrgsEXAMPLE...b0Cm0F0ryHEXAMPLE",
  "Resources": [{
```

```

    "Arn": "arn:aws:iam::123456789012:policy/service-role/Some-Policy-For-A-
Service-Role",
    "LastReportedAt": "2022-07-21T12:34:42Z",
    "OwningAccountId": "123456789012",
    "Properties": [],
    "Region": "global",
    "ResourceType": "iam:policy",
    "Service": "iam"
  }, {
    "Arn": "arn:aws:iam::123456789012:policy/service-role/Another-Policy-For-A-
Service-Role",
    "LastReportedAt": "2022-07-21T12:34:42Z",
    "OwningAccountId": "123456789012",
    "Properties": [],
    "Region": "global",
    "ResourceType": "iam:policy",
    "Service": "iam"
  }, {
    ... TRUNCATED FOR BREVITY ...
  }],
  "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/my-default-
view/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111"
}

```

Example 2: To search using a specified view

The following search example search displays all resources ("*") in the specified AWS Region that are visible through the specified view. The results include only resources associated with Amazon EC2 because of the filters attached to the view.

```

aws resource-explorer-2 search \
  -- query-string "*" \
  -- view-arn arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-EC2-view/
EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222

```

Output:

```

HTTP/1.1 200 OK
Date: Tue, 01 Nov 2022 20:00:59 GMT
Content-Type: application/json
Content-Length: <PayloadSizeBytes>

{

```

```
"Count": {
  "Complete": true,
  "TotalResources": 67
},
"Resources": [{
  "Arn": "arn:aws:ec2:us-east-1:123456789012:network-acl/acl-1a2b3c4d",
  "LastReportedAt": "2022-07-21T18:52:02Z",
  "OwningAccountId": "123456789012",
  "Properties": [{
    "Data": [{
      "Key": "Department",
      "Value": "AppDevelopment"
    }, {
      "Key": "Environment",
      "Value": "Production"
    }
  ]],
  "LastReportedAt": "2021-11-15T14:48:29Z",
  "Name": "tags"
}],
"Region": "us-east-1",
"ResourceType": "ec2:network-acl",
"Service": "ec2"
}, {
  "Arn": "arn:aws:ec2:us-east-1:123456789012:subnet/subnet-1a2b3c4d",
  "LastReportedAt": "2022-07-21T21:22:23Z",
  "OwningAccountId": "123456789012",
  "Properties": [{
    "Data": [{
      "Key": "Department",
      "Value": "AppDevelopment"
    }, {
      "Key": "Environment",
      "Value": "Production"
    }
  ]],
  "LastReportedAt": "2021-07-29T19:02:39Z",
  "Name": "tags"
}],
"Region": "us-east-1",
"ResourceType": "ec2:subnet",
"Service": "ec2"
}, {
  "Arn": "arn:aws:ec2:us-east-1:123456789012:dhcp-options/dopt-1a2b3c4d",
  "LastReportedAt": "2022-07-21T06:08:53Z",
  "OwningAccountId": "123456789012",
```

```

    "Properties": [{
      "Data": [{
        "Key": "Department",
        "Value": "AppDevelopment"
      }, {
        "Key": "Environment",
        "Value": "Production"
      }],
      "LastReportedAt": "2021-11-15T15:11:05Z",
      "Name": "tags"
    }],
    "Region": "us-east-1",
    "ResourceType": "ec2:dhcptions",
    "Service": "ec2"
  }, {
    ... TRUNCATED FOR BREVITY ...
  }],
  "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-EC2-
view/EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222"
}

```

For more information, see [Using AWS Resource Explorer to search for resources](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [Search](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

To tag a Resource Explorer view

The following tag-resource example adds the tag key "environment" with the value "production" to the view with the specified ARN.

```

aws resource-explorer-2 tag-resource \
  --resource-arn arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-View//
EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111 \
  --tags environment=production

```

This command produces no output.

For more information, see [Tagging views for access control](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag from a Resource Explorer view

The following `untag-resource` example removes any tag with the key name "environment" from the view with the specified ARN.

```
aws resource-explorer-2 untag-resource \  
  --resource-arn arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-View//  
EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111 \  
  --tag-keys environment
```

This command produces no output.

For more information, see [Tagging views for access control](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-index-type

The following code example shows how to use `update-index-type`.

AWS CLI

To change the type of a Resource Explorer index

The following `update-index-type` example converts the specified index from type `local` to type `aggregator` to turn on the ability to search for resources across all AWS Regions in the account. You must send the request to the AWS Region that contains the index you want to update.

```
aws resource-explorer-2 update-index-type \  
  --index-arn arn:aws:resource-explorer-2:us-east-1:123456789012:index/My-Index//  
EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111 \  
  --index-type aggregator
```

```
--arn arn:aws:resource-explorer-2:us-east-1:123456789012:index/EXAMPLE8-90ab-
cdef-fedc-EXAMPLE11111 \
--type aggregator \
--region us-east-1
```

Output:

```
{
  "Arn": "arn:aws:resource-explorer-2:us-east-1:123456789012:index/EXAMPLE8-90ab-
cdef-fedc-EXAMPLE11111",
  "LastUpdatedAt": "2022-07-13T18:41:58.799Z",
  "State": "updating",
  "Type": "aggregator"
}
```

For more information about changing an index's type, see [Turning on cross-Region search by creating an aggregator index](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [UpdateIndexType](#) in *AWS CLI Command Reference*.

update-view

The following code example shows how to use `update-view`.

AWS CLI

Example 1: To update the `IncludedProperties` field for a Resource Explorer view

The following `update-view` example updates the specified view by adding `tags` to the optional `IncludedProperties`. After running this operation, search operations that use this view include information about the tags attached to the resources that appear in the results.

```
aws resource-explorer-2 update-view \
  --included-properties Name=tags \
  --view-arn arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-View/
EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222
```

Output:

```
{
```

```

"View": {
  "Filters": {
    "FilterString": ""
  },
  "IncludedProperties": [
    {
      "Name": "tags"
    }
  ],
  "LastUpdatedAt": "2022-07-19T17:41:21.710000+00:00",
  "Owner": "123456789012",
  "Scope": "arn:aws:iam::123456789012:root",
  "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-EC2-
Only-View/EXAMPLE8-90ab-cdef-fedc-EXAMPLE11111"
}
}

```

Example 2: To update the filters attached to a view

The following `update-view` example updates the specified view to use a filter that limits results to only resource types that are associated with the Amazon EC2 service.

```

aws resource-explorer-2 update-view \
  --filters FilterString="service:ec2" \
  --view-arn arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-View/
EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222

```

Output:

```

{
  "View": {
    "Filters": {
      "FilterString": "service:ec2"
    },
    "IncludedProperties": [],
    "LastUpdatedAt": "2022-07-19T17:41:21.710000+00:00",
    "Owner": "123456789012",
    "Scope": "arn:aws:iam::123456789012:root",
    "ViewArn": "arn:aws:resource-explorer-2:us-east-1:123456789012:view/My-View/
EXAMPLE8-90ab-cdef-fedc-EXAMPLE22222"
  }
}

```

For more information about views, see [About Resource Explorer views](#) in the *AWS Resource Explorer Users Guide*.

- For API details, see [UpdateView](#) in *AWS CLI Command Reference*.

Resource Groups examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Resource Groups.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-group

The following code example shows how to use `create-group`.

AWS CLI

Example 1: To create a tag-based resource group

The following `create-group` example creates a tag-based resource group of Amazon EC2 instances in the current region. It's based on a query for resources that are tagged with the key `Name`, and the value `WebServers`. The group name is `tbq-WebServer`. The query is in a separate JSON file that is passed to the command.

```
aws resource-groups create-group \  
  --name tbq-WebServer \  
  --resource-query file://query.json
```


Contents of query.json:

```
{
  "Type": "TAG_FILTERS_1_0",
  "Query": "{\"ResourceTypeFilters\": [\"AWS::EC2::Instance\"], \"TagFilters\": [ { \"Key\": \"Name\", \"Values\": [ \"WebServers\" ] } ] }"
```

Output:

```
{
  "Group": {
    "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/tbq-WebServer",
    "Name": "tbq-WebServer"
  },
  "ResourceQuery": {
    "Type": "TAG_FILTERS_1_0",
    "Query": "{\"ResourceTypeFilters\": [\"AWS::EC2::Instance\"], \"TagFilters\": [ { \"Key\": \"Name\", \"Values\": [ \"WebServers\" ] } ] }"
```

Example 2: To create a CloudFormation stack-based resource group

The following `create-group` example creates an AWS CloudFormation stack-based resource group named `sampleCFNstackgroup`. The query includes all resources in the specified CloudFormation stack that are supported by AWS Resource Groups.

```
aws resource-groups create-group \
  --name cbq-CFNstackgroup \
  --resource-query file://query.json
```

Contents of query.json:

```
{
  "Type": "CLOUDFORMATION_STACK_1_0",
  "Query": "{\"ResourceTypeFilters\": [\"AWS::AllSupported\"], \"StackIdentifier\": \"arn:aws:cloudformation:us-west-2:123456789012:stack/MyCFNStack/1415z9z0-z39z-11z8-97z5-500z212zz6fz\"}"
```

Output:

```
{
  "Group": {
    "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/cbq-
CFNstackgroup",
    "Name": "cbq-CFNstackgroup"
  },
  "ResourceQuery": {
    "Type": "CLOUDFORMATION_STACK_1_0",
    "Query": "{\"ResourceTypeFilters\": [\"AWS::AllSupported\"], \"StackIdentifier
\": \"arn:aws:cloudformation:us-east-2:123456789012:stack/MyCFNStack/1415z9z0-
z39z-11z8-97z5-500z212zz6fz\"}"
  }
}
```

For more information, see [Create Groups](#) in the *AWS Resource Groups User Guide*.

- For API details, see [CreateGroup](#) in *AWS CLI Command Reference*.

delete-group

The following code example shows how to use `delete-group`.

AWS CLI**To update the description for a resource group**

The following `delete-group` example updates the specified resource group.

```
aws resource-groups delete-group \
  --group-name tbq-WebServer
```

Output:

```
{
  "Group": {
    "GroupArn": "arn:aws:resource-groups:us-west-2:1234567890:group/tbq-
WebServer",
    "Name": "tbq-WebServer"
  }
}
```

```
}
```

For more information, see [Delete Groups](#) in the *AWS Resource Groups User Guide*.

- For API details, see [DeleteGroup](#) in *AWS CLI Command Reference*.

get-group-query

The following code example shows how to use `get-group-query`.

AWS CLI

To get the query attached to a resource group

The following `get-group-query` example displays query attached to the specified resource group.

```
aws resource-groups get-group-query \  
  --group-name tbq-WebServer
```

Output:

```
{  
  "GroupQuery": {  
    "GroupName": "tbq-WebServer",  
    "ResourceQuery": {  
      "Type": "TAG_FILTERS_1_0",  
      "Query": "{\"ResourceTypeFilters\": [\"AWS::EC2::Instance\"], \"TagFilters  
\": [{\"Key\": \"Name\", \"Values\": [\"WebServers\"]}]}"  
    }  
  }  
}
```

- For API details, see [GetGroupQuery](#) in *AWS CLI Command Reference*.

get-group

The following code example shows how to use `get-group`.

AWS CLI

To get information about a resource group

The following `get-group` example displays details about the specified resource group. To get the query attached to the group, use `get-group-query`.

```
aws resource-groups get-group \  
  --group-name tbq-WebServer
```

Output:

```
{  
  "Group": {  
    "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/tbq-  
WebServer",  
    "Name": "tbq-WebServer",  
    "Description": "A tag-based query resource group of WebServers."  
  }  
}
```

- For API details, see [GetGroup](#) in *AWS CLI Command Reference*.

get-tags

The following code example shows how to use `get-tags`.

AWS CLI

To retrieve the tags attached to a resource group

The following `get-tags` example displays the tag key and value pairs attached to the specified resource group (the group itself, not its members).

```
aws resource-groups get-tags \  
  --arn arn:aws:resource-groups:us-west-2:123456789012:group/tbq-WebServer
```

Output:

```
{  
  "Arn": "arn:aws:resource-groups:us-west-2:123456789012:group/tbq-WebServer",  
  "Tags": {  
    "QueryType": "tags",  
    "QueryResources": "ec2-instances"  
  }  
}
```

```
}
```

- For API details, see [GetTags](#) in *AWS CLI Command Reference*.

list-group-resources

The following code example shows how to use `list-group-resources`.

AWS CLI

To list all of the resources in a resource group

Example 1: The following `list-resource-groups` example lists all of the resources that are part of the specified resource group.

```
aws resource-groups list-group-resources \  
  --group-name tbq-WebServer
```

Output:

```
{  
  "ResourceIdentifiers": [  
    {  
      "ResourceArn": "arn:aws:ec2:us-west-2:123456789012:instance/  
i-09f77fa38c12345ab",  
      "ResourceType": "AWS::EC2::Instance"  
    }  
  ]  
}
```

Example 2: The following example lists all of the resources in the group that also have a 'resource-type' of the 'AWS::EC2::Instance':

```
aws resource-groups list-group-resources --group-name tbq-WebServer --filters  
Name=resource-type,Values=AWS::EC2::Instance
```

- For API details, see [ListGroupResources](#) in *AWS CLI Command Reference*.

list-groups

The following code example shows how to use `list-groups`.

AWS CLI

To list the available resource groups

The following `list-groups` example displays a list of all of the resource groups.

```
aws resource-groups list-groups
```

Output:

```
{
  "GroupIdentifiers": [
    {
      "GroupName": "tbq-WebServer",
      "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/tbq-WebServer3"
    },
    {
      "GroupName": "cbq-CFNStackQuery",
      "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/cbq-CFNStackQuery"
    }
  ],
  "Groups": [
    {
      "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/tbq-WebServer",
      "Name": "tbq-WebServer"
    },
    {
      "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/cbq-CFNStackQuery",
      "Name": "cbq-CFNStackQuery"
    }
  ]
}
```

- For API details, see [ListGroups](#) in *AWS CLI Command Reference*.

`list-resource-groups`

The following code example shows how to use `list-resource-groups`.

AWS CLI

To list all of the resources in a resource group

The following `list-resource-groups` example lists all of the resources that are part of the specified resource group.

```
aws resource-groups list-group-resources \  
  --group-name tbq-WebServer
```

Output:

```
{  
  "ResourceIdentifiers": [  
    {  
      "ResourceArn": "arn:aws:ec2:us-west-2:123456789012:instance/  
i-09f77fa38c12345ab",  
      "ResourceType": "AWS::EC2::Instance"  
    }  
  ]  
}
```

- For API details, see [ListResourceGroups](#) in *AWS CLI Command Reference*.

put-group-configuration

The following code example shows how to use `put-group-configuration`.

AWS CLI

To attach a service configuration to a resource group

Example 1: The following `put-group-configuration` example specifies that the resource group is to contain only Amazon EC2 capacity reservations for instances in the C5 or M5 families.

```
aws resource-groups put-group-configuration \  
  --group MyTestGroup \  
  --configuration file://config.json
```

Contents of `config.json`:

```
[
  {
    "Type": "AWS::EC2::HostManagement",
    "Parameters": [
      {
        "Name": "allowed-host-families",
        "Values": [ "c5", "m5" ]
      },
      {
        "Name": "any-host-based-license-configuration",
        "Values": [ "true" ]
      }
    ]
  },
  {
    "Type": "AWS::ResourceGroups::Generic",
    "Parameters": [
      {
        "Name": "allowed-resource-types",
        "Values": [ "AWS::EC2::Host" ]
      },
      {
        "Name": "deletion-protection",
        "Values": [ "UNLESS_EMPTY" ]
      }
    ]
  }
]
```

This command produces no output if successful.

For more information, see [Service configurations for resource groups](#) in the *Resource Groups API Reference Guide*.

- For API details, see [PutGroupConfiguration](#) in *AWS CLI Command Reference*.

search-resources

The following code example shows how to use search-resources.

AWS CLI

To find resources that match a query

The following `search-resources` example retrieves a list of all AWS resources that match the specified query.

```
aws resource-groups search-resources \  
  --resource-query file://query.json
```

Contents of `query.json`:

```
{  
  "Type": "TAG_FILTERS_1_0",  
  "Query": "{\"ResourceTypeFilters\": [\"AWS::EC2::Instance\"], \"TagFilters\":  
  [{\"Key\": \"Patch Group\", \"Values\": [\"Dev\"]}]}"  
}
```

Output:

```
{  
  "ResourceIdentifiers": [  
    {  
      "ResourceArn": "arn:aws:ec2:us-west-2:123456789012:instance/  
i-01a23bc45d67890ef",  
      "ResourceType": "AWS::EC2::Instance"  
    }  
  ]  
}
```

- For API details, see [SearchResources](#) in *AWS CLI Command Reference*.

tag

The following code example shows how to use `tag`.

AWS CLI

To attach a tag to a resource group

The following `tag` example attaches the specified tag key and value pairs to the specified resource group (the group itself, not its members).

```
aws resource-groups tag \  
  --tags QueryType=tags,QueryResources=ec2-instances \  
  --resource-group-id <resource-group-id>
```

```
--arn arn:aws:resource-groups:us-west-2:128716708097:group/tbq-WebServer
```

Output:

```
{
  "Arn": "arn:aws:resource-groups:us-west-2:128716708097:group/tbq-WebServer",
  "Tags": {
    "QueryType": "tags",
    "QueryResources": "ec2-instances"
  }
}
```

For more information, see [Manage tags](#) in the *AWS Resource Groups User Guide*.

- For API details, see [Tag](#) in *AWS CLI Command Reference*.

untag

The following code example shows how to use untag.

AWS CLI

To remove tags from a resource group

The following untags example removes any tag with the specified key from the resource group itself, not its members.

```
aws resource-groups untag \
  --arn arn:aws:resource-groups:us-west-2:123456789012:group/tbq-WebServer \
  --keys QueryType
```

Output:

```
{
  "Arn": "arn:aws:resource-groups:us-west-2:123456789012:group/tbq-WebServer",
  "Keys": [
    "QueryType"
  ]
}
```

For more information, see [Manage tags](#) in the *AWS Resource Groups User Guide*.

- For API details, see [Untag](#) in *AWS CLI Command Reference*.

update-group-query

The following code example shows how to use update-group-query.

AWS CLI

Example 1: To update the query for a tag-based resource group

The following update-group-query example updates the query attached to the specified tag-based resource group.

```
aws resource-groups update-group-query \  
  --group-name tbq-WebServer \  
  --resource-query '{"Type":"TAG_FILTERS_1_0", "Query":{"ResourceTypeFilters\  
["AWS::EC2::Instance"],\TagFilters":{"Key":"Name", "Values":["WebServers\  
"]}}}'
```

Output:

```
{  
  "Group": {  
    "GroupArn": "arn:aws:resource-groups:us-east-2:123456789012:group/tbq-  
WebServer",  
    "Name": "tbq-WebServer"  
  },  
  "ResourceQuery": {  
    "Type": "TAG_FILTERS_1_0",  
    "Query": "{\ResourceTypeFilters\":[\AWS::EC2::Instance],\TagFilters\  
[{\Key\:\Name\", \Values\:[\WebServers\]}]}"  
  }  
}
```

For more information, see [Update Groups](#) in the *AWS Resource Groups User Guide*.

Example 2: To update the query for a CloudFormation stack-based resource group

The following update-group-query example updates the query attached to the specified AWS CloudFormation stack-based resource group.

```
aws resource-groups update-group-query \  
  --group-name cbq-CFNstackgroup \  
  --resource-query '{"Type": "CLOUDFORMATION_STACK_1_0", "Query":  
"{\ResourceTypeFilters\":[\AWS::AllSupported],\StackIdentifier\  
":
```

```
\\"arn:aws:cloudformation:us-west-2:123456789012:stack/MyCFNStack/1415z9z0-z39z-11z8-97z5-500z212zz6fz\\"}"}'
```

Output:

```
{
  "Group": {
    "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/cbq-CFNstackgroup",
    "Name": "cbq-CFNstackgroup"
  },
  "ResourceQuery": {
    "Type": "CLOUDFORMATION_STACK_1_0",
    "Query": "{\"ResourceTypeFilters\": [\"AWS::AllSupported\"], \"StackIdentifier\": \"arn:aws:cloudformation:us-west-2:123456789012:stack/MyCFNStack/1415z9z0-z39z-11z8-97z5-500z212zz6fz\"}"
  }
}
```

For more information, see [Update Groups](#) in the *AWS Resource Groups User Guide*.

- For API details, see [UpdateGroupQuery](#) in *AWS CLI Command Reference*.

update-group

The following code example shows how to use `update-group`.

AWS CLI**To update the description for a resource group**

The following `update-group` example updates the description for the specified resource group.

```
aws resource-groups update-group \
  --group-name tbq-WebServer \
  --description "Resource group for all web server resources."
```

Output:

```
{
  "Group": {
```

```
    "GroupArn": "arn:aws:resource-groups:us-west-2:123456789012:group/tbq-
WebServer",
    "Name": "tbq-WebServer"
    "Description": "Resource group for all web server resources."
  }
}
```

For more information, see [Update Groups](#) in the *AWS Resource Groups User Guide*.

- For API details, see [UpdateGroup](#) in *AWS CLI Command Reference*.

Resource Groups Tagging API examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Resource Groups Tagging API.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

get-resources

The following code example shows how to use `get-resources`.

AWS CLI

To get a list of tagged resources

The following `get-resources` example displays a list of resources in the account that are tagged with the specified key name and value.

```
aws resourcegroupstaggingapi get-resources \  
  --tag-filters Key=Environment,Values=Production \  
  --tags-per-page 100
```

Output:

```
{  
  "ResourceTagMappingList": [  
    {  
      "ResourceARN": " arn:aws:inspector:us-west-2:123456789012:target/0-  
nvgVhaxX/template/0-7sbz2Kz0",  
      "Tags": [  
        {  
          "Key": "Environment",  
          "Value": "Production"  
        }  
      ]  
    }  
  ]  
}
```

For more information, see [GetResources](#) in the *Resource Groups Tagging API Reference*.

- For API details, see [GetResources](#) in *AWS CLI Command Reference*.

get-tag-keys

The following code example shows how to use `get-tag-keys`.

AWS CLI

To get a list of all tag keys

The following `get-tag-keys` example retrieves the list of all tag key names used by resources in the account.

```
aws resourcegroupstaggingapi get-tag-keys
```

Output:

```
{
```

```
"TagKeys": [  
  "Environment",  
  "CostCenter",  
  "Department"  
]  
}
```

For more information, see [GetTagKeys](#) in the *Resource Groups Tagging API Reference*.

- For API details, see [GetTagKeys](#) in *AWS CLI Command Reference*.

get-tag-values

The following code example shows how to use `get-tag-values`.

AWS CLI

To get a list of all tag values

The following `get-tag-values` example displays all of the values used for the specified key for all resources in the

```
aws resourcegroupstaggingapi get-tag-values \  
  --key=Environment
```

Output:

```
{  
  "TagValues": [  
    "Alpha",  
    "Gamma",  
    "Production"  
  ]  
}
```

For more information, see [GetTagValues](#) in the *Resource Groups Tagging API Reference*.

- For API details, see [GetTagValues](#) in *AWS CLI Command Reference*.

tag-resources

The following code example shows how to use `tag-resources`.

AWS CLI

To attach a tag to a resource

The following `tag-resources` example tags the specified resource with a key name and value.

```
aws resourcegroupstaggingapi tag-resources \  
  --resource-arn-list arn:aws:s3:::MyProductionBucket \  
  --tags Environment=Production,CostCenter=1234
```

Output:

```
{  
  "FailedResourcesMap": {}  
}
```

For more information, see [TagResources](#) in the *Resource Groups Tagging API Reference*.

- For API details, see [TagResources](#) in *AWS CLI Command Reference*.

untag-resources

The following code example shows how to use `untag-resources`.

AWS CLI

To remove a tag from a resource

The following `untag-resources` example removes the specified tag keys and any associated values from the specified resource.

```
aws resourcegroupstaggingapi untag-resources \  
  --resource-arn-list arn:aws:s3:::awsexamplebucket \  
  --tag-keys Environment CostCenter
```

Output:

```
{  
  "FailedResourcesMap": {}  
}
```


For more information, see [UntagResources](#) in the *Resource Groups Tagging API Reference*.

- For API details, see [UntagResources](#) in *AWS CLI Command Reference*.

AWS RoboMaker examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS RoboMaker.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

batch-describe-simulation-job

The following code example shows how to use `batch-describe-simulation-job`.

AWS CLI

To batch describe simulation jobs

The following `batch-describe-simulation-job` example retrieves details for the three specified simulation jobs.

Command:

```
aws robomaker batch-describe-simulation-job \  
--job arn:aws:robomaker:us-west-2:111111111111:simulation-job/sim-66bbb3gpxm8x  
arn:aws:robomaker:us-west-2:111111111111:simulation-job/sim-p0cpdrrwng2n  
arn:aws:robomaker:us-west-2:111111111111:simulation-job/sim-g8h6tg1mblgw
```

Output:

```

{
  "jobs": [
    {
      "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-job/sim-66bbb3gpxm8x",
      "status": "Completed",
      "lastUpdatedAt": 1548959178.0,
      "failureBehavior": "Continue",
      "clientRequestToken": "6020408e-b05c-4310-9f13-4ed71c5221ed",
      "outputLocation": {
        "s3Bucket": "awsrobomakerobjecttracker-111111111-bundlesbucket-2lk584kiq1oa",
        "s3Prefix": "output"
      },
      "maxJobDurationInSeconds": 3600,
      "simulationTimeMillis": 0,
      "iamRole": "arn:aws:iam::111111111111:role/AWSRoboMakerObjectTracker-154895-SimulationJobRole-14D5ASA7PQE3A",
      "simulationApplications": [
        {
          "application": "arn:aws:robomaker:us-west-2:111111111111:simulation-application/AWSRoboMakerObjectTracker-1548959046124_NPvyfcatq/1548959170096",
          "applicationVersion": "$LATEST",
          "launchConfig": {
            "packageName": "object_tracker_simulation",
            "launchFile": "local_training.launch",
            "environmentVariables": {
              "MARKOV_PRESET_FILE": "object_tracker.py",
              "MODEL_S3_BUCKET": "awsrobomakerobjecttracker-111111111-bundlesbucket-2lk584kiq1oa",
              "MODEL_S3_PREFIX": "model-store",
              "ROS_AWS_REGION": "us-west-2"
            }
          }
        }
      ]
    },
    {
      "tags": {},
      "vpcConfig": {
        "subnets": [
          "subnet-716dd52a",
          "subnet-43c22325",
        ]
      }
    }
  ]
}

```

```

        "subnet-3f526976"
      ],
      "securityGroups": [
        "sg-3fb40545"
      ],
      "vpcId": "vpc-99895eff",
      "assignPublicIp": true
    }
  },
  {
    "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-job/sim-
p0cpdrrwng2n",
    "status": "Completed",
    "lastUpdatedAt": 1548168817.0,
    "failureBehavior": "Continue",
    "clientRequestToken": "e4a23e75-f9a7-411d-835f-21881c82c58b",
    "outputLocation": {
      "s3Bucket": "awsrobomakercloudwatch-111111111111-
bundlesbucket-14e5s9jvwtmv7",
      "s3Prefix": "output"
    },
    "maxJobDurationInSeconds": 3600,
    "simulationTimeMillis": 0,
    "iamRole": "arn:aws:iam::111111111111:role/
AWSRoboMakerCloudWatch-154766341-SimulationJobRole-G00BWTQ8YBG6",
    "robotApplications": [
      {
        "application": "arn:aws:robomaker:us-west-2:111111111111:robot-
application/AWSRoboMakerCloudWatch-1547663411642_NZbpqEJ3T/1547663517377",
        "applicationVersion": "$LATEST",
        "launchConfig": {
          "packageName": "cloudwatch_robot",
          "launchFile": "await_commands.launch",
          "environmentVariables": {
            "LAUNCH_ID": "1548168752173",
            "ROS_AWS_REGION": "us-west-2"
          }
        }
      }
    ]
  },
  "simulationApplications": [
    {

```

```

        "application": "arn:aws:robomaker:us-
west-2:111111111111:simulation-application/
AWSRoboMakerCloudWatch-1547663411642_0LI6D1h6/1547663521470",
        "applicationVersion": "$LATEST",
        "launchConfig": {
            "packageName": "cloudwatch_simulation",
            "launchFile": "bookstore_turtlebot_navigation.launch",
            "environmentVariables": {
                "LAUNCH_ID": "1548168752173",
                "ROS_AWS_REGION": "us-west-2",
                "TURTLEBOT3_MODEL": "waffle_pi"
            }
        }
    },
    "tags": {},
    "vpcConfig": {
        "subnets": [
            "subnet-716dd52a",
            "subnet-43c22325",
            "subnet-3f526976"
        ],
        "securityGroups": [
            "sg-3fb40545"
        ],
        "vpcId": "vpc-99895eff",
        "assignPublicIp": true
    }
},
{
    "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-job/sim-
g8h6tglmblgw",
    "status": "Canceled",
    "lastUpdatedAt": 1546543442.0,
    "failureBehavior": "Fail",
    "clientRequestToken": "d796bbb4-2a2c-1abc-f2a9-0d9e547d853f",
    "outputLocation": {
        "s3Bucket": "sample-bucket",
        "s3Prefix": "SimulationLog_115490482698"
    },
    "maxJobDurationInSeconds": 28800,
    "simulationTimeMillis": 0,
    "iamRole": "arn:aws:iam:111111111111:role/RoboMakerSampleTheFirst",
    "robotApplications": [

```

```

        {
            "application": "arn:aws:robomaker:us-west-2:111111111111:robot-
application/RoboMakerHelloWorldRobot/1546541208251",
            "applicationVersion": "$LATEST",
            "launchConfig": {
                "packageName": "hello_world_robot",
                "launchFile": "rotate.launch"
            }
        }
    ],
    "simulationApplications": [
        {
            "application": "arn:aws:robomaker:us-
west-2:111111111111:simulation-application/
RoboMakerHelloWorldSimulation/1546541198985",
            "applicationVersion": "$LATEST",
            "launchConfig": {
                "packageName": "hello_world_simulation",
                "launchFile": "empty_world.launch"
            }
        }
    ],
    "tags": {}
}
],
"unprocessedJobs": []
}

```

- For API details, see [BatchDescribeSimulationJob](#) in *AWS CLI Command Reference*.

cancel-simulation-job

The following code example shows how to use `cancel-simulation-job`.

AWS CLI

To cancel a simulation job

The following `cancel-simulation-job` example cancels the specified simulation job.

```

aws robomaker cancel-simulation-job \
  --job arn:aws:robomaker:us-west-2:111111111111:simulation-job/sim-66bbb3gpxm8x

```

- For API details, see [CancelSimulationJob](#) in *AWS CLI Command Reference*.

create-deployment-job

The following code example shows how to use `create-deployment-job`.

AWS CLI

To create a deployment job

This example creates a deployment job for fleet `MyFleet`. It includes an environment variable named `"ENVIRONMENT"`. It also attaches a tag named `"Region"`.

Command:

```
aws robomaker create-deployment-job --deployment-config
concurrentDeploymentPercentage=20, failureThresholdPercentage=25
--fleet arn:aws:robomaker:us-west-2:111111111111:deployment-fleet/
Trek/1539894765711 --tags Region=West --deployment-application-configs
application=arn:aws:robomaker:us-west-2:111111111111:robot-application/
RoboMakerVoiceInteractionRobot/1546537110575, applicationVersion=1, launchConfig={environmentV
```

Output:

```
{
  "arn": "arn:aws:robomaker:us-west-2:111111111111:deployment-job/sim-0974h36s4v0t",
  "fleet": "arn:aws:robomaker:us-west-2:111111111111:deployment-fleet/
MyFleet/1539894765711",
  "status": "Pending",
  "deploymentApplicationConfigs": [
    {
      "application": "arn:aws:robomaker:us-west-2:111111111111:robot-
application/RoboMakerVoiceInteractionRobot/1546537110575",
      "applicationVersion": "1",
      "launchConfig": {
        "packageName": "voice_interaction_robot",
        "launchFile": "await_commands.launch",
        "environmentVariables": {
          "ENVIRONMENT": "Beta"
        }
      }
    }
  ]
}
```

```
],
"createdAt": 1550770236.0,
"deploymentConfig": {
  "concurrentDeploymentPercentage": 20,
  "failureThresholdPercentage": 25
},
"tags": {
  "Region": "West"
}
}
```

- For API details, see [CreateDeploymentJob](#) in *AWS CLI Command Reference*.

create-fleet

The following code example shows how to use `create-fleet`.

AWS CLI

To create a fleet

This example creates a fleet. It attaches a tag named `Region`.

Command:

```
aws robomaker create-fleet --name MyFleet --tags Region=East
```

Output:

```
{
  "arn": "arn:aws:robomaker:us-west-2:111111111111:deployment-fleet/MyOtherFleet/1550771394395",
  "name": "MyFleet",
  "createdAt": 1550771394.0,
  "tags": {
    "Region": "East"
  }
}
```

- For API details, see [CreateFleet](#) in *AWS CLI Command Reference*.

create-robot-application-version

The following code example shows how to use `create-robot-application-version`.

AWS CLI

To create a robot application version

This example creates a robot application version.

Command:

```
aws robomaker create-robot-application-version --application arn:aws:robomaker:us-west-2:111111111111:robot-application/MyRobotApplication/1551201873931
```

Output:

```
{
  "arn": "arn:aws:robomaker:us-west-2:111111111111:robot-application/MyRobotApplication/1551201873931",
  "name": "MyRobotApplication",
  "version": "1",
  "sources": [
    {
      "s3Bucket": "my-bucket",
      "s3Key": "my-robot-application.tar.gz",
      "etag": "f8cf5526f1c6e7b3a72c3ed3f79c5493-70",
      "architecture": "ARMHF"
    }
  ],
  "robotSoftwareSuite": {
    "name": "ROS",
    "version": "Kinetic"
  },
  "lastUpdatedAt": 1551201873.0,
  "revisionId": "9986bb8d-a695-4ab4-8810-9f4a74d1aa00"
  "tags": {}
}
```

- For API details, see [CreateRobotApplicationVersion](#) in *AWS CLI Command Reference*.

create-robot-application

The following code example shows how to use create-robot-application.

AWS CLI

To create a robot application

This example creates a robot application.

Command:

```
aws robomaker create-robot-application --name MyRobotApplication --sources
s3Bucket=my-bucket,s3Key=my-robot-application.tar.gz,architecture=X86_64 --robot-
software-suite name=ROS,version=Kinetic
```

Output:

```
{
  "arn": "arn:aws:robomaker:us-west-2:111111111111:robot-application/
MyRobotApplication/1551201873931",
  "name": "MyRobotApplication",
  "version": "$LATEST",
  "sources": [
    {
      "s3Bucket": "my-bucket",
      "s3Key": "my-robot-application.tar.gz",
      "architecture": "ARMHF"
    }
  ],
  "robotSoftwareSuite": {
    "name": "ROS",
    "version": "Kinetic"
  },
  "lastUpdatedAt": 1551201873.0,
  "revisionId": "1f3cb539-9239-4841-a656-d3efcffa07e1",
  "tags": {}
}
```

- For API details, see [CreateRobotApplication](#) in *AWS CLI Command Reference*.

create-robot

The following code example shows how to use create-robot.

AWS CLI

To create a robot

This example creates a robot. It uses the ARMHF architecture. It also attaches a tag named Region.

Command:

```
aws robomaker create-robot --name MyRobot --architecture ARMHF --greengrass-group-id
0f728a3c-7dbf-4a3e-976d-d16a8360caba --tags Region=East
```

Output:

```
{
  "arn": "arn:aws:robomaker:us-west-2:111111111111:robot/MyRobot/1550772324398",
  "name": "MyRobot",
  "createdAt": 1550772325.0,
  "greengrassGroupId": "0f728a3c-7dbf-4a3e-976d-d16a8360caba",
  "architecture": "ARMHF",
  "tags": {
    "Region": "East"
  }
}
```

- For API details, see [CreateRobot](#) in *AWS CLI Command Reference*.

create-simulation-application-version

The following code example shows how to use create-simulation-application-version.

AWS CLI

To create a simulation application version

This example creates a robot application version.

Command:

```
aws robomaker create-simulation-application-version --application
arn:aws:robomaker:us-west-2:111111111111:robot-application/
MySimulationApplication/1551203427605
```

Output:

```
{
  "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-application/
MyRobotApplication/1551203427605",
  "name": "MyRobotApplication",
  "version": "1",
  "sources": [
    {
      "s3Bucket": "my-bucket",
      "s3Key": "my-simulation-application.tar.gz",
      "etag": "00d8a94ff113856688c4fce618ae0f45-94",
      "architecture": "X86_64"
    }
  ],
  "simulationSoftwareSuite": {
    "name": "Gazebo",
    "version": "7"
  },
  "robotSoftwareSuite": {
    "name": "ROS",
    "version": "Kinetic"
  },
  "renderingEngine": {
    "name": "OGRE",
    "version": "1.x"
  },
  "lastUpdatedAt": 1551203853.0,
  "revisionId": "ee753e53-519c-4d37-895d-65e79bcd1914",
  "tags": {}
}
```

- For API details, see [CreateSimulationApplicationVersion](#) in *AWS CLI Command Reference*.

create-simulation-application

The following code example shows how to use create-simulation-application.

AWS CLI

To create a simulation application

This example creates a simulation application.

Command:

```
aws robomaker create-simulation-application --name MyRobotApplication --sources
s3Bucket=my-bucket,s3Key=my-simulation-application.tar.gz,architecture=ARMHF
--robot-software-suite name=ROS,version=Kinetic --simulation-software-suite
name=Gazebo,version=7 --rendering-engine name=OGRE,version=1.x
```

Output:

```
{
  "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-application/
MyRobotApplication/1551203301792",
  "name": "MyRobotApplication",
  "version": "$LATEST",
  "sources": [
    {
      "s3Bucket": "my-bucket",
      "s3Key": "my-simulation-application.tar.gz",
      "architecture": "X86_64"
    }
  ],
  "simulationSoftwareSuite": {
    "name": "Gazebo",
    "version": "7"
  },
  "robotSoftwareSuite": {
    "name": "ROS",
    "version": "Kinetic"
  },
  "renderingEngine": {
    "name": "OGRE",
    "version": "1.x"
  },
  "lastUpdatedAt": 1551203301.0,
  "revisionId": "ee753e53-519c-4d37-895d-65e79bcd1914",
  "tags": {}
}
```

- For API details, see [CreateSimulationApplication](#) in *AWS CLI Command Reference*.

create-simulation-job

The following code example shows how to use `create-simulation-job`.

AWS CLI

To create a simulation job

This example creates a simulation job. It uses a robot application and a simulation application.

Command:

```
aws robomaker create-simulation-job --max-job-duration-
in-seconds 3600 --iam-role arn:aws:iam::111111111111:role/
AWSRoboMakerCloudWatch-154766341-SimulationJobRole-G00BWTQ8YBG6 --robot-
applications application=arn:aws:robomaker:us-west-2:111111111111:robot-application/
MyRobotApplication/1551203485821,launchConfig={packageName=hello_world_robot,launchFile=rota
--simulation-applications application=arn:aws:robomaker:us-
west-2:111111111111:simulation-application/
MySimulationApplication/1551203427605,launchConfig={packageName=hello_world_simulation,launch
--tags Region=North
```

Output:

```
{
  "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-job/sim-w7m68wpr05h8",
  "status": "Pending",
  "lastUpdatedAt": 1551213837.0,
  "failureBehavior": "Fail",
  "clientRequestToken": "b283ccce-e468-43ee-8642-be76a9d69f15",
  "maxJobDurationInSeconds": 3600,
  "simulationTimeMillis": 0,
  "iamRole": "arn:aws:iam::111111111111:role/MySimulationRole",
  "robotApplications": [
    {
      "application": "arn:aws:robomaker:us-west-2:111111111111:robot-
application/MyRobotApplication/1551203485821",
      "applicationVersion": "$LATEST",
      "launchConfig": {
        "packageName": "hello_world_robot",
```

```

        "launchFile": "rotate.launch"
    }
}
],
"simulationApplications": [
    {
        "application": "arn:aws:robomaker:us-west-2:111111111111:simulation-
application/MySimulationApplication/1551203427605",
        "applicationVersion": "$LATEST",
        "launchConfig": {
            "packageName": "hello_world_simulation",
            "launchFile": "empty_world.launch"
        }
    }
],
"tags": {
    "Region": "North"
}
}
}

```

- For API details, see [CreateSimulationJob](#) in *AWS CLI Command Reference*.

delete-fleet

The following code example shows how to use `delete-fleet`.

AWS CLI

To delete a fleet

This example deletes a fleet.

Command:

```
aws robomaker delete-fleet --fleet arn:aws:robomaker:us-
west-2:111111111111:deployment-fleet/MyFleet/1550771394395
```

- For API details, see [DeleteFleet](#) in *AWS CLI Command Reference*.

delete-robot-application

The following code example shows how to use `delete-robot-application`.

AWS CLI

To delete a robot application

This example deletes a robot application.

Command:

```
aws robomaker delete-robot-application --application arn:aws:robomaker:us-west-2:111111111111:robot-application/MyRobotApplication/1551203485821
```

- For API details, see [DeleteRobotApplication](#) in *AWS CLI Command Reference*.

delete-robot

The following code example shows how to use delete-robot.

AWS CLI

To delete a robot

This example deletes a robot.

Command:

```
aws robomaker delete-robot --robot arn:aws:robomaker:us-west-2:111111111111:robot/MyRobot/1540829698778
```

- For API details, see [DeleteRobot](#) in *AWS CLI Command Reference*.

delete-simulation-application

The following code example shows how to use delete-simulation-application.

AWS CLI

To delete a simulation application

This example deletes a simulation application.

Command:

```
aws robomaker delete-simulation-application --application arn:aws:robomaker:us-west-2:111111111111:simulation-application/MySimulationApplication/1551203427605
```

- For API details, see [DeleteSimulationApplication](#) in *AWS CLI Command Reference*.

deregister-robot

The following code example shows how to use `deregister-robot`.

AWS CLI

To deregister a robot from a fleet

This example deregisters a robot from a fleet.

Command:

```
aws robomaker deregister-robot --fleet arn:aws:robomaker:us-west-2:111111111111:deployment-fleet/MyFleet/1550771358907 --robot arn:aws:robomaker:us-west-2:111111111111:robot/MyRobot/1550772324398
```

Output:

```
{
  "fleet": "arn:aws:robomaker:us-west-2:111111111111:deployment-fleet/MyFleet/1550771358907",
  "robot": "arn:aws:robomaker:us-west-2:111111111111:robot/MyRobot/1550772324398"
}
```

- For API details, see [DeregisterRobot](#) in *AWS CLI Command Reference*.

describe-deployment-job

The following code example shows how to use `describe-deployment-job`.

AWS CLI

To describe a deployment job

The following `describe-deployment-job` example retrieves the details about the specified deployment job.


```
aws robomaker describe-deployment-job \  
  --job arn:aws:robomaker:us-west-2:111111111111:deployment-job/deployment-  
xl8qssl6pbcn
```

Output:

```
{  
  "arn": "arn:aws:robomaker:us-west-2:111111111111:deployment-job/deployment-  
xl8qssl6pbcn",  
  "fleet": "arn:aws:robomaker:us-west-2:111111111111:deployment-fleet/  
Trek/1539894765711",  
  "status": "InProgress",  
  "deploymentConfig": {  
    "concurrentDeploymentPercentage": 20,  
    "failureThresholdPercentage": 25  
  },  
  "deploymentApplicationConfigs": [  
    {  
      "application": "arn:aws:robomaker:us-west-2:111111111111:robot-  
application/RoboMakerHelloWorldRobot/1546541208251",  
      "applicationVersion": "1",  
      "launchConfig": {  
        "packageName": "hello_world_robot",  
        "launchFile": "rotate.launch"  
      }  
    }  
  ],  
  "createdAt": 1551218369.0,  
  "robotDeploymentSummary": [  
    {  
      "arn": "arn:aws:robomaker:us-west-2:111111111111:robot/  
MyRobot/1540834232469",  
      "deploymentStartTime": 1551218376.0,  
      "status": "Deploying",  
      "progressDetail": {}  
    }  
  ],  
  "tags": {}  
}
```

- For API details, see [DescribeDeploymentJob](#) in *AWS CLI Command Reference*.

describe-fleet

The following code example shows how to use describe-fleet.

AWS CLI

To describe a fleet

The following describe-fleet example retrieves the details for the specified fleet.

```
aws robomaker describe-fleet \  
  --fleet arn:aws:robomaker:us-west-2:111111111111:deployment-fleet/  
  MyFleet/1550771358907
```

Output:

```
{  
  "name": "MyFleet",  
  "arn": "arn:aws:robomaker:us-west-2:111111111111:deployment-fleet/  
  MyFleet/1539894765711",  
  "robots": [  
    {  
      "arn": "arn:aws:robomaker:us-west-2:111111111111:robot/  
  MyRobot/1540834232469",  
      "createdAt": 1540834232.0  
    },  
    {  
      "arn": "arn:aws:robomaker:us-west-2:111111111111:robot/  
  MyOtherRobot/1540829698778",  
      "createdAt": 1540829698.0  
    }  
  ],  
  "createdAt": 1539894765.0,  
  "lastDeploymentStatus": "Succeeded",  
  "lastDeploymentJob": "arn:aws:robomaker:us-west-2:111111111111:deployment-job/  
  deployment-xl8qssl6pbcn",  
  "lastDeploymentTime": 1551218369.0,  
  "tags": {}  
}
```

- For API details, see [DescribeFleet](#) in *AWS CLI Command Reference*.

describe-robot-application

The following code example shows how to use `describe-robot-application`.

AWS CLI

To describe a robot application

This example describes a robot application.

Command:

```
aws robomaker describe-robot-application --application arn:aws:robomaker:us-west-2:111111111111:robot-application/MyRobotApplication/1551203485821
```

Output:

```
{
  "arn": "arn:aws:robomaker:us-west-2:111111111111:robot-application/MyRobotApplication/1551203485821",
  "name": "MyRobotApplication",
  "version": "$LATEST",
  "sources": [
    {
      "s3Bucket": "my-bucket",
      "s3Key": "my-robot-application.tar.gz",
      "architecture": "X86_64"
    }
  ],
  "robotSoftwareSuite": {
    "name": "ROS",
    "version": "Kinetic"
  },
  "revisionId": "e72efe0d-f44f-4333-b604-f6fa5c6bb50b",
  "lastUpdatedAt": 1551203485.0,
  "tags": {}
}
```

- For API details, see [DescribeRobotApplication](#) in *AWS CLI Command Reference*.

describe-robot

The following code example shows how to use `describe-robot`.

AWS CLI

To describe a robot

This example describes a robot.

Command:

```
aws robomaker describe-robot --robot arn:aws:robomaker:us-west-2:111111111111:robot/MyRobot/1550772324398
```

Output:

```
{
  "arn": "arn:aws:robomaker:us-west-2:111111111111:robot/MyRobot/1550772324398",
  "name": "MyRobot",
  "status": "Available",
  "greengrassGroupId": "0f728a3c-7dbf-4a3e-976d-d16a8360caba",
  "createdAt": 1550772325.0,
  "architecture": "ARMHF",
  "tags": {
    "Region": "East"
  }
}
```

- For API details, see [DescribeRobot](#) in *AWS CLI Command Reference*.

describe-simulation-application

The following code example shows how to use `describe-simulation-application`.

AWS CLI

To describe a simulation application

This example describes a simulation application.

Command:

```
aws robomaker describe-simulation-application --application arn:aws:robomaker:us-west-2:111111111111:simulation-application/MySimulationApplication/1551203427605
```

Output:

```
{
  "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-application/
MySimulationApplication/1551203427605",
  "name": "MySimulationApplication",
  "version": "$LATEST",
  "sources": [
    {
      "s3Bucket": "my-bucket",
      "s3Key": "my-simulation-application.tar.gz",
      "architecture": "X86_64"
    }
  ],
  "simulationSoftwareSuite": {
    "name": "Gazebo",
    "version": "7"
  },
  "robotSoftwareSuite": {
    "name": "ROS",
    "version": "Kinetic"
  },
  "renderingEngine": {
    "name": "OGRE",
    "version": "1.x"
  },
  "revisionId": "783674ab-b7b8-42d9-b01f-9373907987e5",
  "lastUpdatedAt": 1551203427.0,
  "tags": {}
}
```

- For API details, see [DescribeSimulationApplication](#) in *AWS CLI Command Reference*.

describe-simulation-job

The following code example shows how to use `describe-simulation-job`.

AWS CLI**To describe a simulation job**

This example describes a simulation job.

Command:

```
aws robomaker describe-simulation-job --job arn:aws:robomaker:us-west-2:111111111111:simulation-job/sim-pql32v7pfjy6
```

Output:

```
{
  "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-job/sim-pql32v7pfjy6",
  "status": "Running",
  "lastUpdatedAt": 1551219349.0,
  "failureBehavior": "Continue",
  "clientRequestToken": "a19ec4b5-e50d-3591-33da-c2e593c60615",
  "outputLocation": {
    "s3Bucket": "my-output-bucket",
    "s3Prefix": "output"
  },
  "maxJobDurationInSeconds": 3600,
  "simulationTimeMillis": 0,
  "iamRole": "arn:aws:iam:111111111111:role/MySimulationRole",
  "robotApplications": [
    {
      "application": "arn:aws:robomaker:us-west-2:111111111111:robot-application/MyRobotApplication/1551206341136",
      "applicationVersion": "$LATEST",
      "launchConfig": {
        "packageName": "hello_world_robot",
        "launchFile": "rotate.launch"
      }
    }
  ],
  "simulationApplications": [
    {
      "application": "arn:aws:robomaker:us-west-2:111111111111:simulation-application/MySimulationApplication/1551206347967",
      "applicationVersion": "$LATEST",
      "launchConfig": {
        "packageName": "hello_world_simulation",
        "launchFile": "empty_world.launch"
      }
    }
  ],
  "tags": {}
}
```

```
}
```

- For API details, see [DescribeSimulationJob](#) in *AWS CLI Command Reference*.

list-deployment-jobs

The following code example shows how to use `list-deployment-jobs`.

AWS CLI

To list deployment jobs

The following `list-deployment-jobs` example retrieves a list of deployment jobs.

```
aws robomaker list-deployment-jobs
```

Output:

```
{
  "deploymentJobs": [
    {
      "arn": "arn:aws:robomaker:us-west-2:111111111111:deployment-job/sim-6293szzm56rv",
      "fleet": "arn:aws:robomaker:us-west-2:111111111111:deployment-fleet/MyFleet/1539894765711",
      "status": "InProgress",
      "deploymentApplicationConfigs": [
        {
          "application": "arn:aws:robomaker:us-west-2:111111111111:robot-application/HelloWorldRobot/1546537110575",
          "applicationVersion": "1",
          "launchConfig": {
            "packageName": "hello_world_robot",
            "launchFile": "rotate.launch",
            "environmentVariables": {
              "ENVIRONMENT": "Desert"
            }
          }
        }
      ]
    },
    "deploymentConfig": {
      "concurrentDeploymentPercentage": 20,
    }
  ]
}
```

```

        "failureThresholdPercentage": 25
      },
      "createdAt": 1550689373.0
    },
    {
      "arn": "arn:aws:robomaker:us-west-2:111111111111:deployment-job/
deployment-4w4g69p25zdb",
      "fleet": "arn:aws:robomaker:us-west-2:111111111111:deployment-fleet/
MyFleet/1539894765711",
      "status": "Pending",
      "deploymentApplicationConfigs": [
        {
          "application": "arn:aws:robomaker:us-west-2:111111111111:robot-
application/AWSRoboMakerHelloWorld-1544562726923_YGHM_sh5M/1544562822877",
          "applicationVersion": "1",
          "launchConfig": {
            "packageName": "fail",
            "launchFile": "fail"
          }
        }
      ],
      "deploymentConfig": {
        "concurrentDeploymentPercentage": 20,
        "failureThresholdPercentage": 25
      },
      "failureReason": "",
      "failureCode": "",
      "createdAt": 1544719763.0
    }
  ]
}

```

- For API details, see [ListDeploymentJobs](#) in *AWS CLI Command Reference*.

list-fleets

The following code example shows how to use `list-fleets`.

AWS CLI

To list fleets

This example lists fleets. A maximum of 20 fleets will be returned.

Command:

```
aws robomaker list-fleets --max-items 20
```

Output:

```
{
  "fleetDetails": [
    {
      "name": "Trek",
      "arn": "arn:aws:robomaker:us-west-2:111111111111:deployment-fleet/MyFleet/1539894765711",
      "createdAt": 1539894765.0,
      "lastDeploymentStatus": "Failed",
      "lastDeploymentJob": "arn:aws:robomaker:us-west-2:111111111111:deployment-job/deployment-4w4g69p25zdb",
      "lastDeploymentTime": 1544719763.0
    }
  ]
}
```

- For API details, see [ListFleets](#) in *AWS CLI Command Reference*.

list-robot-applications

The following code example shows how to use `list-robot-applications`.

AWS CLI**To list robot applications**

This example lists robot applications. Results are limited to 20 robot applications.

Command:

```
aws robomaker list-robot-applications --max-results 20
```

Output:

```
{
```

```
"robotApplicationSummaries": [  
  {  
    "name": "MyRobot",  
    "arn": "arn:aws:robomaker:us-west-2:111111111111:robot-application/  
MyRobot/1546537110575",  
    "version": "$LATEST",  
    "lastUpdatedAt": 1546540372.0  
  },  
  {  
    "name": "AnotherRobot",  
    "arn": "arn:aws:robomaker:us-west-2:111111111111:robot-application/  
AnotherRobot/1546541208251",  
    "version": "$LATEST",  
    "lastUpdatedAt": 1546541208.0  
  },  
  {  
    "name": "MySuperRobot",  
    "arn": "arn:aws:robomaker:us-west-2:111111111111:robot-application/  
MySuperRobot/1547663517377",  
    "version": "$LATEST",  
    "lastUpdatedAt": 1547663517.0  
  }  
]  
}
```

- For API details, see [ListRobotApplications](#) in *AWS CLI Command Reference*.

list-robots

The following code example shows how to use `list-robots`.

AWS CLI

To list robots

This example lists robots. A maximum of 20 robots will be returned.

Command:

```
aws robomaker list-robots --max-results 20
```

Output:

```
{
  "robots": [
    {
      "arn": "arn:aws:robomaker:us-west-2:111111111111:robot/Robot100/1544035373264",
      "name": "Robot100",
      "status": "Available",
      "createdAt": 1544035373.0,
      "architecture": "X86_64"
    },
    {
      "arn": "arn:aws:robomaker:us-west-2:111111111111:robot/Robot101/1542146976587",
      "name": "Robot101",
      "status": "Available",
      "createdAt": 1542146976.0,
      "architecture": "X86_64"
    },
    {
      "arn": "arn:aws:robomaker:us-west-2:111111111111:robot/Robot102/1540834232469",
      "fleetArn": "arn:aws:robomaker:us-west-2:111111111111:deployment-fleet/Trek/1539894765711",
      "status": "Available",
      "createdAt": 1540834232.0,
      "architecture": "X86_64",
      "lastDeploymentJob": "arn:aws:robomaker:us-west-2:111111111111:deployment-job/deployment-jb007b75gl5f",
      "lastDeploymentTime": 1550689533.0
    },
    {
      "arn": "arn:aws:robomaker:us-west-2:111111111111:robot/MyRobot/1540829698778",
      "name": "MyRobot",
      "status": "Registered",
      "createdAt": 1540829698.0,
      "architecture": "X86_64"
    }
  ]
}
```

- For API details, see [ListRobots](#) in *AWS CLI Command Reference*.

list-simulation-applications

The following code example shows how to use `list-simulation-applications`.

AWS CLI

To list simulation applications

This example lists simulation applications. A maximum of 20 simulation applications will be returned.

Command:

```
aws robomaker list-simulation-applications --max-results 20
```

Output:

```
{
  "simulationApplicationSummaries": [
    {
      "name": "AWSRoboMakerObjectTracker-1548959046124_NPvyfcatq",
      "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-application/AWSRoboMakerObjectTracker-1548959046124_NPvyfcatq/1548959170096",
      "version": "$LATEST",
      "lastUpdatedAt": 1548959170.0
    },
    {
      "name": "RoboMakerHelloWorldSimulation",
      "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-application/RoboMakerHelloWorldSimulation/1546541198985",
      "version": "$LATEST",
      "lastUpdatedAt": 1546541198.0
    },
    {
      "name": "RoboMakerObjectTrackerSimulation",
      "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-application/RoboMakerObjectTrackerSimulation/1545846795615",
      "version": "$LATEST",
      "lastUpdatedAt": 1545847405.0
    },
    {
      "name": "RoboMakerVoiceInteractionSimulation",
      "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-application/RoboMakerVoiceInteractionSimulation/1546537100507",

```

```

        "version": "$LATEST",
        "lastUpdatedAt": 1546540352.0
    },
    {
        "name": "AWSRoboMakerCloudWatch-1547663411642_0LI6D1h6",
        "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-application/
AWSRoboMakerCloudWatch-1547663411642_0LI6D1h6/1547663521470",
        "version": "$LATEST",
        "lastUpdatedAt": 1547663521.0
    },
    {
        "name": "AWSRoboMakerDeepRacer-1545848257672_1YZCaieQ-",
        "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-application/
AWSRoboMakerDeepRacer-1545848257672_1YZCaieQ-/1545848370525",
        "version": "$LATEST",
        "lastUpdatedAt": 1545848370.0
    }
]
}

```

- For API details, see [ListSimulationApplications](#) in *AWS CLI Command Reference*.

list-simulation-jobs

The following code example shows how to use `list-simulation-jobs`.

AWS CLI

To list simulation jobs

This example lists simulation jobs.

Command:

```
aws robomaker list-simulation-jobs
```

Output:

```

{
  "simulationJobSummaries": [
    {
      "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-job/
sim-66bbb3gpxm8x",

```

```
    "lastUpdatedAt": 1548959178.0,
    "status": "Completed",
    "simulationApplicationNames": [
      "AWSRoboMakerObjectTracker-1548959046124_NPvyfcatq"
    ],
    "robotApplicationNames": [
      null
    ]
  },
  {
    "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-job/sim-
b27c4rkrtzcx",
    "lastUpdatedAt": 1543514088.0,
    "status": "Canceled",
    "simulationApplicationNames": [
      "AWSRoboMakerPersonDetection-1543513948280_T8rHW2_lu"
    ],
    "robotApplicationNames": [
      "AWSRoboMakerPersonDetection-1543513948280_EYaMT0mYb"
    ]
  },
  {
    "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-job/
sim-51vxjbyz4q8t",
    "lastUpdatedAt": 1543508858.0,
    "status": "Canceled",
    "simulationApplicationNames": [
      "AWSRoboMakerCloudWatch-1543504747391_1FF9ZQyx6"
    ],
    "robotApplicationNames": [
      "AWSRoboMakerCloudWatch-1543504747391_axbYa3S3K"
    ]
  },
  {
    "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-job/sim-
kgf1fqxflqbx",
    "lastUpdatedAt": 1543504862.0,
    "status": "Completed",
    "simulationApplicationNames": [
      "AWSRoboMakerCloudWatch-1543504747391_1FF9ZQyx6"
    ],
    "robotApplicationNames": [
      "AWSRoboMakerCloudWatch-1543504747391_axbYa3S3K"
    ]
  }
]
```

```

    },
    {
      "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-job/sim-
vw8lvh061nqt",
      "lastUpdatedAt": 1543441430.0,
      "status": "Completed",
      "simulationApplicationNames": [
        "AWSRoboMakerHelloWorld-1543437372341__yb_Jg961"
      ],
      "robotApplicationNames": [
        "AWSRoboMakerHelloWorld-1543437372341_lNbmKHvs9"
      ]
    },
    {
      "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-job/sim-
txy5ypxmh84",
      "lastUpdatedAt": 1543437488.0,
      "status": "Completed",
      "simulationApplicationNames": [
        "AWSRoboMakerHelloWorld-1543437372341__yb_Jg961"
      ],
      "robotApplicationNames": [
        "AWSRoboMakerHelloWorld-1543437372341_lNbmKHvs9"
      ]
    }
  ]
}

```

- For API details, see [ListSimulationJobs](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags for a resource

This example lists tags for an AWS RoboMaker resource.

Command:

```
aws robomaker list-tags-for-resource --resource-arn "arn:aws:robomaker:us-west-2:111111111111:robot/Robby_the_Robot/1544035373264"
```

Output:

```
{
  "tags": {
    "Region": "North",
    "Stage": "Initial"
  }
}
```

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

register-robot

The following code example shows how to use `register-robot`.

AWS CLI

To register a robot

This example registers a robot to a fleet.

Command:

```
aws robomaker register-robot --fleet arn:aws:robomaker:us-west-2:111111111111:deployment-fleet/MyFleet/1550771358907 --robot arn:aws:robomaker:us-west-2:111111111111:robot/MyRobot/1550772324398
```

Output:

```
{
  "fleet": "arn:aws:robomaker:us-west-2:111111111111:deployment-fleet/MyFleet/1550771358907",
  "robot": "arn:aws:robomaker:us-west-2:111111111111:robot/MyRobot/1550772324398"
}
```

- For API details, see [RegisterRobot](#) in *AWS CLI Command Reference*.

restart-simulation-job

The following code example shows how to use `restart-simulation-job`.

AWS CLI

To restart a simulation

This example restarts a simulation.

Command:

```
aws robomaker restart-simulation-job --job arn:aws:robomaker:us-west-2:111111111111:simulation-job/sim-t6rdgt70mftr
```

- For API details, see [RestartSimulationJob](#) in *AWS CLI Command Reference*.

sync-deployment-job

The following code example shows how to use `sync-deployment-job`.

AWS CLI

To sync a deployment job

This example synchronizes a deployment job.

Command:

```
aws robomaker sync-deployment-job --fleet arn:aws:robomaker:us-west-2:111111111111:deployment-fleet/Trek/1539894765711
```

Output:

```
{
  "arn": "arn:aws:robomaker:us-west-2:111111111111:deployment-job/
deployment-09ccxs3tlfms",
  "fleet": "arn:aws:robomaker:us-west-2:111111111111:deployment-fleet/
MyFleet/1539894765711",
  "status": "Pending",
  "deploymentConfig": {
    "concurrentDeploymentPercentage": 20,
```

```
    "failureThresholdPercentage": 25
  },
  "deploymentApplicationConfigs": [
    {
      "application": "arn:aws:robomaker:us-west-2:111111111111:robot-
application/MyRobotApplication/1546541208251",
      "applicationVersion": "1",
      "launchConfig": {
        "packageName": "hello_world_simulation",
        "launchFile": "empty_world.launch"
      }
    }
  ],
  "createdAt": 1551286954.0
}
```

- For API details, see [SyncDeploymentJob](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To tag a resource

This example tags a resource. It attaches two tags: Region and Stage.

Command:

```
aws robomaker tag-resource --resource-arn "arn:aws:robomaker:us-
west-2:111111111111:robot/MyRobot/1544035373264" --tags Region=North,Stage=Initial
```

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To untag a resource

This example removes a tag from a resource. It removes the Region tag.

Command:

```
aws robomaker untag-resource --resource-arn "arn:aws:robomaker:us-west-2:111111111111:robot/MyRobot/1544035373264" --tag-keys Region
```

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-robot-application

The following code example shows how to use update-robot-application.

AWS CLI

To update a robot application

This example updates a robot application.

Command:

```
aws robomaker update-robot-application --application arn:aws:robomaker:us-west-2:111111111111:robot-application/MyRobotApplication/1551203485821 --sources s3Bucket=my-bucket,s3Key=my-robot-application.tar.gz,architecture=X86_64 --robot-software-suite name=ROS,version=Kinetic
```

Output:

```
{
  "arn": "arn:aws:robomaker:us-west-2:111111111111:robot-application/MyRobotApplication/1551203485821",
  "name": "MyRobotApplication",
  "version": "$LATEST",
  "sources": [
    {
      "s3Bucket": "my-bucket",
      "s3Key": "my-robot-application.tar.gz",
      "architecture": "X86_64"
    }
  ],
  "robotSoftwareSuite": {
```

```
    "name": "ROS",
    "version": "Kinetic"
  },
  "lastUpdatedAt": 1551287993.0,
  "revisionId": "20b5e331-24fd-4504-8b8c-531afe5f4c94"
}
```

- For API details, see [UpdateRobotApplication](#) in *AWS CLI Command Reference*.

update-simulation-application

The following code example shows how to use `update-simulation-application`.

AWS CLI

To update a simulation application

This example updates a simulation application.

Command:

```
aws robomaker update-simulation-application --application
arn:aws:robomaker:us-west-2:111111111111:simulation-application/
MySimulationApplication/1551203427605 --sources s3Bucket=my-bucket,s3Key=my-
simulation-application.tar.gz,architecture=X86_64 --robot-software-suite
name=ROS,version=Kinetic --simulation-software-suite name=Gazebo,version=7 --
rendering-engine name=OGRE,version=1.x
```

Output:

```
{
  "arn": "arn:aws:robomaker:us-west-2:111111111111:simulation-application/
MySimulationApplication/1551203427605",
  "name": "MySimulationApplication",
  "version": "$LATEST",
  "sources": [
    {
      "s3Bucket": "my-bucket",
      "s3Key": "my-simulation-application.tar.gz",
      "architecture": "X86_64"
    }
  ],
}
```

```
"simulationSoftwareSuite": {
  "name": "Gazebo",
  "version": "7"
},
"robotSoftwareSuite": {
  "name": "ROS",
  "version": "Kinetic"
},
"renderingEngine": {
  "name": "OGRE",
  "version": "1.x"
},
"lastUpdatedAt": 1551289361.0,
"revisionId": "4a22cb5d-93c5-4cef-9311-52bdd119b79e"
}
```

- For API details, see [UpdateSimulationApplication](#) in *AWS CLI Command Reference*.

Route 53 examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Route 53.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

change-resource-record-sets

The following code example shows how to use `change-resource-sets`.

AWS CLI

To create, update, or delete a resource record set

The following `change-resource-record-sets` command creates a resource record set using the `hosted-zone-id` `Z1R8UBAEXAMPLE` and the JSON-formatted configuration in the file `C:\awscli\route53\change-resource-record-sets.json`:

```
aws route53 change-resource-record-sets --hosted-zone-id Z1R8UBAEXAMPLE --change-batch file://C:\awscli\route53\change-resource-record-sets.json
```

For more information, see `POST ChangeResourceRecordSets` in the *Amazon Route 53 API Reference*.

The configuration in the JSON file depends on the kind of resource record set you want to create:

BasicWeightedAliasWeighted AliasLatencyLatency AliasFailoverFailover Alias

Basic Syntax:

```
{
  "Comment": "optional comment about the changes in this change batch request",
  "Changes": [
    {
      "Action": "CREATE"|"DELETE"|"UPSERT",
      "ResourceRecordSet": {
        "Name": "DNS domain name",
        "Type": "SOA"|"A"|"TXT"|"NS"|"CNAME"|"MX"|"PTR"|"SRV"|"SPF"|"AAAA",
        "TTL": time to live in seconds,
        "ResourceRecords": [
          {
            "Value": "applicable value for the record type"
          },
          {...}
        ]
      }
    },
    {...}
  ]
}
```

Weighted Syntax:

```
{
  "Comment": "optional comment about the changes in this change batch request",
  "Changes": [
    {
      "Action": "CREATE"|"DELETE"|"UPSERT",
      "ResourceRecordSet": {
        "Name": "DNS domain name",
        "Type": "SOA"|"A"|"TXT"|"NS"|"CNAME"|"MX"|"PTR"|"SRV"|"SPF"|"AAAA",
        "SetIdentifier": "unique description for this resource record set",
        "Weight": value between 0 and 255,
        "TTL": time to live in seconds,
        "ResourceRecords": [
          {
            "Value": "applicable value for the record type"
          },
          {...}
        ],
        "HealthCheckId": "optional ID of an Amazon Route 53 health check"
      }
    },
    {...}
  ]
}
```

Alias Syntax:

```
{
  "Comment": "optional comment about the changes in this change batch request",
  "Changes": [
    {
      "Action": "CREATE"|"DELETE"|"UPSERT",
      "ResourceRecordSet": {
        "Name": "DNS domain name",
        "Type": "SOA"|"A"|"TXT"|"NS"|"CNAME"|"MX"|"PTR"|"SRV"|"SPF"|"AAAA",
        "AliasTarget": {
          "HostedZoneId": "hosted zone ID for your CloudFront distribution, Amazon
          S3 bucket, Elastic Load Balancing load balancer, or Amazon Route 53 hosted zone",
          "DNSName": "DNS domain name for your CloudFront distribution, Amazon S3
          bucket, Elastic Load Balancing load balancer, or another resource record set in
          this hosted zone",
          "EvaluateTargetHealth": true|false
        }
      }
    }
  ]
}
```

```

    },
    "HealthCheckId": "optional ID of an Amazon Route 53 health check"
  }
},
{...}
]
}

```

Weighted Alias Syntax:

```

{
  "Comment": "optional comment about the changes in this change batch request",
  "Changes": [
    {
      "Action": "CREATE"|"DELETE"|"UPSERT",
      "ResourceRecordSet": {
        "Name": "DNS domain name",
        "Type": "SOA"|"A"|"TXT"|"NS"|"CNAME"|"MX"|"PTR"|"SRV"|"SPF"|"AAAA",
        "SetIdentifier": "unique description for this resource record set",
        "Weight": value between 0 and 255,
        "AliasTarget": {
          "HostedZoneId": "hosted zone ID for your CloudFront distribution, Amazon
          S3 bucket, Elastic Load Balancing load balancer, or Amazon Route 53 hosted zone",
          "DNSName": "DNS domain name for your CloudFront distribution, Amazon S3
          bucket, Elastic Load Balancing load balancer, or another resource record set in
          this hosted zone",
          "EvaluateTargetHealth": true|false
        },
        "HealthCheckId": "optional ID of an Amazon Route 53 health check"
      }
    },
    {...}
  ]
}

```

Latency Syntax:

```

{
  "Comment": "optional comment about the changes in this change batch request",
  "Changes": [
    {
      "Action": "CREATE"|"DELETE"|"UPSERT",
      "ResourceRecordSet": {

```



```

    "Name": "DNS domain name",
    "Type": "SOA"|"A"|"TXT"|"NS"|"CNAME"|"MX"|"PTR"|"SRV"|"SPF"|"AAAA",
    "SetIdentifier": "unique description for this resource record set",
    "Region": "Amazon EC2 region name",
    "TTL": time to live in seconds,
    "ResourceRecords": [
      {
        "Value": "applicable value for the record type"
      },
      {...}
    ],
    "HealthCheckId": "optional ID of an Amazon Route 53 health check"
  }
},
{...}
]
}

```

Latency Alias Syntax:

```

{
  "Comment": "optional comment about the changes in this change batch request",
  "Changes": [
    {
      "Action": "CREATE"|"DELETE"|"UPSERT",
      "ResourceRecordSet": {
        "Name": "DNS domain name",
        "Type": "SOA"|"A"|"TXT"|"NS"|"CNAME"|"MX"|"PTR"|"SRV"|"SPF"|"AAAA",
        "SetIdentifier": "unique description for this resource record set",
        "Region": "Amazon EC2 region name",
        "AliasTarget": {
          "HostedZoneId": "hosted zone ID for your CloudFront distribution, Amazon
          S3 bucket, Elastic Load Balancing load balancer, or Amazon Route 53 hosted zone",
          "DNSName": "DNS domain name for your CloudFront distribution, Amazon S3
          bucket, Elastic Load Balancing load balancer, or another resource record set in
          this hosted zone",
          "EvaluateTargetHealth": true|false
        },
        "HealthCheckId": "optional ID of an Amazon Route 53 health check"
      }
    },
    {...}
  ]
}

```

```
}

```

Failover Syntax:

```
{
  "Comment": "optional comment about the changes in this change batch request",
  "Changes": [
    {
      "Action": "CREATE"|"DELETE"|"UPSERT",
      "ResourceRecordSet": {
        "Name": "DNS domain name",
        "Type": "SOA"|"A"|"TXT"|"NS"|"CNAME"|"MX"|"PTR"|"SRV"|"SPF"|"AAAA",
        "SetIdentifier": "unique description for this resource record set",
        "Failover": "PRIMARY" | "SECONDARY",
        "TTL": time to live in seconds,
        "ResourceRecords": [
          {
            "Value": "applicable value for the record type"
          },
          {...}
        ],
        "HealthCheckId": "ID of an Amazon Route 53 health check"
      }
    },
    {...}
  ]
}
```

Failover Alias Syntax:

```
{
  "Comment": "optional comment about the changes in this change batch request",
  "Changes": [
    {
      "Action": "CREATE"|"DELETE"|"UPSERT",
      "ResourceRecordSet": {
        "Name": "DNS domain name",
        "Type": "SOA"|"A"|"TXT"|"NS"|"CNAME"|"MX"|"PTR"|"SRV"|"SPF"|"AAAA",
        "SetIdentifier": "unique description for this resource record set",
        "Failover": "PRIMARY" | "SECONDARY",
        "AliasTarget": {
          "HostedZoneId": "hosted zone ID for your CloudFront distribution, Amazon
S3 bucket, Elastic Load Balancing load balancer, or Amazon Route 53 hosted zone",

```

```

        "DNSName": "DNS domain name for your CloudFront distribution, Amazon S3
        bucket, Elastic Load Balancing load balancer, or another resource record set in
        this hosted zone",
        "EvaluateTargetHealth": true|false
    },
    "HealthCheckId": "optional ID of an Amazon Route 53 health check"
}
},
{...}
]
}

```

- For API details, see [ChangeResourceRecordSets](#) in *AWS CLI Command Reference*.

change-tags-for-resource

The following code example shows how to use `change-tags-for-resource`.

AWS CLI

The following command adds a tag named `owner` to a healthcheck resource specified by ID:

```
aws route53 change-tags-for-resource --resource-type healthcheck --resource-id
6233434j-18c1-34433-ba8e-3443434 --add-tags Key=owner,Value=myboss
```

The following command removes a tag named `owner` from a hosted zone resource specified by ID:

```
aws route53 change-tags-for-resource --resource-type hostedzone --resource-id
Z1523434445 --remove-tag-keys owner
```

- For API details, see [ChangeTagsForResource](#) in *AWS CLI Command Reference*.

create-health-check

The following code example shows how to use `create-health-check`.

AWS CLI

To create a health check

The following `create-health-check` command creates a health check using the caller reference `2014-04-01-18:47` and the JSON-formatted configuration in the file `C:\awscli\route53\create-health-check.json`:

```
aws route53 create-health-check --caller-reference 2014-04-01-18:47 --health-check-config file://C:\awscli\route53\create-health-check.json
```

JSON syntax:

```
{
  "IPAddress": "IP address of the endpoint to check",
  "Port": port on the endpoint to check--required when Type is "TCP",
  "Type": "HTTP"|"HTTPS"|"HTTP_STR_MATCH"|"HTTPS_STR_MATCH"|"TCP",
  "ResourcePath": "path of the file that you want Amazon Route 53 to request--all Types except TCP",
  "FullyQualifiedDomainName": "domain name of the endpoint to check--all Types except TCP",
  "SearchString": "if Type is HTTP_STR_MATCH or HTTPS_STR_MATCH, the string to search for in the response body from the specified resource",
  "RequestInterval": 10 | 30,
  "FailureThreshold": integer between 1 and 10
}
```

To add the health check to a Route 53 resource record set, use the `change-resource-record-sets` command.

For more information, see Amazon Route 53 Health Checks and DNS Failover in the *Amazon Route 53 Developer Guide*.

- For API details, see [CreateHealthCheck](#) in *AWS CLI Command Reference*.

create-hosted-zone

The following code example shows how to use `create-hosted-zone`.

AWS CLI

To create a hosted zone

The following `create-hosted-zone` command adds a hosted zone named `example.com` using the caller reference `2014-04-01-18:47`. The optional comment includes a space, so it must be enclosed in quotation marks:

```
aws route53 create-hosted-zone --name example.com --caller-reference
2014-04-01-18:47 --hosted-zone-config Comment="command-line version"
```

For more information, see *Working with Hosted Zones* in the *Amazon Route 53 Developer Guide*.

- For API details, see [CreateHostedZone](#) in *AWS CLI Command Reference*.

delete-health-check

The following code example shows how to use `delete-health-check`.

AWS CLI

To delete a health check

The following `delete-health-check` command deletes the health check with a `health-check-id` of `e75b48d9-547a-4c3d-88a5-ae4002397608`:

```
aws route53 delete-health-check --health-check-id e75b48d9-547a-4c3d-88a5-
ae4002397608
```

- For API details, see [DeleteHealthCheck](#) in *AWS CLI Command Reference*.

delete-hosted-zone

The following code example shows how to use `delete-hosted-zone`.

AWS CLI

To delete a hosted zone

The following `delete-hosted-zone` command deletes the hosted zone with an `id` of `Z36KTIQEXAMPLE`:

```
aws route53 delete-hosted-zone --id Z36KTIQEXAMPLE
```

- For API details, see [DeleteHostedZone](#) in *AWS CLI Command Reference*.

get-change

The following code example shows how to use `get-change`.

AWS CLI

To get the status of a change to resource record sets

The following `get-change` command gets the status and other information about the `change-resource-record-sets` request that has an `Id` of `/change/CWPIK4URU2I5S`:

```
aws route53 get-change --id /change/CWPIK4URU2I5S
```

- For API details, see [GetChange](#) in *AWS CLI Command Reference*.

get-health-check

The following code example shows how to use `get-health-check`.

AWS CLI

To get information about a health check

The following `get-health-check` command gets information about the health check that has a `health-check-id` of `02ec8401-9879-4259-91fa-04e66d094674`:

```
aws route53 get-health-check --health-check-id 02ec8401-9879-4259-91fa-04e66d094674
```

- For API details, see [GetHealthCheck](#) in *AWS CLI Command Reference*.

get-hosted-zone

The following code example shows how to use `get-hosted-zone`.

AWS CLI

To get information about a hosted zone

The following `get-hosted-zone` command gets information about the hosted zone with an `id` of `Z1R8UBAEXAMPLE`:

```
aws route53 get-hosted-zone --id Z1R8UBAEXAMPLE
```

- For API details, see [GetHostedZone](#) in *AWS CLI Command Reference*.

list-health-checks

The following code example shows how to use `list-health-checks`.

AWS CLI

To list the health checks associated with the current AWS account

The following `list-health-checks` command lists detailed information about the first 100 health checks that are associated with the current AWS account.:

```
aws route53 list-health-checks
```

If you have more than 100 health checks, or if you want to list them in groups smaller than 100, include the `--max-items` parameter. For example, to list health checks one at a time, use the following command:

```
aws route53 list-health-checks --max-items 1
```

To view the next health check, take the value of `NextToken` from the response to the previous command, and include it in the `--starting-token` parameter, for example:

```
aws route53 list-health-checks --max-items 1 --starting-token Z3M3LMPEXAMPLE
```

- For API details, see [ListHealthChecks](#) in *AWS CLI Command Reference*.

list-hosted-zones-by-name

The following code example shows how to use `list-hosted-zones-by-name`.

AWS CLI

The following command lists up to 100 hosted zones ordered by domain name:

```
aws route53 list-hosted-zones-by-name
```

Output:

```
{
  "HostedZones": [
    {
      "ResourceRecordSetCount": 2,
      "CallerReference": "test20150527-2",
      "Config": {
        "Comment": "test2",
        "PrivateZone": false
      },
      "Id": "/hostedzone/Z119WBBTVP5WFX",
      "Name": "2.example.com."
    },
    {
      "ResourceRecordSetCount": 2,
      "CallerReference": "test20150527-1",
      "Config": {
        "Comment": "test",
        "PrivateZone": false
      },
      "Id": "/hostedzone/Z3P5QSUBK4POTI",
      "Name": "www.example.com."
    }
  ],
  "IsTruncated": false,
  "MaxItems": "100"
}
```

The following command lists hosted zones ordered by name, beginning with `www.example.com`:

```
aws route53 list-hosted-zones-by-name --dns-name www.example.com
```

Output:

```
{
  "HostedZones": [
    {
      "ResourceRecordSetCount": 2,
      "CallerReference": "mwunderl20150527-1",
      "Config": {
```



```
        "Comment": "test",
        "PrivateZone": false
    },
    "Id": "/hostedzone/Z3P5QSUBK4P0TI",
    "Name": "www.example.com."
}
],
"DNSName": "www.example.com",
"IsTruncated": false,
"MaxItems": "100"
}
```

- For API details, see [ListHostedZonesByName](#) in *AWS CLI Command Reference*.

list-hosted-zones

The following code example shows how to use `list-hosted-zones`.

AWS CLI

To list the hosted zones associated with the current AWS account

The following `list-hosted-zones` command lists summary information about the first 100 hosted zones that are associated with the current AWS account.:

```
aws route53 list-hosted-zones
```

If you have more than 100 hosted zones, or if you want to list them in groups smaller than 100, include the `--max-items` parameter. For example, to list hosted zones one at a time, use the following command:

```
aws route53 list-hosted-zones --max-items 1
```

To view information about the next hosted zone, take the value of `NextToken` from the response to the previous command, and include it in the `--starting-token` parameter, for example:

```
aws route53 list-hosted-zones --max-items 1 --starting-token Z3M3LMPEXAMPLE
```

- For API details, see [ListHostedZones](#) in *AWS CLI Command Reference*.

list-query-logging-configs

The following code example shows how to use `list-query-logging-configs`.

AWS CLI

To list query logging configurations

The following `list-query-logging-configs` example lists information about the first 100 query logging configurations in your AWS account, for the hosted zone Z10X3WQEXAMPLE.

```
aws route53 list-query-logging-configs \
  --hosted-zone-id Z10X3WQEXAMPLE
```

Output:

```
{
  "QueryLoggingConfigs": [
    {
      "Id": "964ff34e-ae03-4f06-80a2-9683cexample",
      "HostedZoneId": "Z10X3WQEXAMPLE",
      "CloudWatchLogsLogGroupArn": "arn:aws:logs:us-east-1:111122223333:log-
group:/aws/route53/example.com:*"
    }
  ]
}
```

For more information, see [Logging DNS queries](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [ListQueryLoggingConfigs](#) in *AWS CLI Command Reference*.

list-resource-record-sets

The following code example shows how to use `list-resource-record-sets`.

AWS CLI

To list the resource record sets in a hosted zone

The following `list-resource-record-sets` command lists summary information about the first 100 resource record sets in a specified hosted zone.:

```
aws route53 list-resource-record-sets --hosted-zone-id Z2LD58HEXAMPLE
```

If the hosted zone contains more than 100 resource record sets, or if you want to list them in groups smaller than 100, include the `--max-items` parameter. For example, to list resource record sets one at a time, use the following command:

```
aws route53 list-resource-record-sets --hosted-zone-id Z2LD58HEXAMPLE --max-items 1
```

To view information about the next resource record set in the hosted zone, take the value of `NextToken` from the response to the previous command, and include it in the `--starting-token` parameter, for example:

```
aws route53 list-resource-record-sets --hosted-zone-id Z2LD58HEXAMPLE --max-items 1  
--starting-token Z3M3LMPEXAMPLE
```

To view all the resource record sets of a particular name, use the `--query` parameter to filter them out. For example:

```
aws route53 list-resource-record-sets --hosted-zone-id Z2LD58HEXAMPLE --query  
"ResourceRecordSets[?Name == 'example.domain.']"
```

- For API details, see [ListResourceRecordSets](#) in *AWS CLI Command Reference*.

Route 53 domain registration examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Route 53 domain registration.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

check-domain-availability

The following code example shows how to use check-domain-availability.

AWS CLI

To determine whether you can register a domain name with Route 53

The following check-domain-availability command returns information about whether the domain name example.com is available to be registered using Route 53.

This command runs only in the us-east-1 Region. If your default region is set to us-east-1, you can omit the region parameter.

```
aws route53domains check-domain-availability \  
  --region us-east-1 \  
  --domain-name example.com
```

Output:

```
{  
  "Availability": "UNAVAILABLE"  
}
```

Route 53 supports a large number of top-level domains (TLDs), such as .com and .jp, but we don't support all available TLDs. If you check the availability of a domain and Route 53 doesn't support the TLD, check-domain-availability returns the following message.

```
An error occurred (UnsupportedTLD) when calling the CheckDomainAvailability  
operation: <top-level domain> tld is not supported.
```

For a list of the TLDs that you can use when registering a domain with Route 53, see [Domains That You Can Register with Amazon Route 53](#) in the *Amazon Route 53 Developer Guide*. For more information about registering domains with Amazon Route 53, see [Registering a New Domain](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [CheckDomainAvailability](#) in *AWS CLI Command Reference*.

check-domain-transferability

The following code example shows how to use `check-domain-transferability`.

AWS CLI

To determine whether a domain can be transferred to Route 53

The following `check-domain-transferability` command returns information about whether you can transfer the domain name `example.com` to Route 53.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains check-domain-transferability \  
  --region us-east-1 \  
  --domain-name example.com
```

Output:

```
{  
  "Transferability": {  
    "Transferable": "UNTRANSFERABLE"  
  }  
}
```

For more information, see [Transferring Registration for a Domain to Amazon Route 53](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [CheckDomainTransferability](#) in *AWS CLI Command Reference*.

delete-tags-for-domain

The following code example shows how to use `delete-tags-for-domain`.

AWS CLI

To delete tags for a domain

The following `delete-tags-for-domain` command deletes three tags from the specified domain. Note that you specify only the tag key, not the tag value.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains delete-tags-for-domain \  
  --region us-east-1 \  
  --domain-name example.com \  
  --tags-to-delete accounting-key hr-key engineering-key
```

This command produces no output.

To confirm that the tags were deleted, you can run [list-tags-for-domain](#) . For more information, see [Tagging Amazon Route 53 Resources](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [DeleteTagsForDomain](#) in *AWS CLI Command Reference*.

disable-domain-auto-renew

The following code example shows how to use `disable-domain-auto-renew`.

AWS CLI

To disable automatic renewal of a domain

The following `disable-domain-auto-renew` command configures Route 53 to *not* automatically renew the domain `example.com` before registration for the domain expires.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains disable-domain-auto-renew \  
  --region us-east-1 \  
  --domain-name example.com
```

This command produces no output.

To confirm that the setting was changed, you can run [get-domain-detail](#) . If automatic renewal is disabled, the value of `AutoRenew` is `False`. For more information about automatic renewal, see [Renewing Registration for a Domain <https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/domain-renew.html>](https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/domain-renew.html) in the *Amazon Route 53 Developer Guide*.

- For API details, see [DisableDomainAutoRenew](#) in *AWS CLI Command Reference*.

disable-domain-transfer-lock

The following code example shows how to use `disable-domain-transfer-lock`.

AWS CLI

To disable the transfer lock on a domain

The following `disable-domain-transfer-lock` command removes the transfer lock on the domain `example.com` so that the domain can be transferred to another registrar. This command changes the `clientTransferProhibited` status.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains disable-domain-transfer-lock \  
  --region us-east-1 \  
  --domain-name example.com
```

Output:

```
{  
  "OperationId": "3f28e0ac-126a-4113-9048-cc930example"  
}
```

To confirm that the transfer lock has been changed, you can run [get-domain-detail](#). When the transfer lock is disabled, the value of `StatusList` does *not* include `clientTransferProhibited`.

For more information about the transfer process, see [Transferring a Domain from Amazon Route 53 to Another Registrar](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [DisableDomainTransferLock](#) in *AWS CLI Command Reference*.

enable-domain-auto-renew

The following code example shows how to use `enable-domain-auto-renew`.

AWS CLI

To enable automatic renewal of a domain

The following `enable-domain-auto-renew` command configures Route 53 to automatically renew the domain `example.com` before registration for the domain expires.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains enable-domain-auto-renew \  
  --region us-east-1 \  
  --domain-name example.com
```

This command produces no output. To confirm that the setting was changed, you can run [get-domain-detail](#). If automatic renewal is enabled, the value of `AutoRenew` is `True`.

For more information about automatic renewal, see [Renewing Registration for a Domain](https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/domain-renew.html) <<https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/domain-renew.html>> in the *Amazon Route 53 Developer Guide*.

- For API details, see [EnableDomainAutoRenew](#) in *AWS CLI Command Reference*.

enable-domain-transfer-lock

The following code example shows how to use `enable-domain-transfer-lock`.

AWS CLI

To enable the transfer lock on a domain

The following `enable-domain-transfer-lock` command locks the specified domain so that it can't be transferred to another registrar. This command changes the `clientTransferProhibited` status.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains enable-domain-transfer-lock \  
  --region us-east-1 \  
  --domain-name example.com
```

Output:

```
{
```



```
"OperationId": "3f28e0ac-126a-4113-9048-cc930example"  
}
```

To confirm that the transfer lock has been changed, you can run [get-domain-detail](#). When the transfer lock is enabled, the value of `StatusList` includes `clientTransferProhibited`.

For more information about the transfer process, see [Transferring a Domain from Amazon Route 53 to Another Registrar](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [EnableDomainTransferLock](#) in *AWS CLI Command Reference*.

get-contact-reachability-status

The following code example shows how to use `get-contact-reachability-status`.

AWS CLI

To determine whether the registrant contact has responded to a confirmation email

The following `get-contact-reachability-status` command returns information about whether the registrant contact for the specified domain has responded to a confirmation email.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains get-contact-reachability-status \  
  --region us-east-1 \  
  --domain-name example.com
```

Output:

```
{  
  "domainName": "example.com",  
  "status": "DONE"  
}
```

For more information, see [Resending Authorization and Confirmation Emails](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [GetContactReachabilityStatus](#) in *AWS CLI Command Reference*.

get-domain-detail

The following code example shows how to use `get-domain-detail`.

AWS CLI

To get detailed information about a specified domain

The following `get-domain-detail` command displays detailed information about the specified domain.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains get-domain-detail \  
  --region us-east-1 \  
  --domain-name example.com
```

Output:

```
{  
  "DomainName": "example.com",  
  "Nameservers": [  
    {  
      "Name": "ns-2048.awsdns-64.com",  
      "GlueIps": []  
    },  
    {  
      "Name": "ns-2049.awsdns-65.net",  
      "GlueIps": []  
    },  
    {  
      "Name": "ns-2050.awsdns-66.org",  
      "GlueIps": []  
    },  
    {  
      "Name": "ns-2051.awsdns-67.co.uk",  
      "GlueIps": []  
    }  
  ],  
  "AutoRenew": true,  
  "AdminContact": {  
    "FirstName": "Saanvi",
```

```
    "LastName": "Sarkar",
    "ContactType": "COMPANY",
    "OrganizationName": "Example",
    "AddressLine1": "123 Main Street",
    "City": "Anytown",
    "State": "WA",
    "CountryCode": "US",
    "ZipCode": "98101",
    "PhoneNumber": "+1.8005551212",
    "Email": "ssarkar@example.com",
    "ExtraParams": []
  },
  "RegistrantContact": {
    "FirstName": "Alejandro",
    "LastName": "Rosalez",
    "ContactType": "COMPANY",
    "OrganizationName": "Example",
    "AddressLine1": "123 Main Street",
    "City": "Anytown",
    "State": "WA",
    "CountryCode": "US",
    "ZipCode": "98101",
    "PhoneNumber": "+1.8005551212",
    "Email": "arosalez@example.com",
    "ExtraParams": []
  },
  "TechContact": {
    "FirstName": "Wang",
    "LastName": "Xiulan",
    "ContactType": "COMPANY",
    "OrganizationName": "Example",
    "AddressLine1": "123 Main Street",
    "City": "Anytown",
    "State": "WA",
    "CountryCode": "US",
    "ZipCode": "98101",
    "PhoneNumber": "+1.8005551212",
    "Email": "wxiulan@example.com",
    "ExtraParams": []
  },
  "AdminPrivacy": true,
  "RegistrantPrivacy": true,
  "TechPrivacy": true,
  "RegistrarName": "Amazon Registrar, Inc.",
```

```
"WhoIsServer": "whois.registrar.amazon.com",
"RegistrarUrl": "http://registrar.amazon.com",
"AbuseContactEmail": "abuse@registrar.amazon.com",
"AbuseContactPhone": "+1.2062661000",
"CreationDate": 1444934889.601,
"ExpirationDate": 1602787689.0,
"StatusList": [
  "clientTransferProhibited"
]
}
```

- For API details, see [GetDomainDetail](#) in *AWS CLI Command Reference*.

get-domain-suggestions

The following code example shows how to use `get-domain-suggestions`.

AWS CLI

To get a list of suggested domain names

The following `get-domain-suggestions` command displays a list of suggested domain names based on the domain name `example.com`. The response includes only domain names that are available. This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains get-domain-suggestions \
  --region us-east-1 \
  --domain-name example.com \
  --suggestion-count 10 \
  --only-available
```

Output:

```
{
  "SuggestionsList": [
    {
      "DomainName": "egzaampal.com",
      "Availability": "AVAILABLE"
    },
    {
```

```
    "DomainName": "examplelaw.com",
    "Availability": "AVAILABLE"
  },
  {
    "DomainName": "examplehouse.net",
    "Availability": "AVAILABLE"
  },
  {
    "DomainName": "homeexample.net",
    "Availability": "AVAILABLE"
  },
  {
    "DomainName": "examplelist.com",
    "Availability": "AVAILABLE"
  },
  {
    "DomainName": "examplenews.net",
    "Availability": "AVAILABLE"
  },
  {
    "DomainName": "officeexample.com",
    "Availability": "AVAILABLE"
  },
  {
    "DomainName": "exampleworld.com",
    "Availability": "AVAILABLE"
  },
  {
    "DomainName": "exampleart.com",
    "Availability": "AVAILABLE"
  }
]
}
```

- For API details, see [GetDomainSuggestions](#) in *AWS CLI Command Reference*.

get-operation-detail

The following code example shows how to use get-operation-detail.

AWS CLI

To get the current status of an operation

Some domain registration operations operate asynchronously and return a response before they finish. These operations return an operation ID that you can use to get the current status. The following `get-operation-detail` command returns the status of the specified operation.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains get-operation-detail \  
  --region us-east-1 \  
  --operation-id edbd8d63-7fe7-4343-9bc5-54033example
```

Output:

```
{  
  "OperationId": "edbd8d63-7fe7-4343-9bc5-54033example",  
  "Status": "SUCCESSFUL",  
  "DomainName": "example.com",  
  "Type": "DOMAIN_LOCK",  
  "SubmittedDate": 1573749367.864  
}
```

- For API details, see [GetOperationDetail](#) in *AWS CLI Command Reference*.

list-domains

The following code example shows how to use `list-domains`.

AWS CLI

To list the domains that are registered with the current AWS account

The following `list-domains` command lists summary information about the domains that are registered with the current AWS account.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains list-domains
```

```
--region us-east-1
```

Output:

```
{
  "Domains": [
    {
      "DomainName": "example.com",
      "AutoRenew": true,
      "TransferLock": true,
      "Expiry": 1602712345.0
    },
    {
      "DomainName": "example.net",
      "AutoRenew": true,
      "TransferLock": true,
      "Expiry": 1602723456.0
    },
    {
      "DomainName": "example.org",
      "AutoRenew": true,
      "TransferLock": true,
      "Expiry": 1602734567.0
    }
  ]
}
```

- For API details, see [ListDomains](#) in *AWS CLI Command Reference*.

list-operations

The following code example shows how to use `list-operations`.

AWS CLI**To list the status of operations that return an operation ID**

Some domain registration operations run asynchronously and return a response before they finish. These operations return an operation ID that you can use to get the current status. The following `list-operations` command lists summary information, including the status, about the current domain-registration operations.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains list-operations
  --region us-east-1
```

Output:

```
{
  "Operations": [
    {
      "OperationId": "aab9822f-1da0-4bf3-8a15-fd4e0example",
      "Status": "SUCCESSFUL",
      "Type": "DOMAIN_LOCK",
      "SubmittedDate": 1455321739.986
    },
    {
      "OperationId": "c24379ed-76be-42f8-bdad-9379bexample",
      "Status": "SUCCESSFUL",
      "Type": "UPDATE_NAMESERVER",
      "SubmittedDate": 1468960475.109
    },
    {
      "OperationId": "f47e1297-ef9e-4c2b-ae1e-a5fcbexample",
      "Status": "SUCCESSFUL",
      "Type": "RENEW_DOMAIN",
      "SubmittedDate": 1473561835.943
    },
    {
      "OperationId": "75584f23-b15f-459e-aed7-dc6f5example",
      "Status": "SUCCESSFUL",
      "Type": "UPDATE_DOMAIN_CONTACT",
      "SubmittedDate": 1547501003.41
    }
  ]
}
```

The output includes all the operations that return an operation ID and that you have performed on all the domains that you have ever registered using the current AWS account. If you want to get only the operations that you submitted after a specified date, you can include the `submitted-since` parameter and specify a date in Unix format and Coordinated Universal

Time (UTC). The following command gets the status of all operations that were submitted after 12:00 am UTC on January 1, 2020.

```
aws route53domains list-operations \  
  --submitted-since 1577836800
```

- For API details, see [ListOperations](#) in *AWS CLI Command Reference*.

list-tags-for-domain

The following code example shows how to use `list-tags-for-domain`.

AWS CLI

To list tags for a domain

The following `list-tags-for-domain` command lists the tags that are currently associated with the specified domain.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains list-tags-for-domain \  
  --region us-east-1 \  
  --domain-name example.com
```

Output:

```
{  
  "TagList": [  
    {  
      "Key": "key1",  
      "Value": "value1"  
    },  
    {  
      "Key": "key2",  
      "Value": "value2"  
    }  
  ]  
}
```

For more information, see [Tagging Amazon Route 53 Resources](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [ListTagsForDomain](#) in *AWS CLI Command Reference*.

register-domain

The following code example shows how to use `register-domain`.

AWS CLI

To register a domain

The following `register-domain` command registers a domain, retrieving all parameter values from a JSON-formatted file.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains register-domain \  
  --region us-east-1 \  
  --cli-input-json file://register-domain.json
```

Contents of `register-domain.json`:

```
{  
  "DomainName": "example.com",  
  "DurationInYears": 1,  
  "AutoRenew": true,  
  "AdminContact": {  
    "FirstName": "Martha",  
    "LastName": "Rivera",  
    "ContactType": "PERSON",  
    "OrganizationName": "Example",  
    "AddressLine1": "1 Main Street",  
    "City": "Anytown",  
    "State": "WA",  
    "CountryCode": "US",  
    "ZipCode": "98101",  
    "PhoneNumber": "+1.8005551212",  
    "Email": "mrivera@example.com"  
  },  
  "RegistrantContact": {
```

```
"FirstName": "Li",
"LastName": "Juan",
"ContactType": "PERSON",
"OrganizationName": "Example",
"AddressLine1": "1 Main Street",
"City": "Anytown",
"State": "WA",
"CountryCode": "US",
"ZipCode": "98101",
"PhoneNumber": "+1.8005551212",
"Email": "ljuan@example.com"
},
"TechContact": {
  "FirstName": "Mateo",
  "LastName": "Jackson",
  "ContactType": "PERSON",
  "OrganizationName": "Example",
  "AddressLine1": "1 Main Street",
  "City": "Anytown",
  "State": "WA",
  "CountryCode": "US",
  "ZipCode": "98101",
  "PhoneNumber": "+1.8005551212",
  "Email": "mjackson@example.com"
},
"PrivacyProtectAdminContact": true,
"PrivacyProtectRegistrantContact": true,
"PrivacyProtectTechContact": true
}
```

Output:

```
{
  "OperationId": "b114c44a-9330-47d1-a6e8-a0b11example"
}
```

To confirm that the operation succeeded, you can run `get-operation-detail`. For more information, see [get-operation-detail](#).

For more information, see [Registering a New Domain](#) in the *Amazon Route 53 Developer Guide*.

For information about which top-level domains (TLDs) require values for `ExtraParams` and what the valid values are, see [ExtraParam](#) in the *Amazon Route 53 API Reference*.

- For API details, see [RegisterDomain](#) in *AWS CLI Command Reference*.

renew-domain

The following code example shows how to use `renew-domain`.

AWS CLI

To renew a domain

The following `renew-domain` command renews the specified domain for five years. To get the value for `current-expiry-year`, use the `get-domain-detail` command, and convert the value of `ExpirationDate` from Unix format.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains renew-domain \  
  --region us-east-1 \  
  --domain-name example.com \  
  --duration-in-years 5 \  
  --current-expiry-year 2020
```

Output:

```
{  
  "OperationId": "3f28e0ac-126a-4113-9048-cc930example"  
}
```

To confirm that the operation succeeded, you can run `get-operation-detail`. For more information, see [get-operation-detail](#).

The registry for each top-level domain (TLD), such as `.com` or `.org`, controls the maximum number of years that you can renew a domain for. To get the maximum renewal period for your domain, see the "Registration and Renewal Period" section for your TLD in [Domains That You Can Register with Amazon Route 53](#) in the *Amazon Route 53 Developer Guide*.

For more information, see [Renewing Registration for a Domain](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [RenewDomain](#) in *AWS CLI Command Reference*.

resent-contact-reachability-email

The following code example shows how to use `resent-contact-reachability-email`.

AWS CLI

To resend the confirmation email to the current email address for the registrant contact

The following `resent-contact-reachability-email` command resends the confirmation email to the current email address for the registrant contact for the `example.com` domain.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains resent-contact-reachability-email \  
  --region us-east-1 \  
  --domain-name example.com
```

Output:

```
{  
  "domainName": "example.com",  
  "emailAddress": "moliveira@example.com",  
  "isAlreadyVerified": true  
}
```

If the value of `isAlreadyVerified` is `true`, as in this example, the registrant contact has already confirmed that the specified email address is reachable.

For more information, see [Resending Authorization and Confirmation Emails](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [ResendContactReachabilityEmail](#) in *AWS CLI Command Reference*.

retrieve-domain-auth-code

The following code example shows how to use `retrieve-domain-auth-code`.

AWS CLI

To get the authorization code for a domain so you can transfer the domain to another registrar

The following `retrieve-domain-auth-code` command gets the current authorization code for the `example.com` domain. You give this value to another domain registrar when you want to transfer the domain to that registrar.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains retrieve-domain-auth-code \  
  --region us-east-1 \  
  --domain-name example.com
```

Output:

```
{  
  "AuthCode": ")o!v3dJeXampLe"  
}
```

For more information, see [Transferring a Domain from Amazon Route 53 to Another Registrar](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [RetrieveDomainAuthCode](#) in *AWS CLI Command Reference*.

transfer-domain

The following code example shows how to use `transfer-domain`.

AWS CLI

To transfer a domain to Amazon Route 53

The following `transfer-domain` command transfers a domain to Route 53, with the parameters provided by the JSON-formatted file `C:\temp\transfer-domain.json`.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains transfer-domain \  
--region us-east-1 \  
--cli-input-json file://C:\temp\transfer-domain.json
```

Contents of `transfer-domain.json`:

```
{  
  "DomainName": "example.com",  
  "DurationInYears": 1,  
  "Nameservers": [  
    {  
      "Name": "ns-2048.awsdns-64.com"  
    },  
    {  
      "Name": "ns-2049.awsdns-65.net"  
    },  
    {  
      "Name": "ns-2050.awsdns-66.org"  
    },  
    {  
      "Name": "ns-2051.awsdns-67.co.uk"  
    }  
  ],  
  "AuthCode": ")o!v3dJeXampLe",  
  "AutoRenew": true,  
  "AdminContact": {  
    "FirstName": "Martha",  
    "LastName": "Rivera",  
    "ContactType": "PERSON",  
    "OrganizationName": "Example",  
    "AddressLine1": "1 Main Street",  
    "City": "Anytown",  
    "State": "WA",  
    "CountryCode": "US",  
    "ZipCode": "98101",  
    "PhoneNumber": "+1.8005551212",  
    "Email": "mrivera@example.com"  
  },  
  "RegistrantContact": {  
    "FirstName": "Li",  
    "LastName": "Juan",  
    "ContactType": "PERSON",  
    "OrganizationName": "Example",
```

```
    "AddressLine1": "1 Main Street",
    "City": "Anytown",
    "State": "WA",
    "CountryCode": "US",
    "ZipCode": "98101",
    "PhoneNumber": "+1.8005551212",
    "Email": "ljuan@example.com"
  },
  "TechContact": {
    "FirstName": "Mateo",
    "LastName": "Jackson",
    "ContactType": "PERSON",
    "OrganizationName": "Example",
    "AddressLine1": "1 Main Street",
    "City": "Anytown",
    "State": "WA",
    "CountryCode": "US",
    "ZipCode": "98101",
    "PhoneNumber": "+1.8005551212",
    "Email": "mjackson@example.com"
  },
  "PrivacyProtectAdminContact": true,
  "PrivacyProtectRegistrantContact": true,
  "PrivacyProtectTechContact": true
}
```

Output:

```
{
  "OperationId": "b114c44a-9330-47d1-a6e8-a0b11example"
}
```

To confirm that the operation succeeded, you can run `get-operation-detail`. For more information, see [get-operation-detail](#).

For more information, see [Transferring Registration for a Domain to Amazon Route 53](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [TransferDomain](#) in *AWS CLI Command Reference*.

update-domain-contact-privacy

The following code example shows how to use `update-domain-contact-privacy`.

AWS CLI

To update the privacy settings for the contacts for a domain

The following `update-domain-contact-privacy` command turns off privacy protection for the administrative contact for the `example.com` domain. This command runs only in the `us-east-1` Region.

If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains update-domain-contact-privacy \  
  --region us-east-1 \  
  --domain-name example.com \  
  --no-admin-privacy
```

Output:

```
{  
  "OperationId": "b3a219e9-d801-4244-b533-b7256example"  
}
```

To confirm that the operation succeeded, you can run `get-operation-detail`. For more information, see [get-operation-detail](#).

For more information, see [Enabling or Disabling Privacy Protection for Contact Information for a Domain](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [UpdateDomainContactPrivacy](#) in *AWS CLI Command Reference*.

update-domain-contact

The following code example shows how to use `update-domain-contact`.

AWS CLI

To update the contact information for a domain

The following `update-domain-contact` command updates the contact information for a domain, getting the parameters from the JSON-formatted file `C:\temp\update-domain-contact.json`.

This command runs only in the us-east-1 Region. If your default region is set to us-east-1, you can omit the region parameter.

```
aws route53domains update-domain-contact \  
  --region us-east-1 \  
  --cli-input-json file://C:\temp\update-domain-contact.json
```

Contents of update-domain-contact.json:

```
{  
  "AdminContact": {  
    "AddressLine1": "101 Main Street",  
    "AddressLine2": "Suite 1a",  
    "City": "Seattle",  
    "ContactType": "COMPANY",  
    "CountryCode": "US",  
    "Email": "w.xiulan@example.com",  
    "FirstName": "Wang",  
    "LastName": "Xiulan",  
    "OrganizationName": "Example",  
    "PhoneNumber": "+1.8005551212",  
    "State": "WA",  
    "ZipCode": "98101"  
  },  
  "DomainName": "example.com",  
  "RegistrantContact": {  
    "AddressLine1": "101 Main Street",  
    "AddressLine2": "Suite 1a",  
    "City": "Seattle",  
    "ContactType": "COMPANY",  
    "CountryCode": "US",  
    "Email": "w.xiulan@example.com",  
    "FirstName": "Wang",  
    "LastName": "Xiulan",  
    "OrganizationName": "Example",  
    "PhoneNumber": "+1.8005551212",  
    "State": "WA",  
    "ZipCode": "98101"  
  },  
  "TechContact": {  
    "AddressLine1": "101 Main Street",  
    "AddressLine2": "Suite 1a",  
    "City": "Seattle",
```

```
"ContactType": "COMPANY",
"CountryCode": "US",
"Email": "w.xiulan@example.com",
"FirstName": "Wang",
"LastName": "Xiulan",
"OrganizationName": "Example",
"PhoneNumber": "+1.8005551212",
"State": "WA",
"ZipCode": "98101"
}
}
```

Output:

```
{
  "OperationId": "b3a219e9-d801-4244-b533-b7256example"
}
```

To confirm that the operation succeeded, you can run [get-domain-detail](#) . For more information, see [Updating Contact Information for a Domain](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [UpdateDomainContact](#) in *AWS CLI Command Reference*.

update-domain-nameservers

The following code example shows how to use `update-domain-nameservers`.

AWS CLI

To update the name servers for a domain

The following `update-domain-nameservers` command updates the name servers for a domain.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains update-domain-nameservers \
  --region us-east-1 \
  --domain-name example.com \
  --nameservers Name=ns-1.awsdns-01.org Name=ns-2.awsdns-02.co.uk
Name=ns-3.awsdns-03.net Name=ns-4.awsdns-04.com
```

Output:

```
{
  "OperationId": "f1691ec4-0e7a-489e-82e0-b19d3example"
}
```

To confirm that the operation succeeded, you can run [get-domain-detail](#) .

For more information, see [Adding or Changing Name Servers and Glue Records for a Domain](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [UpdateDomainNameservers](#) in *AWS CLI Command Reference*.

update-tags-for-domain

The following code example shows how to use `update-tags-for-domain`.

AWS CLI**To add or update tags for a domain**

The following `update-tags-for-domain` command adds or updates two keys and the corresponding values for the `example.com` domain. To update the value for a key, just include the key and the new value. You can add or update tags in only one domain at a time.

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains update-tags-for-domain \
  --region us-east-1 \
  --domain-name example.com \
  --tags-to-update "Key=key1,Value=value1" "Key=key2,Value=value2"
```

This command produces no output. To confirm that the tags were added or updated, you can run [list-tags-for-domain](#) .

For more information, see [Tagging Amazon Route 53 Resources](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [UpdateTagsForDomain](#) in *AWS CLI Command Reference*.

view-billing

The following code example shows how to use `view-billing`.

AWS CLI

To get billing information for domain registration charges for the current AWS account

The following `view-billing` command returns all the domain-related billing records for the current account for the period from January 1, 2018 (1514764800 in Unix time) and midnight on December 31, 2019 (1577836800 in Unix time).

This command runs only in the `us-east-1` Region. If your default region is set to `us-east-1`, you can omit the `region` parameter.

```
aws route53domains view-billing \  
  --region us-east-1 \  
  --start-time 1514764800 \  
  --end-time 1577836800
```

Output:

```
{  
  "BillingRecords": [  
    {  
      "DomainName": "example.com",  
      "Operation": "RENEW_DOMAIN",  
      "InvoiceId": "149962827",  
      "BillDate": 1536618063.181,  
      "Price": 12.0  
    },  
    {  
      "DomainName": "example.com",  
      "Operation": "RENEW_DOMAIN",  
      "InvoiceId": "290913289",  
      "BillDate": 1568162630.884,  
      "Price": 12.0  
    }  
  ]  
}
```

For more information, see [ViewBilling](#) in the *Amazon Route 53 API Reference*.

- For API details, see [ViewBilling](#) in *AWS CLI Command Reference*.

Route 53 Resolver examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Route 53 Resolver.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-firewall-rule-group

The following code example shows how to use `associate-firewall-rule-group`.

AWS CLI

To associate a firewall rule group with a VPC

The following `associate-firewall-rule-group` example associates a DNS Firewall rule group with an Amazon VPC.

```
aws route53resolver associate-firewall-rule-group \  
  --name test-association \  
  --firewall-rule-group-id rslvr-frg-47f93271fexample \  
  --vpc-id vpc-31e92222 \  
  --priority 101
```

Output:

```
{
  "FirewallRuleGroupAssociation": {
    "Id": "rslvr-frgassoc-57e8873d7example",
    "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-rule-group-association/rslvr-frgassoc-57e8873d7example",
    "FirewallRuleGroupId": "rslvr-frg-47f93271fexample",
    "VpcId": "vpc-31e92222",
    "Name": "test-association",
    "Priority": 101,
    "MutationProtection": "DISABLED",
    "Status": "UPDATING",
    "StatusMessage": "Creating Firewall Rule Group Association",
    "CreatorRequestId": "2ca1a304-32b3-4f5f-bc4c-EXAMPLE11111",
    "CreationTime": "2021-05-25T21:47:48.755768Z",
    "ModificationTime": "2021-05-25T21:47:48.755768Z"
  }
}
```

For more information, see [Managing associations between your VPC and Route 53 Resolver DNS Firewall rule groups](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [AssociateFirewallRuleGroup](#) in *AWS CLI Command Reference*.

associate-resolver-endpoint-ip-address

The following code example shows how to use `associate-resolver-endpoint-ip-address`.

AWS CLI

To associate another IP address with a Resolver endpoint

The following `associate-resolver-endpoint-ip-address` example associates another IP address with an inbound Resolver endpoint. If you specify only a subnet ID and omit the IP address from the `--ip-address` parameter, Resolver chooses an IP address for you from among the available IP addresses in the specified subnet.

```
aws route53resolver associate-resolver-endpoint-ip-address \
  --resolver-endpoint-id rslvr-in-497098ad5example \
  --ip-address="SubnetId=subnet-12d8exam,Ip=192.0.2.118"
```

Output:

```
{
  "ResolverEndpoint": {
    "Id": "rslvr-in-497098ad5example",
    "CreatorRequestId": "AWSConsole.25.0123456789",
    "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-endpoint/
rslvr-in-497098ad5example",
    "Name": "my-inbound-endpoint",
    "SecurityGroupIds": [
      "sg-05cd7b25d6example"
    ],
    "Direction": "INBOUND",
    "IpAddressCount": 3,
    "HostVPCId": "vpc-304bexam",
    "Status": "UPDATING",
    "StatusMessage": "Updating the Resolver Endpoint",
    "CreationTime": "2020-01-02T23:25:45.538Z",
    "ModificationTime": "2020-01-02T23:25:45.538Z"
  }
}
```

For more information, see [Values That You Specify When You Create or Edit Inbound Endpoints](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [AssociateResolverEndpointIpAddress](#) in *AWS CLI Command Reference*.

associate-resolver-rule

The following code example shows how to use `associate-resolver-rule`.

AWS CLI

To associate a Resolver rule with a VPC

The following `associate-resolver-rule` example associates a Resolver rule with an Amazon VPC. After you run the command, Resolver starts to forward DNS queries to your network based on the settings in the rule, such as the domain name of the queries that are forwarded.

```
aws route53resolver associate-resolver-rule \
  --name my-resolver-rule-association \
  --resolver-rule-id rslvr-rr-42b60677c0example \
  --vpc-id vpc-304bexam
```


Output:

```
{
  "ResolverRuleAssociation": {
    "Id": "rslvr-rrassoc-d61cbb2c8bexample",
    "ResolverRuleId": "rslvr-rr-42b60677c0example",
    "Name": "my-resolver-rule-association",
    "VPCId": "vpc-304bexam",
    "Status": "CREATING",
    "StatusMessage": "[Trace id: 1-5dc5a8fa-ec2cc480d2ef07617example] Creating
the association."
  }
}
```

For more information, see [Forwarding Outbound DNS Queries to Your Network](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [AssociateResolverRule](#) in *AWS CLI Command Reference*.

create-firewall-domain-list

The following code example shows how to use `create-firewall-domain-list`.

AWS CLI**To create a Route 53 Resolver DNS Firewall domain list**

The following `create-firewall-domain-list` example creates a Route 53 Resolver DNS Firewall domain list, named `test`, in your AWS account.

```
aws route53resolver create-firewall-domain-list \
  --creator-request-id my-request-id \
  --name test
```

Output:

```
{
  "FirewallDomainList": {
    "Id": "rslvr-fdl-d61cbb2cbexample",
    "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-domain-list/
rslvr-fdl-d61cbb2cbexample",
```

```
    "Name": "test",
    "DomainCount": 0,
    "Status": "COMPLETE",
    "StatusMessage": "Created Firewall Domain List",
    "CreatorRequestId": "my-request-id",
    "CreationTime": "2021-05-25T15:55:51.115365Z",
    "ModificationTime": "2021-05-25T15:55:51.115365Z"
  }
}
```

For more information, see [Managing your own domain lists](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [CreateFirewallDomainList](#) in *AWS CLI Command Reference*.

create-firewall-rule-group

The following code example shows how to use `create-firewall-rule-group`.

AWS CLI

To create a Firewall rule group

The following `create-firewall-rule-group` example creates a DNS Firewall rule group.

```
aws route53resolver create-firewall-rule-group \
  --creator-request-id my-request-id \
  --name test
```

Output:

```
{
  "FirewallRuleGroup": {
    "Id": "rslvr-frg-47f93271fexample",
    "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-rule-group/rslvr-frg-47f93271fexample",
    "Name": "test",
    "RuleCount": 0,
    "Status": "COMPLETE",
    "StatusMessage": "Created Firewall Rule Group",
    "OwnerId": "123456789012",
    "CreatorRequestId": "my-request-id",
```

```

    "ShareStatus": "NOT_SHARED",
    "CreationTime": "2021-05-25T18:59:26.490017Z",
    "ModificationTime": "2021-05-25T18:59:26.490017Z"
  }
}

```

For more information, see [Managing rule groups and rules in DNS Firewall](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [CreateFirewallRuleGroup](#) in *AWS CLI Command Reference*.

create-firewall-rule

The following code example shows how to use `create-firewall-rule`.

AWS CLI

To create a firewall rule

The following `create-firewall-rule` example creates a firewall rule in a DNS Firewall rule for domains listed in a DNS Firewall domain list.

```

aws route53resolver create-firewall-rule \
  --name allow-rule \
  --firewall-rule-group-id rslvr-frg-47f93271fexample \
  --firewall-domain-list-id rslvr-fdl-9e956e9ffexample \
  --priority 101 \
  --action ALLOW

```

Output:

```

{
  "FirewallRule": {
    "FirewallRuleGroupId": "rslvr-frg-47f93271fexample",
    "FirewallDomainListId": "rslvr-fdl-9e956e9ffexample",
    "Name": "allow-rule",
    "Priority": 101,
    "Action": "ALLOW",
    "CreatorRequestId": "d81e3fb7-020b-415e-939f-EXAMPLE11111",
    "CreationTime": "2021-05-25T21:44:00.346093Z",
    "ModificationTime": "2021-05-25T21:44:00.346093Z"
  }
}

```

```
}
}
```

For more information, see [Managing rule groups and rules in DNS Firewall](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [CreateFirewallRule](#) in *AWS CLI Command Reference*.

create-resolver-endpoint

The following code example shows how to use `create-resolver-endpoint`.

AWS CLI

To create an inbound Resolver endpoint

The following `create-resolver-endpoint` example creates an inbound Resolver endpoint. You can use the same command to create both inbound and outbound endpoints.

```
aws route53resolver create-resolver-endpoint --name my-inbound-endpoint --creator-request-id 2020-01-01-18:47 --security-group-ids "sg-f62bexam" --direction INBOUND --ip-addresses SubnetId=subnet-ba47exam,Ip=192.0.2.255 SubnetId=subnet-12d8exam,Ip=192.0.2.254
```

Output:

```
{
  "ResolverEndpoint": {
    "Id": "rslvr-in-f9ab8a03f1example",
    "CreatorRequestId": "2020-01-01-18:47",
    "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-endpoint/rslvr-in-f9ab8a03f1example",
    "Name": "my-inbound-endpoint",
    "SecurityGroupIds": [
      "sg-f62bexam"
    ],
    "Direction": "INBOUND",
    "IpAddressCount": 2,
    "HostVPCId": "vpc-304examp",
    "Status": "CREATING",
    "StatusMessage": "[Trace id: 1-5dc1ff84-f3477826e4a190025example] Creating the Resolver Endpoint",
    "CreationTime": "2020-01-01T23:02:29.583Z",
```

```
    "ModificationTime": "2020-01-01T23:02:29.583Z"
  }
}
```

To create an outbound Resolver endpoint

The following `create-resolver-endpoint` example creates an outbound resolver endpoint using the values in the JSON-formatted document `create-outbound-resolver-endpoint.json`.

```
aws route53resolver create-resolver-endpoint \
  --cli-input-json file://c:\temp\create-outbound-resolver-endpoint.json
```

Contents of `create-outbound-resolver-endpoint.json`:

```
{
  "CreatorRequestId": "2020-01-01-18:47",
  "Direction": "OUTBOUND",
  "IpAddresses": [
    {
      "Ip": "192.0.2.255",
      "SubnetId": "subnet-ba47exam"
    },
    {
      "Ip": "192.0.2.254",
      "SubnetId": "subnet-12d8exam"
    }
  ],
  "Name": "my-outbound-endpoint",
  "SecurityGroupIds": [ "sg-05cd7b25d6example" ],
  "Tags": [
    {
      "Key": "my-key-name",
      "Value": "my-key-value"
    }
  ]
}
```

For more information, see [Resolving DNS Queries Between VPCs and Your Network](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [CreateResolverEndpoint](#) in *AWS CLI Command Reference*.

create-resolver-rule

The following code example shows how to use `create-resolver-rule`.

AWS CLI

To create a Resolver rule

The following `create-resolver-rule` example creates a Resolver forwarding rule. The rule uses the outbound endpoint `rslvr-out-d5e5920e37example` to forward DNS queries for `example.com` to the IP addresses `10.24.8.75` and `10.24.8.156`.

```
aws route53resolver create-resolver-rule \
  --creator-request-id 2020-01-02-18:47 \
  --domain-name example.com \
  --name my-rule \
  --resolver-endpoint-id rslvr-out-d5e5920e37example \
  --rule-type FORWARD \
  --target-ips "Ip=10.24.8.75" "Ip=10.24.8.156"
```

Output:

```
{
  "ResolverRule": {
    "Status": "COMPLETE",
    "RuleType": "FORWARD",
    "ResolverEndpointId": "rslvr-out-d5e5920e37example",
    "Name": "my-rule",
    "DomainName": "example.com.",
    "CreationTime": "2022-05-10T21:35:30.923187Z",
    "TargetIps": [
      {
        "Ip": "10.24.8.75",
        "Port": 53
      },
      {
        "Ip": "10.24.8.156",
        "Port": 53
      }
    ],
    "CreatorRequestId": "2022-05-10-16:33",
    "ModificationTime": "2022-05-10T21:35:30.923187Z",
```

```

    "ShareStatus": "NOT_SHARED",
    "Arn": "arn:aws:route53resolver:us-east-1:111117012054:resolver-rule/rslvr-rr-b1e0b905e93611111",
    "OwnerId": "111111111111",
    "Id": "rslvr-rr-rslvr-rr-b1e0b905e93611111",
    "StatusMessage": "[Trace id: 1-22222222-3e56afcc71a3724664f22e24]
Successfully created Resolver Rule."
  }
}

```

- For API details, see [CreateResolverRule](#) in *AWS CLI Command Reference*.

delete-firewall-domain-list

The following code example shows how to use `delete-firewall-domain-list`.

AWS CLI

To delete a Route 53 Resolver DNS Firewall domain list

The following `delete-firewall-domain-list` example deletes a Route 53 Resolver DNS Firewall domain list, named `test`, in your AWS account.

```
aws route53resolver delete-firewall-domain-list \
  --firewall-domain-list-id rslvr-fdl-9e956e9ffexample
```

Output:

```

{
  "FirewallDomainList": {
    "Id": "rslvr-fdl-9e956e9ffexample",
    "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-domain-list/rslvr-fdl-9e956e9ffexample",
    "Name": "test",
    "DomainCount": 6,
    "Status": "DELETING",
    "StatusMessage": "Deleting the Firewall Domain List",
    "CreatorRequestId": "my-request-id",
    "CreationTime": "2021-05-25T15:55:51.115365Z",
    "ModificationTime": "2021-05-25T18:58:05.588024Z"
  }
}

```

```
}
```

For more information, see [Managing your own domain lists](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [DeleteFirewallDomainList](#) in *AWS CLI Command Reference*.

delete-firewall-rule-group

The following code example shows how to use `delete-firewall-rule-group`.

AWS CLI

To delete a firewall rule group

The following `delete-firewall-rule-group` example deletes a firewall rule group.

```
aws route53resolver delete-firewall-rule-group \  
  --firewall-rule-group-id rslvr-frg-47f93271fexample
```

Output:

```
{  
  "FirewallRuleGroup": {  
    "Id": "rslvr-frg-47f93271fexample",  
    "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-rule-group/  
rslvr-frg-47f93271fexample",  
    "Name": "test",  
    "RuleCount": 0,  
    "Status": "UPDATING",  
    "StatusMessage": "Updating Firewall Rule Group",  
    "OwnerId": "123456789012",  
    "CreatorRequestId": "my-request-id",  
    "ShareStatus": "NOT_SHARED",  
    "CreationTime": "2021-05-25T18:59:26.490017Z",  
    "ModificationTime": "2021-05-25T21:51:53.028688Z"  
  }  
}
```

For more information, see [Managing rule groups and rules in DNS Firewall](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [DeleteFirewallRuleGroup](#) in *AWS CLI Command Reference*.

delete-firewall-rule

The following code example shows how to use `delete-firewall-rule`.

AWS CLI

To delete a firewall rule

The following `delete-firewall-rule` example deletes a specified firewall rule.

```
aws route53resolver delete-firewall-rule \  
  --firewall-rule-group-id rslvr-frg-47f93271fexample \  
  --firewall-domain-list-id rslvr-fdl-9e956e9ffexample
```

Output:

```
{  
  "FirewallRule": {  
    "FirewallRuleGroupId": "rslvr-frg-47f93271fexample",  
    "FirewallDomainListId": "rslvr-fdl-9e956e9ffexample",  
    "Name": "allow-rule",  
    "Priority": 102,  
    "Action": "ALLOW",  
    "CreatorRequestId": "d81e3fb7-020b-415e-939f-EXAMPLE11111",  
    "CreationTime": "2021-05-25T21:44:00.346093Z",  
    "ModificationTime": "2021-05-25T21:45:59.611600Z"  
  }  
}
```

For more information, see [Managing rule groups and rules in DNS Firewall](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [DeleteFirewallRule](#) in *AWS CLI Command Reference*.

delete-resolver-endpoint

The following code example shows how to use `delete-resolver-endpoint`.

AWS CLI

To delete a Resolver endpoint

The following `delete-resolver-endpoint` example deletes the specified endpoint.

Important If you delete an inbound endpoint, DNS queries from your network are no longer forwarded to Resolver in the VPC that you specified in the endpoint. If you delete an outbound endpoint, Resolver stops forwarding DNS queries from your VPC to your network for rules that specify the deleted outbound endpoint.

```
aws route53resolver delete-resolver-endpoint \  
  --resolver-endpoint-id rslvr-in-497098ad59example
```

Output:

```
{  
  "ResolverEndpoint": {  
    "Id": "rslvr-in-497098ad59example",  
    "CreatorRequestId": "AWSConsole.25.157290example",  
    "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-endpoint/  
rslvr-in-497098ad59example",  
    "Name": "my-inbound-endpoint",  
    "SecurityGroupIds": [  
      "sg-05cd7b25d6example"  
    ],  
    "Direction": "INBOUND",  
    "IpAddressCount": 5,  
    "HostVPCId": "vpc-304bexam",  
    "Status": "DELETING",  
    "StatusMessage": "[Trace id: 1-5dc5b658-811b5be0922bbc382example] Deleting  
ResolverEndpoint.",  
    "CreationTime": "2020-01-01T23:25:45.538Z",  
    "ModificationTime": "2020-01-02T23:25:45.538Z"  
  }  
}
```

- For API details, see [DeleteResolverEndpoint](#) in *AWS CLI Command Reference*.

delete-resolver-rule

The following code example shows how to use `delete-resolver-rule`.

AWS CLI

To delete a Resolver rule

The following `delete-resolver-rule` example deletes the specified rule.

Note If a rule is associated with any VPCs, you must first disassociate the rule from the VPCs before you can delete it.

```
aws route53resolver delete-resolver-rule \  
  --resolver-rule-id rslvr-rr-5b3809426bexample
```

Output:

```
{  
  "ResolverRule": {  
    "Id": "rslvr-rr-5b3809426bexample",  
    "CreatorRequestId": "2020-01-03-18:47",  
    "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-rule/rslvr-  
rr-5b3809426bexample",  
    "DomainName": "zenith.example.com.",  
    "Status": "DELETING",  
    "StatusMessage": "[Trace id: 1-5dc5e05b-602e67b052cb74f05example] Deleting  
Resolver Rule.",  
    "RuleType": "FORWARD",  
    "Name": "my-resolver-rule",  
    "TargetIps": [  
      {  
        "Ip": "192.0.2.50",  
        "Port": 53  
      }  
    ],  
    "ResolverEndpointId": "rslvr-out-d5e5920e3example",  
    "OwnerId": "111122223333",  
    "ShareStatus": "NOT_SHARED"  
  }  
}
```

- For API details, see [DeleteResolverRule](#) in *AWS CLI Command Reference*.

disassociate-firewall-rule-group

The following code example shows how to use `disassociate-firewall-rule-group`.

AWS CLI

To disassociate a firewall rule group from a VPC

The following `disassociate-firewall-rule-group` example disassociates a DNS Firewall rule group from an Amazon VPC.

```
aws route53resolver disassociate-firewall-rule-group \  
  --firewall-rule-group-association-id rslvr-frgassoc-57e8873d7example
```

Output:

```
{  
  "FirewallRuleGroupAssociation": {  
    "Id": "rslvr-frgassoc-57e8873d7example",  
    "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-rule-group-  
association/rslvr-frgassoc-57e8873d7example",  
    "FirewallRuleGroupId": "rslvr-frg-47f93271fexample",  
    "VpcId": "vpc-31e92222",  
    "Name": "test-association",  
    "Priority": 103,  
    "MutationProtection": "DISABLED",  
    "Status": "DELETING",  
    "StatusMessage": "Deleting the Firewall Rule Group Association",  
    "CreatorRequestId": "2ca1a304-32b3-4f5f-bc4c-EXAMPLE11111",  
    "CreationTime": "2021-05-25T21:47:48.755768Z",  
    "ModificationTime": "2021-05-25T21:51:02.377887Z"  
  }  
}
```

For more information, see [Managing associations between your VPC and Route 53 Resolver DNS Firewall rule groups](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [DisassociateFirewallRuleGroup](#) in *AWS CLI Command Reference*.

disassociate-resolver-endpoint-ip-address

The following code example shows how to use `disassociate-resolver-endpoint-ip-address`.

AWS CLI

To disassociate an IP address from a Resolver endpoint

The following `disassociate-resolver-endpoint-ip-address` example removes an IP address from a specified Resolver inbound or outbound endpoint.

Note An endpoint must have at least two IP addresses. If an endpoint currently has only two IP addresses and you want to replace one address with another address, you must first use [associate-resolver-endpoint-ip-address](#) to associate the new IP address. Then you can disassociate one of the original IP addresses from the endpoint.

```
aws route53resolver disassociate-resolver-endpoint-ip-address \
  --resolver-endpoint-id rslvr-in-f9ab8a03f1example \
  --ip-address="SubnetId=subnet-12d8a459,Ip=172.31.40.121"
```

Output:

```
{
  "ResolverEndpoint": {
    "Id": "rslvr-in-f9ab8a03f1example",
    "CreatorRequestId": "2020-01-01-18:47",
    "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-endpoint/
rslvr-in-f9ab8a03f1example",
    "Name": "my-inbound-endpoint",
    "SecurityGroupIds": [
      "sg-f62bexam"
    ],
    "Direction": "INBOUND",
    "IpAddressCount": 3,
    "HostVPCId": "vpc-304bexam",
    "Status": "UPDATING",
    "StatusMessage": "Updating the Resolver Endpoint",
    "CreationTime": "2020-01-01T23:02:29.583Z",
    "ModificationTime": "2020-01-05T23:02:29.583Z"
  }
}
```

- For API details, see [DisassociateResolverEndpointIpAddress](#) in *AWS CLI Command Reference*.

disassociate-resolver-rule

The following code example shows how to use `disassociate-resolver-rule`.

AWS CLI

To disassociate a Resolver rule from an Amazon VPC

The following `disassociate-resolver-rule` example removes the association between the specified Resolver rule and the specified VPC. You can disassociate a rule from a VPC in the following circumstances:

For DNS queries that originate in this VPC, you want Resolver to stop forwarding queries to your network for the domain name that is specified in the rule. You want to delete the forwarding rule. If a rule is currently associated with one or more VPCs, you must disassociate the rule from all VPCs before you can delete it.

```
aws route53resolver disassociate-resolver-rule \  
  --resolver-rule-id rslvr-rr-4955cb98ceexample \  
  --vpc-id vpc-304bexam
```

Output:

```
{  
  "ResolverRuleAssociation": {  
    "Id": "rslvr-rrassoc-322f4e8b9cexample",  
    "ResolverRuleId": "rslvr-rr-4955cb98ceexample",  
    "Name": "my-resolver-rule-association",  
    "VPCId": "vpc-304bexam",  
    "Status": "DELETING",  
    "StatusMessage": "[Trace id: 1-5dc5ffa2-a26c38004c1f94006example] Deleting  
Association"  
  }  
}
```

- For API details, see [DisassociateResolverRule](#) in *AWS CLI Command Reference*.

get-firewall-config

The following code example shows how to use `get-firewall-config`.

AWS CLI

To get a firewall config for a VPC

The following `get-firewall-config` example retrieves the DNS Firewall behavior for the specified VPC.

```
aws route53resolver get-firewall-config \  
  --vpc-id vpc-304bexam
```

```
--resource-id vpc-31e92222
```

Output:

```
{
  "FirewallConfig": {
    "Id": "rslvr-fc-86016850cexample",
    "ResourceId": "vpc-31e92222",
    "OwnerId": "123456789012",
    "FirewallFailOpen": "DISABLED"
  }
}
```

For more information, see [DNS Firewall VPC configuration](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [GetFirewallConfig](#) in *AWS CLI Command Reference*.

get-firewall-domain-list

The following code example shows how to use `get-firewall-domain-list`.

AWS CLI

To get a Route 53 Resolver DNS Firewall domain list

The following `get-firewall-domain-list` example retrieves the domain list with the ID you specify.

```
aws route53resolver get-firewall-domain-list \
  --firewall-domain-list-id rslvr-fdl-42b60677cexample
```

Output:

```
{
  "FirewallDomainList": {
    "Id": "rslvr-fdl-9e956e9ffexample",
    "Arn": "arn:aws:route53resolver:us-west-2:123457689012:firewall-domain-list/rslvr-fdl-42b60677cexample",
    "Name": "test",
  }
}
```

```
    "DomainCount": 0,  
    "Status": "COMPLETE",  
    "StatusMessage": "Created Firewall Domain List",  
    "CreatorRequestId": "my-request-id",  
    "CreationTime": "2021-05-25T15:55:51.115365Z",  
    "ModificationTime": "2021-05-25T15:55:51.115365Z"  
  }  
}
```

For more information, see [Managing your own domain lists](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [GetFirewallDomainList](#) in *AWS CLI Command Reference*.

get-firewall-rule-group-association

The following code example shows how to use `get-firewall-rule-group-association`.

AWS CLI

To get a firewall rule group association

The following `get-firewall-rule-group-association` example retrieves a firewall rule group association.

```
aws route53resolver get-firewall-rule-group-association \  
  --firewall-rule-group-association-id rslvr-frgassoc-57e8873d7example
```

Output:

```
{  
  "FirewallRuleGroupAssociation": {  
    "Id": "rslvr-frgassoc-57e8873d7example",  
    "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-rule-group-  
association/rslvr-frgassoc-57e8873d7example",  
    "FirewallRuleGroupId": "rslvr-frg-47f93271fexample",  
    "VpcId": "vpc-31e92222",  
    "Name": "test-association",  
    "Priority": 101,  
    "MutationProtection": "DISABLED",  
    "Status": "COMPLETE",  
    "StatusMessage": "Finished rule group association update",  
  }  
}
```



```

    "CreatorRequestId": "2ca1a304-32b3-4f5f-bc4c-EXAMPLE11111",
    "CreationTime": "2021-05-25T21:47:48.755768Z",
    "ModificationTime": "2021-05-25T21:47:48.755768Z"
  }
}

```

For more information, see [Managing associations between your VPC and Route 53 Resolver DNS Firewall rule groups](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [GetFirewallRuleGroupAssociation](#) in *AWS CLI Command Reference*.

get-firewall-rule-group-policy

The following code example shows how to use `get-firewall-rule-group-policy`.

AWS CLI

To get an AWS IAM policy

The following `get-firewall-rule-group-policy` example gets the AWS Identity and Access Management (AWS IAM) policy for sharing the specified rule group.

```

aws route53resolver get-firewall-rule-group-policy \
  --arn arn:aws:route53resolver:us-west-2:AWS_ACCOUNT_ID:firewall-rule-group/
  rslvr-frg-47f93271fexample

```

Output:

```

{
  "FirewallRuleGroupPolicy": "{\"Version\":\"2012-10-17\",
  \"Statement\": [{\"Sid\":\"test\", \"Effect\":\"Allow\", \"Principal\": {\"AWS\": \"arn:aws:iam::AWS_ACCOUNT_ID:root\"}, \"Action\": [\"route53resolver:GetFirewallRuleGroup\", \"route53resolver:ListFirewallRuleGroups\"], \"Resource\": \"arn:aws:route53resolver:us-east-1:AWS_ACCOUNT_ID:firewall-rule-group/rslvr-frg-47f93271fexample\"}]}\"
}

```

For more information, see [Managing rule groups and rules in DNS Firewall](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [GetFirewallRuleGroupPolicy](#) in *AWS CLI Command Reference*.

get-firewall-rule-group

The following code example shows how to use `get-firewall-rule-group`.

AWS CLI

To get a Firewall rule group

The following `get-firewall-rule-group` example retrieves information about a DNS Firewall rule group with the ID you provide.

```
aws route53resolver get-firewall-rule-group \  
  --firewall-rule-group-id rslvr-frg-47f93271fexample
```

Output:

```
{  
  "FirewallRuleGroup": {  
    "Id": "rslvr-frg-47f93271fexample",  
    "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-rule-group/  
rslvr-frg-47f93271fexample",  
    "Name": "test",  
    "RuleCount": 0,  
    "Status": "COMPLETE",  
    "StatusMessage": "Created Firewall Rule Group",  
    "OwnerId": "123456789012",  
    "CreatorRequestId": "my-request-id",  
    "ShareStatus": "NOT_SHARED",  
    "CreationTime": "2021-05-25T18:59:26.490017Z",  
    "ModificationTime": "2021-05-25T18:59:26.490017Z"  
  }  
}
```

For more information, see [Managing rule groups and rules in DNS Firewall](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [GetFirewallRuleGroup](#) in *AWS CLI Command Reference*.

get-resolver-endpoint

The following code example shows how to use `get-resolver-endpoint`.

AWS CLI

To get information about a Resolver endpoint

The following `get-resolver-endpoint` example displays details for the outbound specified endpoint. You can use `get-resolver-endpoint` for both inbound and outbound endpoints by specifying the applicable endpoint ID.

```
aws route53resolver get-resolver-endpoint \  
  --resolver-endpoint-id rslvr-out-d5e5920e37example
```

Output:

```
{  
  "ResolverEndpoint": {  
    "Id": "rslvr-out-d5e5920e37example",  
    "CreatorRequestId": "2020-01-01-18:47",  
    "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-endpoint/  
rslvr-out-d5e5920e37example",  
    "Name": "my-outbound-endpoint",  
    "SecurityGroupIds": [  
      "sg-05cd7b25d6example"  
    ],  
    "Direction": "OUTBOUND",  
    "IpAddressCount": 2,  
    "HostVPCId": "vpc-304bexam",  
    "Status": "OPERATIONAL",  
    "StatusMessage": "This Resolver Endpoint is operational.",  
    "CreationTime": "2020-01-01T23:50:50.979Z",  
    "ModificationTime": "2020-01-02T23:50:50.979Z"  
  }  
}
```

For more information, see [Values That You Specify When You Create or Edit Inbound Endpoints](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [GetResolverEndpoint](#) in *AWS CLI Command Reference*.

`get-resolver-rule-association`

The following code example shows how to use `get-resolver-rule-association`.

AWS CLI

To get information about the association between a Resolver rule and a VPC

The following `get-resolver-rule-association` example displays details about the association between a specified Resolver rule and a VPC. You associate a resolver rule and a VPC using [associate-resolver-rule](#).

```
aws route53resolver get-resolver-rule-association \  
  --resolver-rule-association-id rslvr-rrassoc-d61cbb2c8bexample
```

Output:

```
{  
  "ResolverRuleAssociation": {  
    "Id": "rslvr-rrassoc-d61cbb2c8bexample",  
    "ResolverRuleId": "rslvr-rr-42b60677c0example",  
    "Name": "my-resolver-rule-association",  
    "VPCId": "vpc-304bexam",  
    "Status": "COMPLETE",  
    "StatusMessage": ""  
  }  
}
```

- For API details, see [GetResolverRuleAssociation](#) in *AWS CLI Command Reference*.

get-resolver-rule

The following code example shows how to use `get-resolver-rule`.

AWS CLI

To get information about a Resolver rule

The following `get-resolver-rule` example displays details about the specified Resolver rule, such as the domain name that the rule forwards DNS queries for and the ID of the outbound resolver endpoint that the rule is associated with.

```
aws route53resolver get-resolver-rule \  
  --resolver-rule-id rslvr-rr-42b60677c0example
```

Output:

```
{
  "ResolverRule": {
    "Id": "rslvr-rr-42b60677c0example",
    "CreatorRequestId": "2020-01-01-18:47",
    "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-rule/rslvr-rr-42b60677c0example",
    "DomainName": "example.com.",
    "Status": "COMPLETE",
    "StatusMessage": "[Trace id: 1-5dc4b177-ff1d9d001a0f80005example]
Successfully created Resolver Rule.",
    "RuleType": "FORWARD",
    "Name": "my-rule",
    "TargetIps": [
      {
        "Ip": "192.0.2.45",
        "Port": 53
      }
    ],
    "ResolverEndpointId": "rslvr-out-d5e5920e37example",
    "OwnerId": "111122223333",
    "ShareStatus": "NOT_SHARED"
  }
}
```

For more information, see [Values That You Specify When You Create or Edit Rules](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [GetResolverRule](#) in *AWS CLI Command Reference*.

import-firewall-domains

The following code example shows how to use `import-firewall-domains`.

AWS CLI**To import domains into a domain list**

The following `import-firewall-domains` example imports a set of domains from a file into a DNS Firewall domain list that you specify.

```
aws route53resolver import-firewall-domains \
```

```
--firewall-domain-list-id rslvr-fdl-d61cbb2cbexample \  
--operation REPLACE \  
--domain-file-url s3://PATH/TO/YOUR/FILE
```

Output:

```
{  
  "Id": "rslvr-fdl-d61cbb2cbexample",  
  "Name": "test",  
  "Status": "IMPORTING",  
  "StatusMessage": "Importing domains from provided file."  
}
```

For more information, see [Managing your own domain lists](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [ImportFirewallDomains](#) in *AWS CLI Command Reference*.

list-firewall-configs

The following code example shows how to use `list-firewall-configs`.

AWS CLI

To list firewall configs

The following `list-firewall-configs` example lists your DNS Firewall configurations.

```
aws route53resolver list-firewall-configs
```

Output:

```
{  
  "FirewallConfigs": [  
    {  
      "Id": "rslvr-fc-86016850cexample",  
      "ResourceId": "vpc-31e92222",  
      "OwnerId": "123456789012",  
      "FirewallFailOpen": "DISABLED"  
    }  
  ]  
}
```

For more information, see [DNS Firewall VPC configuration](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [ListFirewallConfigs](#) in *AWS CLI Command Reference*.

list-firewall-domain-lists

The following code example shows how to use `list-firewall-domain-lists`.

AWS CLI

To list all of Route 53 Resolver DNS Firewall domain lists

The following `list-firewall-domain-lists` example lists all the domain lists.

```
aws route53resolver list-firewall-domain-lists
```

Output:

```
{
  "FirewallDomainLists": [
    {
      "Id": "rslvr-fdl-2c46f2ecfexample",
      "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-domain-list/rslvr-fdl-2c46f2ecfexample",
      "Name": "AWSManagedDomainsMalwareDomainList",
      "CreatorRequestId": "AWSManagedDomainsMalwareDomainList",
      "ManagedOwnerName": "Route 53 Resolver DNS Firewall"
    },
    {
      "Id": "rslvr-fdl-aa970e9e1example",
      "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-domain-list/rslvr-fdl-aa970e9e1example",
      "Name": "AWSManagedDomainsBotnetCommandandControl",
      "CreatorRequestId": "AWSManagedDomainsBotnetCommandandControl",
      "ManagedOwnerName": "Route 53 Resolver DNS Firewall"
    },
    {
      "Id": "rslvr-fdl-42b60677cexample",
      "Arn": "arn:aws:route53resolver:us-west-2:123456789111:firewall-domain-list/rslvr-fdl-42b60677cexample",
      "Name": "test",
      "CreatorRequestId": "my-request-id"
    }
  ]
}
```

```
    }  
  ]  
}
```

For more information, see [Route 53 Resolver DNS Firewall domain lists](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [ListFirewallDomainLists](#) in *AWS CLI Command Reference*.

list-firewall-domains

The following code example shows how to use `list-firewall-domains`.

AWS CLI

To list domains in a domain list

The following `list-firewall-domains` example lists the domains in a DNS Firewall domain list that you specify.

```
aws route53resolver list-firewall-domains \  
  --firewall-domain-list-id rslvr-fd1-d61cbb2cbexample
```

Output:

```
{  
  "Domains": [  
    "test1.com.",  
    "test2.com.",  
    "test3.com."  
  ]  
}
```

For more information, see [Managing your own domain lists](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [ListFirewallDomains](#) in *AWS CLI Command Reference*.

list-firewall-rule-group-associations

The following code example shows how to use `list-firewall-rule-group-associations`.

AWS CLI

To list DNS Firewall rule group associations

The following `list-firewall-rule-group-associations` example lists your DNS Firewall rule group associations with Amazon VPCs.

```
aws route53resolver list-firewall-rule-group-associations
```

Output:

```
{
  "FirewallRuleGroupAssociations": [
    {
      "Id": "rslvr-frgassoc-57e8873d7example",
      "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-rule-
group-association/rslvr-frgassoc-57e8873d7example",
      "FirewallRuleGroupId": "rslvr-frg-47f93271fexample",
      "VpcId": "vpc-31e92222",
      "Name": "test-association",
      "Priority": 101,
      "MutationProtection": "DISABLED",
      "Status": "UPDATING",
      "StatusMessage": "Creating Firewall Rule Group Association",
      "CreatorRequestId": "2ca1a304-32b3-4f5f-bc4c-EXAMPLE11111",
      "CreationTime": "2021-05-25T21:47:48.755768Z",
      "ModificationTime": "2021-05-25T21:47:48.755768Z"
    }
  ]
}
```

For more information, see [Managing associations between your VPC and Route 53 Resolver DNS Firewall rule group](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [ListFirewallRuleGroupAssociations](#) in *AWS CLI Command Reference*.

list-firewall-rule-groups

The following code example shows how to use `list-firewall-rule-groups`.

AWS CLI

To get a list of your Firewall rule groups

The following `list-firewall-rule-groups` example lists your DNS Firewall rule groups.

```
aws route53resolver list-firewall-rule-groups
```

Output:

```
{
  "FirewallRuleGroups": [
    {
      "Id": "rslvr-frg-47f93271fexample",
      "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-rule-
group/rslvr-frg-47f93271fexample",
      "Name": "test",
      "OwnerId": "123456789012",
      "CreatorRequestId": "my-request-id",
      "ShareStatus": "NOT_SHARED"
    }
  ]
}
```

For more information, see [Managing rule groups and rules in DNS Firewall](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [ListFirewallRuleGroups](#) in *AWS CLI Command Reference*.

list-firewall-rules

The following code example shows how to use `list-firewall-rules`.

AWS CLI

To list firewall rules

The following `list-firewall-rules` example list all of your DNS Firewall rules within a firewall rule group.

```
aws route53resolver list-firewall-rules \
  --firewall-rule-group-id rslvr-frg-47f93271fexample
```

Output:

```
{
```

```
"FirewallRules": [  
  {  
    "FirewallRuleGroupId": "rslvr-frg-47f93271fexample",  
    "FirewallDomainListId": "rslvr-fdl-9e956e9ffexample",  
    "Name": "allow-rule",  
    "Priority": 101,  
    "Action": "ALLOW",  
    "CreatorRequestId": "d81e3fb7-020b-415e-939f-EXAMPLE11111",  
    "CreationTime": "2021-05-25T21:44:00.346093Z",  
    "ModificationTime": "2021-05-25T21:44:00.346093Z"  
  }  
]  
}
```

For more information, see [Managing rule groups and rules in DNS Firewall](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [ListFirewallRules](#) in *AWS CLI Command Reference*.

list-resolver-endpoint-ip-addresses

The following code example shows how to use `list-resolver-endpoint-ip-addresses`.

AWS CLI

To list IP addresses for a specified inbound or outbound endpoint

The following `list-resolver-endpoint-ip-addresses` example lists information about the IP addresses that are associated with the inbound endpoint `rslvr-in-f9ab8a03f1example`. You can also use `list-resolver-endpoint-ip-addresses` for outbound endpoints by specifying the applicable endpoint ID.

```
aws route53resolver list-resolver-endpoint-ip-addresses \  
  --resolver-endpoint-id rslvr-in-f9ab8a03f1example
```

Output:

```
{  
  "MaxResults": 10,  
  "IpAddresses": [  
    {  
      "IpId": "rni-1de60cdbfeexample",
```

```

        "SubnetId": "subnet-ba47exam",
        "Ip": "192.0.2.44",
        "Status": "ATTACHED",
        "StatusMessage": "This IP address is operational.",
        "CreationTime": "2020-01-03T23:02:29.587Z",
        "ModificationTime": "2020-01-03T23:03:05.555Z"
    },
    {
        "IpId": "rni-aac7085e38example",
        "SubnetId": "subnet-12d8exam",
        "Ip": "192.0.2.45",
        "Status": "ATTACHED",
        "StatusMessage": "This IP address is operational.",
        "CreationTime": "2020-01-03T23:02:29.593Z",
        "ModificationTime": "2020-01-03T23:02:55.060Z"
    }
]
}

```

For more information about the values in the output, see [Values That You Specify When You Create or Edit Inbound Endpoints](#), and [Values That You Specify When You Create or Edit Outbound Endpoints](#), both in the *Amazon Route 53 Developer Guide*.

- For API details, see [ListResolverEndpointIpAddresses](#) in *AWS CLI Command Reference*.

list-resolver-endpoints

The following code example shows how to use `list-resolver-endpoints`.

AWS CLI

To list Resolver endpoints in an AWS Region

The following `list-resolver-endpoints` example lists the inbound and outbound Resolver endpoints that exist in the current account.

```
aws route53resolver list-resolver-endpoints
```

Output:

```
{
  "MaxResults": 10,
```

```

"ResolverEndpoints": [
  {
    "Id": "rslvr-in-497098ad59example",
    "CreatorRequestId": "2020-01-01-18:47",
    "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-
endpoint/rslvr-in-497098ad59example",
    "Name": "my-inbound-endpoint",
    "SecurityGroupIds": [
      "sg-05cd7b25d6example"
    ],
    "Direction": "INBOUND",
    "IpAddressCount": 2,
    "HostVPCId": "vpc-304bexam",
    "Status": "OPERATIONAL",
    "StatusMessage": "This Resolver Endpoint is operational.",
    "CreationTime": "2020-01-01T23:25:45.538Z",
    "ModificationTime": "2020-01-01T23:25:45.538Z"
  },
  {
    "Id": "rslvr-out-d5e5920e37example",
    "CreatorRequestId": "2020-01-01-18:48",
    "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-
endpoint/rslvr-out-d5e5920e37example",
    "Name": "my-outbound-endpoint",
    "SecurityGroupIds": [
      "sg-05cd7b25d6example"
    ],
    "Direction": "OUTBOUND",
    "IpAddressCount": 2,
    "HostVPCId": "vpc-304bexam",
    "Status": "OPERATIONAL",
    "StatusMessage": "This Resolver Endpoint is operational.",
    "CreationTime": "2020-01-01T23:50:50.979Z",
    "ModificationTime": "2020-01-01T23:50:50.979Z"
  }
]
}

```

- For API details, see [ListResolverEndpoints](#) in *AWS CLI Command Reference*.

list-resolver-rule-associations

The following code example shows how to use `list-resolver-rule-associations`.

AWS CLI

To list associations between Resolver rules and VPCs

The following `list-resolver-rule-associations` example lists the associations between resolver rules and VPCs in the current AWS account.

```
aws route53resolver list-resolver-rule-associations
```

Output:

```
{
  "MaxResults": 30,
  "ResolverRuleAssociations": [
    {
      "Id": "rslvr-autodefined-assoc-vpc-304bexam-internet-resolver",
      "ResolverRuleId": "rslvr-autodefined-rr-internet-resolver",
      "Name": "System Rule Association",
      "VPCId": "vpc-304bexam",
      "Status": "COMPLETE",
      "StatusMessage": ""
    },
    {
      "Id": "rslvr-rrassoc-d61cbb2c8bexample",
      "ResolverRuleId": "rslvr-rr-42b60677c0example",
      "Name": "my-resolver-rule-association",
      "VPCId": "vpc-304bexam",
      "Status": "COMPLETE",
      "StatusMessage": ""
    }
  ]
}
```

For more information, see [How Route 53 Resolver Forwards DNS Queries from Your VPCs to Your Network](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [ListResolverRuleAssociations](#) in *AWS CLI Command Reference*.

list-resolver-rules

The following code example shows how to use `list-resolver-rules`.

AWS CLI

To list Resolver rules

The following `list-resolver-rules` example lists all the Resolver rules in the current AWS account.

```
aws route53resolver list-resolver-rules
```

Output:

```
{
  "MaxResults": 30,
  "ResolverRules": [
    {
      "Id": "rslvr-autodefined-rr-internet-resolver",
      "CreatorRequestId": "",
      "Arn": "arn:aws:route53resolver:us-west-2::autodefined-rule/rslvr-
autodefined-rr-internet-resolver",
      "DomainName": ".",
      "Status": "COMPLETE",
      "RuleType": "RECURSIVE",
      "Name": "Internet Resolver",
      "OwnerId": "Route 53 Resolver",
      "ShareStatus": "NOT_SHARED"
    },
    {
      "Id": "rslvr-rr-42b60677c0example",
      "CreatorRequestId": "2020-01-01-18:47",
      "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-rule/
rslvr-rr-42b60677c0bc4e299",
      "DomainName": "example.com.",
      "Status": "COMPLETE",
      "StatusMessage": "[Trace id: 1-5dc4b177-ff1d9d001a0f80005example]
Successfully created Resolver Rule.",
      "RuleType": "FORWARD",
      "Name": "my-rule",
      "TargetIps": [
        {
          "Ip": "192.0.2.45",
          "Port": 53
        }
      ]
    }
  ],
}
```

```
        "ResolverEndpointId": "rslvr-out-d5e5920e37example",
        "OwnerId": "111122223333",
        "ShareStatus": "NOT_SHARED"
    }
]
}
```

For more information, see [How Route 53 Resolver Forwards DNS Queries from Your VPCs to Your Network](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [ListResolverRules](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags for a Resolver resource

The following `list-tags-for-resource` example lists the tags that are assigned to the specified Resolver rule.

```
aws route53resolver list-tags-for-resource \
    --resource-arn "arn:aws:route53resolver:us-west-2:111122223333:resolver-rule/
rslvr-rr-42b60677c0example"
```

Output:

```
{
  "Tags": [
    {
      "Key": "my-key-1",
      "Value": "my-value-1"
    },
    {
      "Key": "my-key-2",
      "Value": "my-value-2"
    }
  ]
}
```


For information about using tags for cost allocation, see [Using Cost Allocation Tags](#) in the *AWS Billing and Cost Management User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

put-firewall-rule-group-policy

The following code example shows how to use `put-firewall-rule-group-policy`.

AWS CLI

To attach an AWS IAM policy to share a Firewall rule group policy

The following `put-firewall-rule-group-policy` example attaches an AWS Identity and Access Management (AWS IAM) policy for sharing the rule group.

```
aws route53resolver put-firewall-rule-group-policy \
  --firewall-rule-group-policy "{\"Version\":\"2012-10-17\",
  \"Statement\": [{\"Sid\":\"test\", \"Effect\":\"Allow\", \"Principal
  \": {\"AWS\": \"arn:aws:iam::AWS_ACCOUNT_ID:root\"}, \"Action\":
  [\"route53resolver:GetFirewallRuleGroup\", \"route53resolver:ListFirewallRuleGroups
  \"], \"Resource\": \"arn:aws:route53resolver:us-east-1:AWS_ACCOUNT_ID:firewall-rule-
  group/rslvr-frg-47f93271fexample\"}]}"
```

Output:

```
{
  "ReturnValue": true
}
```

For more information, see [Managing rule groups and rules in DNS Firewall](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [PutFirewallRuleGroupPolicy](#) in *AWS CLI Command Reference*.

put-resolver-rule-policy

The following code example shows how to use `put-resolver-rule-policy`.

AWS CLI

To share a Resolver rule with another AWS account

The following `put-resolver-rule-policy` example specifies a Resolver rule that you want to share with another AWS account, the account that you want to share the rule with, and the rule-related operations that you want the account to be able to perform on the rules.

Note You must run this command using credentials from the same account that created the rule.

```
aws route53resolver put-resolver-rule-policy \
  --region us-east-1 \
  --arn "arn:aws:route53resolver:us-east-1:111122223333:resolver-rule/rslvr-
rr-42b60677c0example" \
  --resolver-rule-policy "{\"Version\": \"2012-10-17\", \
    \"Statement\": [ { \
      \"Effect\": \"Allow\", \
      \"Principal\": {\"AWS\": \"444455556666\" }, \
      \"Action\": [ \
        \"route53resolver:GetResolverRule\", \
        \"route53resolver:AssociateResolverRule\", \
        \"route53resolver:DisassociateResolverRule\", \
        \"route53resolver:ListResolverRules\", \
        \"route53resolver:ListResolverRuleAssociations\" ], \
      \"Resource\": [ \"arn:aws:route53resolver:us-east-1:111122223333:resolver-
rule/rslvr-rr-42b60677c0example\" ] } ] }"
```

Output:

```
{
  "ReturnValue": true
}
```

After you run `put-resolver-rule-policy`, you can run the following two Resource Access Manager (RAM) commands. You must use the account that you want to share the rule with:

`get-resource-share-invitations` returns the value `resourceShareInvitationArn`. You need this value to accept the invitation to use the shared rule. `accept-resource-share-invitation` accepts the invitation to use the shared rule.

For more information, see the following documentation:

[get-resource-share-invitationsaccept-resource-share-invitationsSharing Forwarding Rules with Other AWS Accounts and Using Shared Rules](#) in the *Amazon Route 53 Developer Guide*

- For API details, see [PutResolverRulePolicy](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To associate tags with a Resolver resource

The following `tag-resource` example associates two tag key/value pairs with the specified Resolver rule.

```
aws route53resolver tag-resource \  
  --resource-arn "arn:aws:route53resolver:us-west-2:111122223333:resolver-rule/  
rslvr-rr-42b60677c0example" \  
  --tags "Key=my-key-1,Value=my-value-1" "Key=my-key-2,Value=my-value-2"
```

This command produces no output.

For information about using tags for cost allocation, see [Using Cost Allocation Tags](#) in the *AWS Billing and Cost Management User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from a Resolver resource

The following `untag-resource` example removes two tags from the specified Resolver rule.

```
aws route53resolver untag-resource \  
  --resource-arn "arn:aws:route53resolver:us-west-2:111122223333:resolver-rule/  
rslvr-rr-42b60677c0example" \  
  --tag-keys my-key-1 my-key-2
```

This command produces no output. To confirm that the tags were removed, you can use [list-tags-for-resource](#).

For information about using tags for cost allocation, see [Using Cost Allocation Tags](#) in the *AWS Billing and Cost Management User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-firewall-config

The following code example shows how to use `update-firewall-config`.

AWS CLI

To update a firewall config

The following `update-firewall-config` example updates DNS Firewall configuration.

```
aws route53resolver update-firewall-config \  
  --resource-id vpc-31e92222 \  
  --firewall-fail-open DISABLED
```

Output:

```
{  
  "FirewallConfig": {  
    "Id": "rslvr-fc-86016850cexample",  
    "ResourceId": "vpc-31e92222",  
    "OwnerId": "123456789012",  
    "FirewallFailOpen": "DISABLED"  
  }  
}
```

For more information, see [DNS Firewall VPC configuration](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [UpdateFirewallConfig](#) in *AWS CLI Command Reference*.

update-firewall-domains

The following code example shows how to use `update-firewall-domains`.

AWS CLI

To update a domain list

The following `update-firewall-domains` example adds the domains to a domain list with the ID you provide.

```
aws route53resolver update-firewall-domains \  
  --firewall-domain-list-id rslvr-fdl-42b60677cexampleb \  
  --operation ADD \  
  --domains test1.com test2.com test3.com
```

Output:

```
{  
  "Id": "rslvr-fdl-42b60677cexample",  
  "Name": "test",  
  "Status": "UPDATING",  
  "StatusMessage": "Updating the Firewall Domain List"  
}
```

For more information, see [Managing your own domain lists](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [UpdateFirewallDomains](#) in *AWS CLI Command Reference*.

update-firewall-rule-group-association

The following code example shows how to use `update-firewall-rule-group-association`.

AWS CLI

To update a firewall rule group association

The following `update-firewall-rule-group-association` example updates a firewall rule group association.

```
aws route53resolver update-firewall-rule-group-association \  
  --firewall-rule-group-association-id rslvr-frgassoc-57e8873d7example \  
  --priority 103
```

Output:

```
{  
  "FirewallRuleGroupAssociation": {  
    "Id": "rslvr-frgassoc-57e8873d7example",
```

```

    "Arn": "arn:aws:route53resolver:us-west-2:123456789012:firewall-rule-group-
association/rslvr-frgassoc-57e8873d7example",
    "FirewallRuleGroupId": "rslvr-frg-47f93271fexample",
    "VpcId": "vpc-31e92222",
    "Name": "test-association",
    "Priority": 103,
    "MutationProtection": "DISABLED",
    "Status": "UPDATING",
    "StatusMessage": "Updating the Firewall Rule Group Association Attributes",
    "CreatorRequestId": "2ca1a304-32b3-4f5f-bc4c-EXAMPLE11111",
    "CreationTime": "2021-05-25T21:47:48.755768Z",
    "ModificationTime": "2021-05-25T21:50:09.272569Z"
  }
}

```

For more information, see [Managing associations between your VPC and Route 53 Resolver DNS Firewall rule group](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [UpdateFirewallRuleGroupAssociation](#) in *AWS CLI Command Reference*.

update-firewall-rule

The following code example shows how to use `update-firewall-rule`.

AWS CLI

To update a firewall rule

The following `update-firewall-rule` example updates a firewall rule with the parameters you specify.

```

aws route53resolver update-firewall-rule \
  --firewall-rule-group-id rslvr-frg-47f93271fexample \
  --firewall-domain-list-id rslvr-fdl-9e956e9ffexample \
  --priority 102

```

Output:

```

{
  "FirewallRule": {
    "FirewallRuleGroupId": "rslvr-frg-47f93271fexample",
    "FirewallDomainListId": "rslvr-fdl-9e956e9ffexample",
    "Name": "allow-rule",

```

```

    "Priority": 102,
    "Action": "ALLOW",
    "CreatorRequestId": "d81e3fb7-020b-415e-939f-EXAMPLE11111",
    "CreationTime": "2021-05-25T21:44:00.346093Z",
    "ModificationTime": "2021-05-25T21:45:59.611600Z"
  }
}

```

For more information, see [Managing rule groups and rules in DNS Firewall](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [UpdateFirewallRule](#) in *AWS CLI Command Reference*.

update-resolver-endpoint

The following code example shows how to use `update-resolver-endpoint`.

AWS CLI

To update the name of a Resolver endpoint

The following `update-resolver-endpoint` example updates the name of a Resolver endpoint. Updating other values isn't supported.

```

aws route53resolver update-resolver-endpoint \
  --resolver-endpoint-id rslvr-in-b5d45e32bdc445f09 \
  --name my-renamed-inbound-endpoint

```

Output:

```

{
  "ResolverEndpoint": {
    "Id": "rslvr-in-b5d45e32bdexample",
    "CreatorRequestId": "2020-01-02-18:48",
    "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-endpoint/rslvr-in-b5d45e32bdexample",
    "Name": "my-renamed-inbound-endpoint",
    "SecurityGroupIds": [
      "sg-f62bexam"
    ],
    "Direction": "INBOUND",
    "IpAddressCount": 2,
    "HostVPCId": "vpc-304bexam",
  }
}

```

```

    "Status": "OPERATIONAL",
    "StatusMessage": "This Resolver Endpoint is operational.",
    "CreationTime": "2020-01-01T18:33:59.265Z",
    "ModificationTime": "2020-01-08T18:33:59.265Z"
  }
}

```

- For API details, see [UpdateResolverEndpoint](#) in *AWS CLI Command Reference*.

update-resolver-rule

The following code example shows how to use `update-resolver-rule`.

AWS CLI

Example 1: To update settings Resolver endpoint

The following `update-resolver-rule` example updates the name of the rule, the IP addresses on your on-premises network that DNS queries are forwarded to, and the ID of the outbound Resolver endpoint that you're using to forward queries to your network.

Note Existing values for `TargetIps` are overwritten, so you must specify all the IP addresses that you want the rule to have after the update.

```

aws route53resolver update-resolver-rule \
  --resolver-rule-id rslvr-rr-1247fa64f3example \
  --config Name="my-2nd-rule",TargetIps=[{Ip=192.0.2.45,Port=53},
{Ip=192.0.2.46,Port=53}],ResolverEndpointId=rslvr-out-7b89ed0d25example

```

Output:

```

{
  "ResolverRule": {
    "Id": "rslvr-rr-1247fa64f3example",
    "CreatorRequestId": "2020-01-02-18:47",
    "Arn": "arn:aws:route53resolver:us-west-2:111122223333:resolver-rule/rslvr-rr-1247fa64f3example",
    "DomainName": "www.example.com.",
    "Status": "COMPLETE",
    "StatusMessage": "[Trace id: 1-5dcc90b9-8a8ee860aba1ebd89example]
Successfully updated Resolver Rule.",
    "RuleType": "FORWARD",

```



```
    "Name": "my-2nd-rule",
    "TargetIps": [
      {
        "Ip": "192.0.2.45",
        "Port": 53
      },
      {
        "Ip": "192.0.2.46",
        "Port": 53
      }
    ],
    "ResolverEndpointId": "rslvr-out-7b89ed0d25example",
    "OwnerId": "111122223333",
    "ShareStatus": "NOT_SHARED"
  }
}
```

Example 2: To update settings Resolver endpoint using a file for ``config`` settings

You can alternatively include the config settings in a JSON file and then specify that file when you call `update-resolver-rule`.

```
aws route53resolver update-resolver-rule \
  --resolver-rule-id rslvr-rr-1247fa64f3example \
  --config file://c:\temp\update-resolver-rule.json
```

Contents of `update-resolver-rule.json`.

```
{
  "Name": "my-2nd-rule",
  "TargetIps": [
    {
      "Ip": "192.0.2.45",
      "Port": 53
    },
    {
      "Ip": "192.0.2.46",
      "Port": 53
    }
  ],
  "ResolverEndpointId": "rslvr-out-7b89ed0d25example"
}
```

For more information, see [Values That You Specify When You Create or Edit Rules](#) in the *Amazon Route 53 Developer Guide*.

- For API details, see [UpdateResolverRule](#) in *AWS CLI Command Reference*.

Amazon S3 examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon S3.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

abort-multipart-upload

The following code example shows how to use `abort-multipart-upload`.

AWS CLI

To abort the specified multipart upload

The following `abort-multipart-upload` command aborts a multipart upload for the key `multipart/01` in the bucket `my-bucket`.

```
aws s3api abort-multipart-upload \  
  --bucket my-bucket \  
  --key multipart/01 \  
  --upload-id  
dfRtDYU0WWCCcH43C3WfbkR0NycyCpTJJvxu2i5GYkZ1jF.Yxwh6XG7WfS2vC4to6HiV6Yj1x.cph0gtNBtJ8P3URCS
```

The upload ID required by this command is output by `create-multipart-upload` and can also be retrieved with `list-multipart-uploads`.

- For API details, see [AbortMultipartUpload](#) in *AWS CLI Command Reference*.

complete-multipart-upload

The following code example shows how to use `complete-multipart-upload`.

AWS CLI

The following command completes a multipart upload for the key `multipart/01` in the bucket `my-bucket`:

```
aws s3api complete-multipart-upload --multipart-upload file://
mpustruct --bucket my-bucket --key 'multipart/01' --upload-id
dfRtDYU0WWCCcH43C3WfbkR0NycyCpTJJvxu2i5GYkZlJF.Yxwh6XG7WfS2vC4to6HiV6Yjlx.cph0gtNBtJ8P3URCS
```

The upload ID required by this command is output by `create-multipart-upload` and can also be retrieved with `list-multipart-uploads`.

The multipart upload option in the above command takes a JSON structure that describes the parts of the multipart upload that should be reassembled into the complete file. In this example, the `file://` prefix is used to load the JSON structure from a file in the local folder named `mpustruct`.

`mpustruct`:

```
{
  "Parts": [
    {
      "ETag": "e868e0f4719e394144ef36531ee6824c",
      "PartNumber": 1
    },
    {
      "ETag": "6bb2b12753d66fe86da4998aa33fffb0",
      "PartNumber": 2
    },
    {
      "ETag": "d0a0112e841abec9c9ec83406f0159c8",
      "PartNumber": 3
    }
  ]
}
```

```
]
}
```

The ETag value for each part is output each time you upload a part using the `upload-part` command and can also be retrieved by calling `list-parts` or calculated by taking the MD5 checksum of each part.

Output:

```
{
  "ETag": "\"3944a9f7a4faab7f78788ff6210f63f0-3\"",
  "Bucket": "my-bucket",
  "Location": "https://my-bucket.s3.amazonaws.com/multipart%2F01",
  "Key": "multipart/01"
}
```

- For API details, see [CompleteMultipartUpload](#) in *AWS CLI Command Reference*.

copy-object

The following code example shows how to use `copy-object`.

AWS CLI

The following command copies an object from `bucket-1` to `bucket-2`:

```
aws s3api copy-object --copy-source bucket-1/test.txt --key test.txt --bucket
bucket-2
```

Output:

```
{
  "CopyObjectResult": {
    "LastModified": "2015-11-10T01:07:25.000Z",
    "ETag": "\"589c8b79c230a6ecd5a7e1d040a9a030\""
  },
  "VersionId": "YdnYvTCVDqRRFA.NFJjy36p0hxifM1kA"
}
```

- For API details, see [CopyObject](#) in *AWS CLI Command Reference*.

cp

The following code example shows how to use cp.

AWS CLI

Example 1: Copying a local file to S3

The following cp command copies a single file to a specified bucket and key:

```
aws s3 cp test.txt s3://mybucket/test2.txt
```

Output:

```
upload: test.txt to s3://mybucket/test2.txt
```

Example 2: Copying a local file to S3 with an expiration date

The following cp command copies a single file to a specified bucket and key that expires at the specified ISO 8601 timestamp:

```
aws s3 cp test.txt s3://mybucket/test2.txt \  
--expires 2014-10-01T20:30:00Z
```

Output:

```
upload: test.txt to s3://mybucket/test2.txt
```

Example 3: Copying a file from S3 to S3

The following cp command copies a single s3 object to a specified bucket and key:

```
aws s3 cp s3://mybucket/test.txt s3://mybucket/test2.txt
```

Output:

```
copy: s3://mybucket/test.txt to s3://mybucket/test2.txt
```

Example 4: Copying an S3 object to a local file

The following cp command copies a single object to a specified file locally:

```
aws s3 cp s3://mybucket/test.txt test2.txt
```

Output:

```
download: s3://mybucket/test.txt to test2.txt
```

Example 5: Copying an S3 object from one bucket to another

The following `cp` command copies a single object to a specified bucket while retaining its original name:

```
aws s3 cp s3://mybucket/test.txt s3://mybucket2/
```

Output:

```
copy: s3://mybucket/test.txt to s3://mybucket2/test.txt
```

Example 6: Recursively copying S3 objects to a local directory

When passed with the parameter `--recursive`, the following `cp` command recursively copies all objects under a specified prefix and bucket to a specified directory. In this example, the bucket `mybucket` has the objects `test1.txt` and `test2.txt`:

```
aws s3 cp s3://mybucket . \  
--recursive
```

Output:

```
download: s3://mybucket/test1.txt to test1.txt  
download: s3://mybucket/test2.txt to test2.txt
```

Example 7: Recursively copying local files to S3

When passed with the parameter `--recursive`, the following `cp` command recursively copies all files under a specified directory to a specified bucket and prefix while excluding some files by using an `--exclude` parameter. In this example, the directory `myDir` has the files `test1.txt` and `test2.jpg`:

```
aws s3 cp myDir s3://mybucket/ \  
--recursive --exclude test2.jpg
```

```
--recursive \  
--exclude "*.jpg"
```

Output:

```
upload: myDir/test1.txt to s3://mybucket/test1.txt
```

Example 8: Recursively copying S3 objects to another bucket

When passed with the parameter `--recursive`, the following `cp` command recursively copies all objects under a specified bucket to another bucket while excluding some objects by using an `--exclude` parameter. In this example, the bucket `mybucket` has the objects `test1.txt` and `another/test1.txt`:

```
aws s3 cp s3://mybucket/ s3://mybucket2/ \  
--recursive \  
--exclude "another/*"
```

Output:

```
copy: s3://mybucket/test1.txt to s3://mybucket2/test1.txt
```

You can combine `--exclude` and `--include` options to copy only objects that match a pattern, excluding all others:

```
aws s3 cp s3://mybucket/logs/ s3://mybucket2/logs/ \  
--recursive \  
--exclude "*" \  
--include "*.log"
```

Output:

```
copy: s3://mybucket/logs/test/test.log to s3://mybucket2/logs/test/test.log  
copy: s3://mybucket/logs/test3.log to s3://mybucket2/logs/test3.log
```

Example 9: Setting the Access Control List (ACL) while copying an S3 object

The following `cp` command copies a single object to a specified bucket and key while setting the ACL to `public-read-write`:

```
aws s3 cp s3://mybucket/test.txt s3://mybucket/test2.txt \  
--acl public-read-write
```

Output:

```
copy: s3://mybucket/test.txt to s3://mybucket/test2.txt
```

Note that if you're using the `--acl` option, ensure that any associated IAM policies include the `"s3:PutObjectAcl"` action:

```
aws iam get-user-policy \  
--user-name myuser \  
--policy-name mypolicy
```

Output:

```
{  
  "UserName": "myuser",  
  "PolicyName": "mypolicy",  
  "PolicyDocument": {  
    "Version": "2012-10-17",  
    "Statement": [  
      {  
        "Action": [  
          "s3:PutObject",  
          "s3:PutObjectAcl"  
        ],  
        "Resource": [  
          "arn:aws:s3:::mybucket/*"  
        ],  
        "Effect": "Allow",  
        "Sid": "Stmt1234567891234"  
      }  
    ]  
  }  
}
```

Example 10: Granting permissions for an S3 object

The following `cp` command illustrates the use of the `--grants` option to grant read access to all users identified by URI and full control to a specific user identified by their Canonical ID:


```
aws s3 cp file.txt s3://mybucket/ --grants read=uri=http://acs.amazonaws.com/groups/global/AllUsers
full=id=79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
```

Output:

```
upload: file.txt to s3://mybucket/file.txt
```

Example 11: Uploading a local file stream to S3

PowerShell may alter the encoding of or add a CRLF to piped input.

The following `cp` command uploads a local file stream from standard input to a specified bucket and key:

```
aws s3 cp - s3://mybucket/stream.txt
```

Example 12: Uploading a local file stream that is larger than 50GB to S3

The following `cp` command uploads a 51GB local file stream from standard input to a specified bucket and key. The `--expected-size` option must be provided, or the upload may fail when it reaches the default part limit of 10,000:

```
aws s3 cp - s3://mybucket/stream.txt --expected-size 54760833024
```

Example 13: Downloading an S3 object as a local file stream

PowerShell may alter the encoding of or add a CRLF to piped or redirected output.

The following `cp` command downloads an S3 object locally as a stream to standard output. Downloading as a stream is not currently compatible with the `--recursive` parameter:

```
aws s3 cp s3://mybucket/stream.txt -
```

Example 14: Uploading to an S3 access point

The following `cp` command uploads a single file (`mydoc.txt`) to the access point (`myaccesspoint`) at the key (`mykey`):

```
aws s3 cp mydoc.txt s3://arn:aws:s3:us-west-2:123456789012:accesspoint/
myaccesspoint/mykey
```

Output:

```
upload: mydoc.txt to s3://arn:aws:s3:us-west-2:123456789012:accesspoint/
myaccesspoint/mykey
```

Example 15: Downloading from an S3 access point

The following `cp` command downloads a single object (`mykey`) from the access point (`myaccesspoint`) to the local file (`mydoc.txt`):

```
aws s3 cp s3://arn:aws:s3:us-west-2:123456789012:accesspoint/myaccesspoint/mykey
mydoc.txt
```

Output:

```
download: s3://arn:aws:s3:us-west-2:123456789012:accesspoint/myaccesspoint/mykey to
mydoc.txt
```

- For API details, see [Cp](#) in *AWS CLI Command Reference*.

create-bucket

The following code example shows how to use `create-bucket`.

AWS CLI**Example 1: To create a bucket**

The following `create-bucket` example creates a bucket named `my-bucket`:

```
aws s3api create-bucket \
  --bucket my-bucket \
  --region us-east-1
```

Output:

```
{
```

```
"Location": "/my-bucket"  
}
```

For more information, see [Creating a bucket](#) in the *Amazon S3 User Guide*.

Example 2: To create a bucket with owner enforced

The following `create-bucket` example creates a bucket named `my-bucket` that uses the bucket owner enforced setting for S3 Object Ownership.

```
aws s3api create-bucket \  
  --bucket my-bucket \  
  --region us-east-1 \  
  --object-ownership BucketOwnerEnforced
```

Output:

```
{  
  "Location": "/my-bucket"  
}
```

For more information, see [Controlling ownership of objects and disabling ACLs](#) in the *Amazon S3 User Guide*.

Example 3: To create a bucket outside of the ``us-east-1`` region

The following `create-bucket` example creates a bucket named `my-bucket` in the `eu-west-1` region. Regions outside of `us-east-1` require the appropriate `LocationConstraint` to be specified in order to create the bucket in the desired region.

```
aws s3api create-bucket \  
  --bucket my-bucket \  
  --region eu-west-1 \  
  --create-bucket-configuration LocationConstraint=eu-west-1
```

Output:

```
{  
  "Location": "http://my-bucket.s3.amazonaws.com/"  
}
```

For more information, see [Creating a bucket](#) in the *Amazon S3 User Guide*.

- For API details, see [CreateBucket](#) in *AWS CLI Command Reference*.

create-multipart-upload

The following code example shows how to use `create-multipart-upload`.

AWS CLI

The following command creates a multipart upload in the bucket `my-bucket` with the key `multipart/01`:

```
aws s3api create-multipart-upload --bucket my-bucket --key 'multipart/01'
```

Output:

```
{
  "Bucket": "my-bucket",
  "UploadId":
  "dfRtDYU0WWCCcH43C3WFbkR0NycyCpTJJvxu2i5GYkZ1jF.Yxwh6XG7WfS2vC4to6HiV6Yj1x.cph0gtNBtJ8P3URC
  "Key": "multipart/01"
}
```

The completed file will be named `01` in a folder called `multipart` in the bucket `my-bucket`. Save the upload ID, key and bucket name for use with the `upload-part` command.

- For API details, see [CreateMultipartUpload](#) in *AWS CLI Command Reference*.

delete-bucket-analytics-configuration

The following code example shows how to use `delete-bucket-analytics-configuration`.

AWS CLI

To delete an analytics configuration for a bucket

The following `delete-bucket-analytics-configuration` example removes the analytics configuration for the specified bucket and ID.

```
aws s3api delete-bucket-analytics-configuration \
  --bucket my-bucket \
```

```
--id 1
```

This command produces no output.

- For API details, see [DeleteBucketAnalyticsConfiguration](#) in *AWS CLI Command Reference*.

delete-bucket-cors

The following code example shows how to use `delete-bucket-cors`.

AWS CLI

The following command deletes a Cross-Origin Resource Sharing configuration from a bucket named `my-bucket`:

```
aws s3api delete-bucket-cors --bucket my-bucket
```

- For API details, see [DeleteBucketCors](#) in *AWS CLI Command Reference*.

delete-bucket-encryption

The following code example shows how to use `delete-bucket-encryption`.

AWS CLI

To delete the server-side encryption configuration of a bucket

The following `delete-bucket-encryption` example deletes the server-side encryption configuration of the specified bucket.

```
aws s3api delete-bucket-encryption \  
  --bucket my-bucket
```

This command produces no output.

- For API details, see [DeleteBucketEncryption](#) in *AWS CLI Command Reference*.

delete-bucket-intelligent-tiering-configuration

The following code example shows how to use `delete-bucket-intelligent-tiering-configuration`.

AWS CLI

To remove an S3 Intelligent-Tiering configuration on a bucket

The following `delete-bucket-intelligent-tiering-configuration` example removes an S3 Intelligent-Tiering configuration, named `ExampleConfig`, on a bucket.

```
aws s3api delete-bucket-intelligent-tiering-configuration \  
  --bucket DOC-EXAMPLE-BUCKET \  
  --id ExampleConfig
```

This command produces no output.

For more information, see [Using S3 Intelligent-Tiering](#) in the *Amazon S3 User Guide*.

- For API details, see [DeleteBucketIntelligentTieringConfiguration](#) in *AWS CLI Command Reference*.

`delete-bucket-inventory-configuration`

The following code example shows how to use `delete-bucket-inventory-configuration`.

AWS CLI

To delete the inventory configuration of a bucket

The following `delete-bucket-inventory-configuration` example deletes the inventory configuration with ID 1 for the specified bucket.

```
aws s3api delete-bucket-inventory-configuration \  
  --bucket my-bucket \  
  --id 1
```

This command produces no output.

- For API details, see [DeleteBucketInventoryConfiguration](#) in *AWS CLI Command Reference*.

`delete-bucket-lifecycle`

The following code example shows how to use `delete-bucket-lifecycle`.

AWS CLI

The following command deletes a lifecycle configuration from a bucket named my-bucket:

```
aws s3api delete-bucket-lifecycle --bucket my-bucket
```

- For API details, see [DeleteBucketLifecycle](#) in *AWS CLI Command Reference*.

delete-bucket-metrics-configuration

The following code example shows how to use delete-bucket-metrics-configuration.

AWS CLI

To delete a metrics configuration for a bucket

The following delete-bucket-metrics-configuration example removes the metrics configuration for the specified bucket and ID.

```
aws s3api delete-bucket-metrics-configuration \  
  --bucket my-bucket \  
  --id 123
```

This command produces no output.

- For API details, see [DeleteBucketMetricsConfiguration](#) in *AWS CLI Command Reference*.

delete-bucket-ownership-controls

The following code example shows how to use delete-bucket-ownership-controls.

AWS CLI

To remove the bucket ownership settings of a bucket

The following delete-bucket-ownership-controls example removes the bucket ownership settings of a bucket.

```
aws s3api delete-bucket-ownership-controls \  
  --bucket DOC-EXAMPLE-BUCKET
```

This command produces no output.

For more information, see [Setting Object Ownership on an existing bucket](#) in the *Amazon S3 User Guide*.

- For API details, see [DeleteBucketOwnershipControls](#) in *AWS CLI Command Reference*.

delete-bucket-policy

The following code example shows how to use `delete-bucket-policy`.

AWS CLI

The following command deletes a bucket policy from a bucket named `my-bucket`:

```
aws s3api delete-bucket-policy --bucket my-bucket
```

- For API details, see [DeleteBucketPolicy](#) in *AWS CLI Command Reference*.

delete-bucket-replication

The following code example shows how to use `delete-bucket-replication`.

AWS CLI

The following command deletes a replication configuration from a bucket named `my-bucket`:

```
aws s3api delete-bucket-replication --bucket my-bucket
```

- For API details, see [DeleteBucketReplication](#) in *AWS CLI Command Reference*.

delete-bucket-tagging

The following code example shows how to use `delete-bucket-tagging`.

AWS CLI

The following command deletes a tagging configuration from a bucket named `my-bucket`:

```
aws s3api delete-bucket-tagging --bucket my-bucket
```

- For API details, see [DeleteBucketTagging](#) in *AWS CLI Command Reference*.

delete-bucket-website

The following code example shows how to use `delete-bucket-website`.

AWS CLI

The following command deletes a website configuration from a bucket named `my-bucket`:

```
aws s3api delete-bucket-website --bucket my-bucket
```

- For API details, see [DeleteBucketWebsite](#) in *AWS CLI Command Reference*.

delete-bucket

The following code example shows how to use `delete-bucket`.

AWS CLI

The following command deletes a bucket named `my-bucket`:

```
aws s3api delete-bucket --bucket my-bucket --region us-east-1
```

- For API details, see [DeleteBucket](#) in *AWS CLI Command Reference*.

delete-object-tagging

The following code example shows how to use `delete-object-tagging`.

AWS CLI

To delete the tag sets of an object

The following `delete-object-tagging` example deletes the tag with the specified key from the object `doc1.rtf`.

```
aws s3api delete-object-tagging \  
  --bucket my-bucket \  
  --key doc1.rtf
```

This command produces no output.

- For API details, see [DeleteObjectTagging](#) in *AWS CLI Command Reference*.

delete-object

The following code example shows how to use `delete-object`.

AWS CLI

The following command deletes an object named `test.txt` from a bucket named `my-bucket`:

```
aws s3api delete-object --bucket my-bucket --key test.txt
```

If bucket versioning is enabled, the output will contain the version ID of the delete marker:

```
{
  "VersionId": "9_gKg5vG56F.TTEUdwkxGpJ3tND1W1Gq",
  "DeleteMarker": true
}
```

For more information about deleting objects, see *Deleting Objects in the Amazon S3 Developer Guide*.

- For API details, see [DeleteObject](#) in *AWS CLI Command Reference*.

delete-objects

The following code example shows how to use `delete-objects`.

AWS CLI

The following command deletes an object from a bucket named `my-bucket`:

```
aws s3api delete-objects --bucket my-bucket --delete file://delete.json
```

`delete.json` is a JSON document in the current directory that specifies the object to delete:

```
{
  "Objects": [
    {
      "Key": "test1.txt"
    }
  ],
  "Quiet": false
}
```

Output:

```
{
  "Deleted": [
    {
      "DeleteMarkerVersionId": "mYAT5Mc6F7aeUL8SS7FAAqUP01koHwzU",
      "Key": "test1.txt",
      "DeleteMarker": true
    }
  ]
}
```

- For API details, see [DeleteObjects](#) in *AWS CLI Command Reference*.

delete-public-access-block

The following code example shows how to use `delete-public-access-block`.

AWS CLI**To delete the block public access configuration for a bucket**

The following `delete-public-access-block` example removes the block public access configuration on the specified bucket.

```
aws s3api delete-public-access-block \
  --bucket my-bucket
```

This command produces no output.

- For API details, see [DeletePublicAccessBlock](#) in *AWS CLI Command Reference*.

get-bucket-accelerate-configuration

The following code example shows how to use `get-bucket-accelerate-configuration`.

AWS CLI**To retrieve the accelerate configuration of a bucket**

The following `get-bucket-accelerate-configuration` example retrieves the accelerate configuration for the specified bucket.

```
aws s3api get-bucket-accelerate-configuration \  
  --bucket my-bucket
```

Output:

```
{  
  "Status": "Enabled"  
}
```

- For API details, see [GetBucketAccelerateConfiguration](#) in *AWS CLI Command Reference*.

get-bucket-acl

The following code example shows how to use `get-bucket-acl`.

AWS CLI

The following command retrieves the access control list for a bucket named `my-bucket`:

```
aws s3api get-bucket-acl --bucket my-bucket
```

Output:

```
{  
  "Owner": {  
    "DisplayName": "my-username",  
    "ID": "7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"  
  },  
  "Grants": [  
    {  
      "Grantee": {  
        "DisplayName": "my-username",  
        "ID":  
"7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"  
      },  
      "Permission": "FULL_CONTROL"  
    }  
  ]  
}
```

- For API details, see [GetBucketAcl](#) in *AWS CLI Command Reference*.

get-bucket-analytics-configuration

The following code example shows how to use `get-bucket-analytics-configuration`.

AWS CLI

To retrieve the analytics configuration for a bucket with a specific ID

The following `get-bucket-analytics-configuration` example displays the analytics configuration for the specified bucket and ID.

```
aws s3api get-bucket-analytics-configuration \  
  --bucket my-bucket \  
  --id 1
```

Output:

```
{  
  "AnalyticsConfiguration": {  
    "StorageClassAnalysis": {},  
    "Id": "1"  
  }  
}
```

- For API details, see [GetBucketAnalyticsConfiguration](#) in *AWS CLI Command Reference*.

get-bucket-cors

The following code example shows how to use `get-bucket-cors`.

AWS CLI

The following command retrieves the Cross-Origin Resource Sharing configuration for a bucket named `my-bucket`:

```
aws s3api get-bucket-cors --bucket my-bucket
```

Output:

```
{  
  "CORSRules": [  
    {
```

```
    "AllowedHeaders": [
      "*"
    ],
    "ExposeHeaders": [
      "x-amz-server-side-encryption"
    ],
    "AllowedMethods": [
      "PUT",
      "POST",
      "DELETE"
    ],
    "MaxAgeSeconds": 3000,
    "AllowedOrigins": [
      "http://www.example.com"
    ]
  },
  {
    "AllowedHeaders": [
      "Authorization"
    ],
    "MaxAgeSeconds": 3000,
    "AllowedMethods": [
      "GET"
    ],
    "AllowedOrigins": [
      "*"
    ]
  }
]
```

- For API details, see [GetBucketCors](#) in *AWS CLI Command Reference*.

get-bucket-encryption

The following code example shows how to use `get-bucket-encryption`.

AWS CLI

To retrieve the server-side encryption configuration for a bucket

The following `get-bucket-encryption` example retrieves the server-side encryption configuration for the bucket `my-bucket`.

```
aws s3api get-bucket-encryption \  
  --bucket my-bucket
```

Output:

```
{  
  "ServerSideEncryptionConfiguration": {  
    "Rules": [  
      {  
        "ApplyServerSideEncryptionByDefault": {  
          "SSEAlgorithm": "AES256"  
        }  
      }  
    ]  
  }  
}
```

- For API details, see [GetBucketEncryption](#) in *AWS CLI Command Reference*.

get-bucket-intelligent-tiering-configuration

The following code example shows how to use `get-bucket-intelligent-tiering-configuration`.

AWS CLI**To retrieve an S3 Intelligent-Tiering configuration on a bucket**

The following `get-bucket-intelligent-tiering-configuration` example retrieves an S3 Intelligent-Tiering configuration, named `ExampleConfig`, on a bucket.

```
aws s3api get-bucket-intelligent-tiering-configuration \  
  --bucket DOC-EXAMPLE-BUCKET \  
  --id ExampleConfig
```

Output:

```
{  
  "IntelligentTieringConfiguration": {  
    "Id": "ExampleConfig2",
```

```
    "Filter": {
      "Prefix": "images"
    },
    "Status": "Enabled",
    "Tierings": [
      {
        "Days": 90,
        "AccessTier": "ARCHIVE_ACCESS"
      },
      {
        "Days": 180,
        "AccessTier": "DEEP_ARCHIVE_ACCESS"
      }
    ]
  }
}
```

For more information, see [Using S3 Intelligent-Tiering](#) in the *Amazon S3 User Guide*.

- For API details, see [GetBucketIntelligentTieringConfiguration](#) in *AWS CLI Command Reference*.

get-bucket-inventory-configuration

The following code example shows how to use `get-bucket-inventory-configuration`.

AWS CLI

To retrieve the inventory configuration for a bucket

The following `get-bucket-inventory-configuration` example retrieves the inventory configuration for the specified bucket with ID 1.

```
aws s3api get-bucket-inventory-configuration \
  --bucket my-bucket \
  --id 1
```

Output:

```
{
  "InventoryConfiguration": {
    "IsEnabled": true,
    "Destination": {
```



```
    "S3BucketDestination": {
      "Format": "ORC",
      "Bucket": "arn:aws:s3::my-bucket",
      "AccountId": "123456789012"
    },
    "IncludedObjectVersions": "Current",
    "Id": "1",
    "Schedule": {
      "Frequency": "Weekly"
    }
  }
}
```

- For API details, see [GetBucketInventoryConfiguration](#) in *AWS CLI Command Reference*.

get-bucket-lifecycle-configuration

The following code example shows how to use `get-bucket-lifecycle-configuration`.

AWS CLI

The following command retrieves the lifecycle configuration for a bucket named `my-bucket`:

```
aws s3api get-bucket-lifecycle-configuration --bucket my-bucket
```

Output:

```
{
  "Rules": [
    {
      "ID": "Move rotated logs to Glacier",
      "Prefix": "rotated/",
      "Status": "Enabled",
      "Transitions": [
        {
          "Date": "2015-11-10T00:00:00.000Z",
          "StorageClass": "GLACIER"
        }
      ]
    },
    {
```

```

        "Status": "Enabled",
        "Prefix": "",
        "NoncurrentVersionTransitions": [
            {
                "NoncurrentDays": 0,
                "StorageClass": "GLACIER"
            }
        ],
        "ID": "Move old versions to Glacier"
    }
]
}

```

- For API details, see [GetBucketLifecycleConfiguration](#) in *AWS CLI Command Reference*.

get-bucket-lifecycle

The following code example shows how to use `get-bucket-lifecycle`.

AWS CLI

The following command retrieves the lifecycle configuration for a bucket named `my-bucket`:

```
aws s3api get-bucket-lifecycle --bucket my-bucket
```

Output:

```

{
  "Rules": [
    {
      "ID": "Move to Glacier after sixty days (objects in logs/2015/)",
      "Prefix": "logs/2015/",
      "Status": "Enabled",
      "Transition": {
        "Days": 60,
        "StorageClass": "GLACIER"
      }
    },
    {
      "Expiration": {
        "Date": "2016-01-01T00:00:00.000Z"
      }
    }
  ],
}

```

```
    "ID": "Delete 2014 logs in 2016.",
    "Prefix": "logs/2014/",
    "Status": "Enabled"
  }
]
```

- For API details, see [GetBucketLifecycle](#) in *AWS CLI Command Reference*.

get-bucket-location

The following code example shows how to use `get-bucket-location`.

AWS CLI

The following command retrieves the location constraint for a bucket named `my-bucket`, if a constraint exists:

```
aws s3api get-bucket-location --bucket my-bucket
```

Output:

```
{
  "LocationConstraint": "us-west-2"
}
```

- For API details, see [GetBucketLocation](#) in *AWS CLI Command Reference*.

get-bucket-logging

The following code example shows how to use `get-bucket-logging`.

AWS CLI

To retrieve the logging status for a bucket

The following `get-bucket-logging` example retrieves the logging status for the specified bucket.

```
aws s3api get-bucket-logging \
```

```
--bucket my-bucket
```

Output:

```
{
  "LoggingEnabled": {
    "TargetPrefix": "",
    "TargetBucket": "my-bucket-logs"
  }
}
```

- For API details, see [GetBucketLogging](#) in *AWS CLI Command Reference*.

get-bucket-metrics-configuration

The following code example shows how to use `get-bucket-metrics-configuration`.

AWS CLI**To retrieve the metrics configuration for a bucket with a specific ID**

The following `get-bucket-metrics-configuration` example displays the metrics configuration for the specified bucket and ID.

```
aws s3api get-bucket-metrics-configuration \
  --bucket my-bucket \
  --id 123
```

Output:

```
{
  "MetricsConfiguration": {
    "Filter": {
      "Prefix": "logs"
    },
    "Id": "123"
  }
}
```

- For API details, see [GetBucketMetricsConfiguration](#) in *AWS CLI Command Reference*.

get-bucket-notification-configuration

The following code example shows how to use `get-bucket-notification-configuration`.

AWS CLI

The following command retrieves the notification configuration for a bucket named `my-bucket`:

```
aws s3api get-bucket-notification-configuration --bucket my-bucket
```

Output:

```
{
  "TopicConfigurations": [
    {
      "Id": "YmQzMmEwM2EjZWVlI0NGItNzVtZjI1MC00ZjgyLWZDBiZWw1",
      "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-notification-topic",
      "Events": [
        "s3:ObjectCreated:*"
      ]
    }
  ]
}
```

- For API details, see [GetBucketNotificationConfiguration](#) in *AWS CLI Command Reference*.

get-bucket-notification

The following code example shows how to use `get-bucket-notification`.

AWS CLI

The following command retrieves the notification configuration for a bucket named `my-bucket`:

```
aws s3api get-bucket-notification --bucket my-bucket
```

Output:

```
{
```

```
"TopicConfiguration": {
  "Topic": "arn:aws:sns:us-west-2:123456789012:my-notification-topic",
  "Id": "YmQzMmEwM2EjZWVlI0NGItNzVtZjI1MC00ZjgyLWZDBiZWw1",
  "Event": "s3:ObjectCreated:*",
  "Events": [
    "s3:ObjectCreated:*"
  ]
}
```

- For API details, see [GetBucketNotification](#) in *AWS CLI Command Reference*.

get-bucket-ownership-controls

The following code example shows how to use `get-bucket-ownership-controls`.

AWS CLI

To retrieve the bucket ownership settings of a bucket

The following `get-bucket-ownership-controls` example retrieves the bucket ownership settings of a bucket.

```
aws s3api get-bucket-ownership-controls \
  --bucket DOC-EXAMPLE-BUCKET
```

Output:

```
{
  "OwnershipControls": {
    "Rules": [
      {
        "ObjectOwnership": "BucketOwnerEnforced"
      }
    ]
  }
}
```

For more information, see [Viewing the Object Ownership setting for an S3 bucket](#) in the *Amazon S3 User Guide*.

- For API details, see [GetBucketOwnershipControls](#) in *AWS CLI Command Reference*.

get-bucket-policy-status

The following code example shows how to use `get-bucket-policy-status`.

AWS CLI

To retrieve the policy status for a bucket indicating whether the bucket is public

The following `get-bucket-policy-status` example retrieves the policy status for the bucket `my-bucket`.

```
aws s3api get-bucket-policy-status \
  --bucket my-bucket
```

Output:

```
{
  "PolicyStatus": {
    "IsPublic": false
  }
}
```

- For API details, see [GetBucketPolicyStatus](#) in *AWS CLI Command Reference*.

get-bucket-policy

The following code example shows how to use `get-bucket-policy`.

AWS CLI

The following command retrieves the bucket policy for a bucket named `my-bucket`:

```
aws s3api get-bucket-policy --bucket my-bucket
```

Output:

```
{
  "Policy": "{\n\"Version\": \"2008-10-17\", \"Statement\": [\n{\n\"Sid\": \"\", \"Effect\n\": \"Allow\", \"Principal\": \"*\", \"Action\": \"s3:GetObject\", \"Resource\":
```

```
\\"arn:aws:s3:::my-bucket/*\\"},{\\"Sid\\":\\"\\",\\"Effect\\":\\"Deny\\",\\"Principal\\":\\"*\\",
\\"Action\\":\\"s3:GetObject\\",\\"Resource\\":\\"arn:aws:s3:::my-bucket/secret/*\\"}]}"
}
```

Get and put a bucket policyThe following example shows how you can download an Amazon S3 bucket policy, make modifications to the file, and then use `put-bucket-policy` to apply the modified bucket policy. To download the bucket policy to a file, you can run:

```
aws s3api get-bucket-policy --bucket mybucket --query Policy --output text >
policy.json
```

You can then modify the `policy.json` file as needed. Finally you can apply this modified policy back to the S3 bucket by running:

`policy.json` file as needed. Finally you can apply this modified policy back to the S3 bucket by running:

file as needed. Finally you can apply this modified policy back to the S3 bucket by running:

```
aws s3api put-bucket-policy --bucket mybucket --policy file://policy.json
```

- For API details, see [GetBucketPolicy](#) in *AWS CLI Command Reference*.

get-bucket-replication

The following code example shows how to use `get-bucket-replication`.

AWS CLI

The following command retrieves the replication configuration for a bucket named `my-bucket`:

```
aws s3api get-bucket-replication --bucket my-bucket
```

Output:

```
{
  "ReplicationConfiguration": {
    "Rules": [
      {
```



```

        "Status": "Enabled",
        "Prefix": "",
        "Destination": {
            "Bucket": "arn:aws:s3:::my-bucket-backup",
            "StorageClass": "STANDARD"
        },
        "ID": "ZmUwNzE4ZmQ4tMjVhOS00MTlkLOGI4NDkzZTIWJjNTUtYTA1"
    },
    ],
    "Role": "arn:aws:iam::123456789012:role/s3-replication-role"
}

```

- For API details, see [GetBucketReplication](#) in *AWS CLI Command Reference*.

get-bucket-request-payment

The following code example shows how to use `get-bucket-request-payment`.

AWS CLI

To retrieve the request payment configuration for a bucket

The following `get-bucket-request-payment` example retrieves the requester pays configuration for the specified bucket.

```
aws s3api get-bucket-request-payment \
  --bucket my-bucket
```

Output:

```
{
  "Payer": "BucketOwner"
}
```

- For API details, see [GetBucketRequestPayment](#) in *AWS CLI Command Reference*.

get-bucket-tagging

The following code example shows how to use `get-bucket-tagging`.

AWS CLI

The following command retrieves the tagging configuration for a bucket named my-bucket:

```
aws s3api get-bucket-tagging --bucket my-bucket
```

Output:

```
{
  "TagSet": [
    {
      "Value": "marketing",
      "Key": "organization"
    }
  ]
}
```

- For API details, see [GetBucketTagging](#) in *AWS CLI Command Reference*.

get-bucket-versioning

The following code example shows how to use get-bucket-versioning.

AWS CLI

The following command retrieves the versioning configuration for a bucket named my-bucket:

```
aws s3api get-bucket-versioning --bucket my-bucket
```

Output:

```
{
  "Status": "Enabled"
}
```

- For API details, see [GetBucketVersioning](#) in *AWS CLI Command Reference*.

get-bucket-website

The following code example shows how to use get-bucket-website.

AWS CLI

The following command retrieves the static website configuration for a bucket named my-bucket:

```
aws s3api get-bucket-website --bucket my-bucket
```

Output:

```
{
  "IndexDocument": {
    "Suffix": "index.html"
  },
  "ErrorDocument": {
    "Key": "error.html"
  }
}
```

- For API details, see [GetBucketWebsite](#) in *AWS CLI Command Reference*.

get-object-acl

The following code example shows how to use `get-object-acl`.

AWS CLI

The following command retrieves the access control list for an object in a bucket named my-bucket:

```
aws s3api get-object-acl --bucket my-bucket --key index.html
```

Output:

```
{
  "Owner": {
    "DisplayName": "my-username",
    "ID": "7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"
  },
  "Grants": [
    {
      "Grantee": {
```

```

        "DisplayName": "my-username",
        "ID":
"7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"
    },
    "Permission": "FULL_CONTROL"
  },
  {
    "Grantee": {
      "URI": "http://acs.amazonaws.com/groups/global/AllUsers"
    },
    "Permission": "READ"
  }
]
}

```

- For API details, see [GetObjectAcl](#) in *AWS CLI Command Reference*.

get-object-attributes

The following code example shows how to use get-object-attributes.

AWS CLI

To retrieves metadata from an object without returning the object itself

The following get-object-attributes example retrieves metadata from the object doc1.rtf.

```

aws s3api get-object-attributes \
  --bucket my-bucket \
  --key doc1.rtf \
  --object-attributes "StorageClass" "ETag" "ObjectSize"

```

Output:

```

{
  "LastModified": "2022-03-15T19:37:31+00:00",
  "VersionId": "IuCPjXTDzHNf1dAuitVBIKJpF2p1fg4P",
  "ETag": "b662d79adeb7c8d787ea7eafb9ef6207",
  "StorageClass": "STANDARD",
  "ObjectSize": 405
}

```

For more information, see [GetObjectAttributes](#) in the Amazon S3 API Reference.

- For API details, see [GetObjectAttributes](#) in *AWS CLI Command Reference*.

get-object-legal-hold

The following code example shows how to use `get-object-legal-hold`.

AWS CLI

Retrieves the Legal Hold status of an object

The following `get-object-legal-hold` example retrieves the Legal Hold status for the specified object.

```
aws s3api get-object-legal-hold \  
  --bucket my-bucket-with-object-lock \  
  --key doc1.rtf
```

Output:

```
{  
  "LegalHold": {  
    "Status": "ON"  
  }  
}
```

- For API details, see [GetObjectLegalHold](#) in *AWS CLI Command Reference*.

get-object-lock-configuration

The following code example shows how to use `get-object-lock-configuration`.

AWS CLI

To retrieve an object lock configuration for a bucket

The following `get-object-lock-configuration` example retrieves the object lock configuration for the specified bucket.

```
aws s3api get-object-lock-configuration \  
  --bucket my-bucket-with-object-lock
```

Output:

```
{
  "ObjectLockConfiguration": {
    "ObjectLockEnabled": "Enabled",
    "Rule": {
      "DefaultRetention": {
        "Mode": "COMPLIANCE",
        "Days": 50
      }
    }
  }
}
```

- For API details, see [GetObjectLockConfiguration](#) in *AWS CLI Command Reference*.

get-object-retention

The following code example shows how to use `get-object-retention`.

AWS CLI**To retrieve the object retention configuration for an object**

The following `get-object-retention` example retrieves the object retention configuration for the specified object.

```
aws s3api get-object-retention \
  --bucket my-bucket-with-object-lock \
  --key doc1.rtf
```

Output:

```
{
  "Retention": {
    "Mode": "GOVERNANCE",
    "RetainUntilDate": "2025-01-01T00:00:00.000Z"
  }
}
```

- For API details, see [GetObjectRetention](#) in *AWS CLI Command Reference*.

get-object-tagging

The following code example shows how to use `get-object-tagging`.

AWS CLI

To retrieve the tags attached to an object

The following `get-object-tagging` example retrieves the values for the specified key from the specified object.

```
aws s3api get-object-tagging \  
  --bucket my-bucket \  
  --key doc1.rtf
```

Output:

```
{  
  "TagSet": [  
    {  
      "Value": "confidential",  
      "Key": "designation"  
    }  
  ]  
}
```

The following `get-object-tagging` example tries to retrieve the tag sets of the object `doc2.rtf`, which has no tags.

```
aws s3api get-object-tagging \  
  --bucket my-bucket \  
  --key doc2.rtf
```

Output:

```
{  
  "TagSet": []  
}
```

The following `get-object-tagging` example retrieves the tag sets of the object `doc3.rtf`, which has multiple tags.

```
aws s3api get-object-tagging \  
  --bucket my-bucket \  
  --key doc3.rtf
```

Output:

```
{  
  "TagSet": [  
    {  
      "Value": "confidential",  
      "Key": "designation"  
    },  
    {  
      "Value": "finance",  
      "Key": "department"  
    },  
    {  
      "Value": "payroll",  
      "Key": "team"  
    }  
  ]  
}
```

- For API details, see [GetObjectTagging](#) in *AWS CLI Command Reference*.

get-object-torrent

The following code example shows how to use `get-object-torrent`.

AWS CLI

The following command creates a torrent for an object in a bucket named `my-bucket`:

```
aws s3api get-object-torrent --bucket my-bucket --key large-video-file.mp4 large-  
video-file.torrent
```

The torrent file is saved locally in the current folder. Note that the output filename (`large-video-file.torrent`) is specified without an option name and must be the last argument in the command.

- For API details, see [GetObjectTorrent](#) in *AWS CLI Command Reference*.

get-object

The following code example shows how to use `get-object`.

AWS CLI

The following example uses the `get-object` command to download an object from Amazon S3:

```
aws s3api get-object --bucket text-content --key dir/my_images.tar.bz2
my_images.tar.bz2
```

Note that the outfile parameter is specified without an option name such as "`--outfile`". The name of the output file must be the last parameter in the command.

The example below demonstrates the use of `--range` to download a specific byte range from an object. Note the byte ranges needs to be prefixed with "`bytes=`":

```
aws s3api get-object --bucket text-content --key dir/my_data --range bytes=8888-9999
my_data_range
```

For more information about retrieving objects, see *Getting Objects in the Amazon S3 Developer Guide*.

- For API details, see [GetObject](#) in *AWS CLI Command Reference*.

get-public-access-block

The following code example shows how to use `get-public-access-block`.

AWS CLI

To set or modify the block public access configuration for a bucket

The following `get-public-access-block` example displays the block public access configuration for the specified bucket.

```
aws s3api get-public-access-block \
--bucket my-bucket
```

Output:

```
{
  "PublicAccessBlockConfiguration": {
    "IgnorePublicAcls": true,
    "BlockPublicPolicy": true,
    "BlockPublicAcls": true,
    "RestrictPublicBuckets": true
  }
}
```

- For API details, see [GetPublicAccessBlock](#) in *AWS CLI Command Reference*.

head-bucket

The following code example shows how to use head-bucket.

AWS CLI

The following command verifies access to a bucket named my-bucket:

```
aws s3api head-bucket --bucket my-bucket
```

If the bucket exists and you have access to it, no output is returned. Otherwise, an error message will be shown. For example:

```
A client error (404) occurred when calling the HeadBucket operation: Not Found
```

- For API details, see [HeadBucket](#) in *AWS CLI Command Reference*.

head-object

The following code example shows how to use head-object.

AWS CLI

The following command retrieves metadata for an object in a bucket named my-bucket:

```
aws s3api head-object --bucket my-bucket --key index.html
```

Output:

```
{
  "AcceptRanges": "bytes",
  "ContentType": "text/html",
  "LastModified": "Thu, 16 Apr 2015 18:19:14 GMT",
  "ContentLength": 77,
  "VersionId": "null",
  "ETag": "\"30a6ec7e1a9ad79c203d05a589c8b400\"",
  "Metadata": {}
}
```

- For API details, see [HeadObject](#) in *AWS CLI Command Reference*.

list-bucket-analytics-configurations

The following code example shows how to use `list-bucket-analytics-configurations`.

AWS CLI**To retrieve a list of analytics configurations for a bucket**

The following `list-bucket-analytics-configurations` retrieves a list of analytics configurations for the specified bucket.

```
aws s3api list-bucket-analytics-configurations \
  --bucket my-bucket
```

Output:

```
{
  "AnalyticsConfigurationList": [
    {
      "StorageClassAnalysis": {},
      "Id": "1"
    }
  ],
  "IsTruncated": false
}
```

- For API details, see [ListBucketAnalyticsConfigurations](#) in *AWS CLI Command Reference*.

list-bucket-intelligent-tiering-configurations

The following code example shows how to use `list-bucket-intelligent-tiering-configurations`.

AWS CLI

To retrieve all S3 Intelligent-Tiering configurations on a bucket

The following `list-bucket-intelligent-tiering-configurations` example retrieves all S3 Intelligent-Tiering configuration on a bucket.

```
aws s3api list-bucket-intelligent-tiering-configurations \
  --bucket DOC-EXAMPLE-BUCKET
```

Output:

```
{
  "IsTruncated": false,
  "IntelligentTieringConfigurationList": [
    {
      "Id": "ExampleConfig",
      "Filter": {
        "Prefix": "images"
      },
      "Status": "Enabled",
      "Tierings": [
        {
          "Days": 90,
          "AccessTier": "ARCHIVE_ACCESS"
        },
        {
          "Days": 180,
          "AccessTier": "DEEP_ARCHIVE_ACCESS"
        }
      ]
    },
    {
      "Id": "ExampleConfig2",
      "Status": "Disabled",
      "Tierings": [
        {
```

```
        "Days": 730,
        "AccessTier": "ARCHIVE_ACCESS"
      }
    ]
  },
  {
    "Id": "ExampleConfig3",
    "Filter": {
      "Tag": {
        "Key": "documents",
        "Value": "taxes"
      }
    },
    "Status": "Enabled",
    "Tierings": [
      {
        "Days": 90,
        "AccessTier": "ARCHIVE_ACCESS"
      },
      {
        "Days": 365,
        "AccessTier": "DEEP_ARCHIVE_ACCESS"
      }
    ]
  }
]
```

For more information, see [Using S3 Intelligent-Tiering](#) in the *Amazon S3 User Guide*.

- For API details, see [ListBucketIntelligentTieringConfigurations](#) in *AWS CLI Command Reference*.

list-bucket-inventory-configurations

The following code example shows how to use `list-bucket-inventory-configurations`.

AWS CLI

To retrieve a list of inventory configurations for a bucket

The following `list-bucket-inventory-configurations` example lists the inventory configurations for the specified bucket.

```
aws s3api list-bucket-inventory-configurations \  
  --bucket my-bucket
```

Output:

```
{  
  "InventoryConfigurationList": [  
    {  
      "IsEnabled": true,  
      "Destination": {  
        "S3BucketDestination": {  
          "Format": "ORC",  
          "Bucket": "arn:aws:s3:::my-bucket",  
          "AccountId": "123456789012"  
        }  
      },  
      "IncludedObjectVersions": "Current",  
      "Id": "1",  
      "Schedule": {  
        "Frequency": "Weekly"  
      }  
    },  
    {  
      "IsEnabled": true,  
      "Destination": {  
        "S3BucketDestination": {  
          "Format": "CSV",  
          "Bucket": "arn:aws:s3:::my-bucket",  
          "AccountId": "123456789012"  
        }  
      },  
      "IncludedObjectVersions": "Current",  
      "Id": "2",  
      "Schedule": {  
        "Frequency": "Daily"  
      }  
    }  
  ],  
  "IsTruncated": false  
}
```

- For API details, see [ListBucketInventoryConfigurations](#) in *AWS CLI Command Reference*.

list-bucket-metrics-configurations

The following code example shows how to use `list-bucket-metrics-configurations`.

AWS CLI

To retrieve a list of metrics configurations for a bucket

The following `list-bucket-metrics-configurations` example retrieves a list of metrics configurations for the specified bucket.

```
aws s3api list-bucket-metrics-configurations \  
  --bucket my-bucket
```

Output:

```
{  
  "IsTruncated": false,  
  "MetricsConfigurationList": [  
    {  
      "Filter": {  
        "Prefix": "logs"  
      },  
      "Id": "123"  
    },  
    {  
      "Filter": {  
        "Prefix": "tmp"  
      },  
      "Id": "234"  
    }  
  ]  
}
```

- For API details, see [ListBucketMetricsConfigurations](#) in *AWS CLI Command Reference*.

list-buckets

The following code example shows how to use `list-buckets`.

AWS CLI

The following command uses the `list-buckets` command to display the names of all your Amazon S3 buckets (across all regions):

```
aws s3api list-buckets --query "Buckets[].Name"
```

The query option filters the output of `list-buckets` down to only the bucket names.

For more information about buckets, see *Working with Amazon S3 Buckets in the Amazon S3 Developer Guide*.

- For API details, see [ListBuckets](#) in *AWS CLI Command Reference*.

list-multipart-uploads

The following code example shows how to use `list-multipart-uploads`.

AWS CLI

The following command lists all of the active multipart uploads for a bucket named `my-bucket`:

```
aws s3api list-multipart-uploads --bucket my-bucket
```

Output:

```
{
  "Uploads": [
    {
      "Initiator": {
        "DisplayName": "username",
        "ID": "arn:aws:iam::0123456789012:user/username"
      },
      "Initiated": "2015-06-02T18:01:30.000Z",
      "UploadId":
"dfRtDYU0WWCCcH43C3WFbkR0NycyCpTJJvxu2i5GYkZ1jF.Yxwh6XG7WfS2vC4to6HiV6Yj1x.cph0gtNBtJ8P3URC",
      "StorageClass": "STANDARD",
      "Key": "multipart/01",
      "Owner": {
        "DisplayName": "aws-account-name",
```



```

        "ID":
        "100719349fc3b6dcd7c820a124bf7aec408092c3d7b51b38494939801fc248b"
    }
    ],
    "CommonPrefixes": []
}

```

In progress multipart uploads incur storage costs in Amazon S3. Complete or abort an active multipart upload to remove its parts from your account.

- For API details, see [ListMultipartUploads](#) in *AWS CLI Command Reference*.

list-object-versions

The following code example shows how to use `list-object-versions`.

AWS CLI

The following command retrieves version information for an object in a bucket named `my-bucket`:

```
aws s3api list-object-versions --bucket my-bucket --prefix index.html
```

Output:

```

{
  "DeleteMarkers": [
    {
      "Owner": {
        "DisplayName": "my-username",
        "ID":
        "7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
      },
      "IsLatest": true,
      "VersionId": "B2VsEK5saUNNHKc0AJj7hIE86RozToyq",
      "Key": "index.html",
      "LastModified": "2015-11-10T00:57:03.000Z"
    },
    {
      "Owner": {
        "DisplayName": "my-username",

```

```

        "ID":
"7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
    },
    "IsLatest": false,
    "VersionId": ".FLQEZscLIcfxSq.jsFJ.szUkmng2Yw6",
    "Key": "index.html",
    "LastModified": "2015-11-09T23:32:20.000Z"
  }
],
"Versions": [
  {
    "LastModified": "2015-11-10T00:20:11.000Z",
    "VersionId": "Rb_l2T8UHDkFEwCgJjhlGPOZC0qJ.vpD",
    "ETag": "\"0622528de826c0df5db1258a23b80be5\"",
    "StorageClass": "STANDARD",
    "Key": "index.html",
    "Owner": {
      "DisplayName": "my-username",
      "ID":
"7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
    },
    "IsLatest": false,
    "Size": 38
  },
  {
    "LastModified": "2015-11-09T23:26:41.000Z",
    "VersionId": "rasWWGpgk9E4s0LyTJgusGeRQKLVIAff",
    "ETag": "\"06225825b8028de826c0df5db1a23be5\"",
    "StorageClass": "STANDARD",
    "Key": "index.html",
    "Owner": {
      "DisplayName": "my-username",
      "ID":
"7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
    },
    "IsLatest": false,
    "Size": 38
  },
  {
    "LastModified": "2015-11-09T22:50:50.000Z",
    "VersionId": "null",
    "ETag": "\"d1f45267a863c8392e07d24dd592f1b9\"",
    "StorageClass": "STANDARD",
    "Key": "index.html",

```

```
    "Owner": {
      "DisplayName": "my-username",
      "ID":
"7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
    },
    "IsLatest": false,
    "Size": 533823
  }
]
}
```

- For API details, see [ListObjectVersions](#) in *AWS CLI Command Reference*.

list-objects-v2

The following code example shows how to use `list-objects-v2`.

AWS CLI

To get a list of objects in a bucket

The following `list-objects-v2` example lists the objects in the specified bucket.

```
aws s3api list-objects-v2 \
  --bucket my-bucket
```

Output:

```
{
  "Contents": [
    {
      "LastModified": "2019-11-05T23:11:50.000Z",
      "ETag": "\"621503c373607d548b37cff8778d992c\"",
      "StorageClass": "STANDARD",
      "Key": "doc1.rtf",
      "Size": 391
    },
    {
      "LastModified": "2019-11-05T23:11:50.000Z",
      "ETag": "\"a2cecc36ab7c7fe3a71a273b9d45b1b5\"",
      "StorageClass": "STANDARD",
      "Key": "doc2.rtf",

```

```
    "Size": 373
  },
  {
    "LastModified": "2019-11-05T23:11:50.000Z",
    "ETag": "\"08210852f65a2e9cb999972539a64d68\"",
    "StorageClass": "STANDARD",
    "Key": "doc3.rtf",
    "Size": 399
  },
  {
    "LastModified": "2019-11-05T23:11:50.000Z",
    "ETag": "\"d1852dd683f404306569471af106988e\"",
    "StorageClass": "STANDARD",
    "Key": "doc4.rtf",
    "Size": 6225
  }
]
}
```

- For API details, see [ListObjectsV2](#) in *AWS CLI Command Reference*.

list-objects

The following code example shows how to use `list-objects`.

AWS CLI

The following example uses the `list-objects` command to display the names of all the objects in the specified bucket:

```
aws s3api list-objects --bucket text-content --query 'Contents[].{Key: Key, Size: Size}'
```

The example uses the `--query` argument to filter the output of `list-objects` down to the key value and size for each object

For more information about objects, see *Working with Amazon S3 Objects in the Amazon S3 Developer Guide*.

- For API details, see [ListObjects](#) in *AWS CLI Command Reference*.

list-parts

The following code example shows how to use `list-parts`.

AWS CLI

The following command lists all of the parts that have been uploaded for a multipart upload with key `multipart/01` in the bucket `my-bucket`:

```
aws s3api list-parts --bucket my-bucket --key 'multipart/01' --upload-id
dfRtDYU0WWCCcH43C3WFbkR0NycyCpTJJvxu2i5GYkZljF.Yxwh6XG7Wfs2vC4to6HiV6Yjlx.cph0gtNBtJ8P3URCS
```

Output:

```
{
  "Owner": {
    "DisplayName": "aws-account-name",
    "ID": "100719349fc3b6dcd7c820a124bf7aec408092c3d7b51b38494939801fc248b"
  },
  "Initiator": {
    "DisplayName": "username",
    "ID": "arn:aws:iam::0123456789012:user/username"
  },
  "Parts": [
    {
      "LastModified": "2015-06-02T18:07:35.000Z",
      "PartNumber": 1,
      "ETag": "\"e868e0f4719e394144ef36531ee6824c\"",
      "Size": 5242880
    },
    {
      "LastModified": "2015-06-02T18:07:42.000Z",
      "PartNumber": 2,
      "ETag": "\"6bb2b12753d66fe86da4998aa33fffb0\"",
      "Size": 5242880
    },
    {
      "LastModified": "2015-06-02T18:07:47.000Z",
      "PartNumber": 3,
      "ETag": "\"d0a0112e841abec9c9ec83406f0159c8\"",
      "Size": 5242880
    }
  ]
}
```

```
"StorageClass": "STANDARD"  
}
```

- For API details, see [ListParts](#) in *AWS CLI Command Reference*.

ls

The following code example shows how to use `ls`.

AWS CLI

Example 1: Listing all user owned buckets

The following `ls` command lists all of the bucket owned by the user. In this example, the user owns the buckets `mybucket` and `mybucket2`. The timestamp is the date the bucket was created, shown in your machine's time zone. This date can change when making changes to your bucket, such as editing its bucket policy. Note if `s3://` is used for the path argument `<S3Uri>`, it will list all of the buckets as well.

```
aws s3 ls
```

Output:

```
2013-07-11 17:08:50 mybucket  
2013-07-24 14:55:44 mybucket2
```

Example 2: Listing all prefixes and objects in a bucket

The following `ls` command lists objects and common prefixes under a specified bucket and prefix. In this example, the user owns the bucket `mybucket` with the objects `test.txt` and `somePrefix/test.txt`. The `LastWriteTime` and `Length` are arbitrary. Note that since the `ls` command has no interaction with the local filesystem, the `s3://` URI scheme is not required to resolve ambiguity and may be omitted.

```
aws s3 ls s3://mybucket
```

Output:

```
PRE somePrefix/
```

```
2013-07-25 17:06:27      88 test.txt
```

Example 3: Listing all prefixes and objects in a specific bucket and prefix

The following `ls` command lists objects and common prefixes under a specified bucket and prefix. However, there are no objects nor common prefixes under the specified bucket and prefix.

```
aws s3 ls s3://mybucket/noExistPrefix
```

Output:

```
None
```

Example 4: Recursively listing all prefixes and objects in a bucket

The following `ls` command will recursively list objects in a bucket. Rather than showing `PRE dirname/` in the output, all the content in a bucket will be listed in order.

```
aws s3 ls s3://mybucket \  
--recursive
```

Output:

```
2013-09-02 21:37:53      10 a.txt  
2013-09-02 21:37:53  2863288 foo.zip  
2013-09-02 21:32:57      23 foo/bar/.baz/a  
2013-09-02 21:32:58      41 foo/bar/.baz/b  
2013-09-02 21:32:57     281 foo/bar/.baz/c  
2013-09-02 21:32:57      73 foo/bar/.baz/d  
2013-09-02 21:32:57     452 foo/bar/.baz/e  
2013-09-02 21:32:57     896 foo/bar/.baz/hooks/bar  
2013-09-02 21:32:57     189 foo/bar/.baz/hooks/foo  
2013-09-02 21:32:57     398 z.txt
```

Example 5: Summarizing all prefixes and objects in a bucket

The following `ls` command demonstrates the same command using the `--human-readable` and `--summarize` options. `--human-readable` displays file size in Bytes/MiB/KiB/GiB/TiB/PiB/EiB. `--summarize` displays the total number of objects and total size at the end of the result listing:

```
aws s3 ls s3://mybucket \  
  --recursive \  
  --human-readable \  
  --summarize
```

Output:

```
2013-09-02 21:37:53  10 Bytes a.txt  
2013-09-02 21:37:53 2.9 MiB foo.zip  
2013-09-02 21:32:57  23 Bytes foo/bar/.baz/a  
2013-09-02 21:32:58  41 Bytes foo/bar/.baz/b  
2013-09-02 21:32:57 281 Bytes foo/bar/.baz/c  
2013-09-02 21:32:57  73 Bytes foo/bar/.baz/d  
2013-09-02 21:32:57 452 Bytes foo/bar/.baz/e  
2013-09-02 21:32:57 896 Bytes foo/bar/.baz/hooks/bar  
2013-09-02 21:32:57 189 Bytes foo/bar/.baz/hooks/foo  
2013-09-02 21:32:57 398 Bytes z.txt  
  
Total Objects: 10  
Total Size: 2.9 MiB
```

Example 6: Listing from an S3 access point

The following `ls` command list objects from access point (myaccesspoint):

```
aws s3 ls s3://arn:aws:s3:us-west-2:123456789012:accesspoint/myaccesspoint/
```

Output:

```
2013-07-25 17:06:27          PRE somePrefix/  
                   88 test.txt
```

- For API details, see [Ls](#) in *AWS CLI Command Reference*.

mb

The following code example shows how to use `mb`.

AWS CLI

Example 1: Create a bucket

The following `mb` command creates a bucket. In this example, the user makes the bucket `mybucket`. The bucket is created in the region specified in the user's configuration file:

```
aws s3 mb s3://mybucket
```

Output:

```
make_bucket: s3://mybucket
```

Example 2: Create a bucket in the specified region

The following `mb` command creates a bucket in a region specified by the `--region` parameter. In this example, the user makes the bucket `mybucket` in the region `us-west-1`:

```
aws s3 mb s3://mybucket \  
  --region us-west-1
```

Output:

```
make_bucket: s3://mybucket
```

- For API details, see [Mb](#) in *AWS CLI Command Reference*.

mv

The following code example shows how to use `mv`.

AWS CLI

Example 1: Move a local file to the specified bucket

The following `mv` command moves a single file to a specified bucket and key.

```
aws s3 mv test.txt s3://mybucket/test2.txt
```

Output:

```
move: test.txt to s3://mybucket/test2.txt
```

Example 2: Move an object to the specified bucket and key

The following `mv` command moves a single `s3` object to a specified bucket and key.

```
aws s3 mv s3://mybucket/test.txt s3://mybucket/test2.txt
```

Output:

```
move: s3://mybucket/test.txt to s3://mybucket/test2.txt
```

Example 3: Move an S3 object to the local directory

The following `mv` command moves a single object to a specified file locally.

```
aws s3 mv s3://mybucket/test.txt test2.txt
```

Output:

```
move: s3://mybucket/test.txt to test2.txt
```

Example 4: Move an object with its original name to the specified bucket

The following `mv` command moves a single object to a specified bucket while retaining its original name:

```
aws s3 mv s3://mybucket/test.txt s3://mybucket2/
```

Output:

```
move: s3://mybucket/test.txt to s3://mybucket2/test.txt
```

Example 5: Move all objects and prefixes in a bucket to the local directory

When passed with the parameter `--recursive`, the following `mv` command recursively moves all objects under a specified prefix and bucket to a specified directory. In this example, the bucket `mybucket` has the objects `test1.txt` and `test2.txt`.

```
aws s3 mv s3://mybucket . \
```

```
--recursive
```

Output:

```
move: s3://mybucket/test1.txt to test1.txt
move: s3://mybucket/test2.txt to test2.txt
```

Example 6: Move all objects and prefixes in a bucket to the local directory, except ``.jpg`` files

When passed with the parameter `--recursive`, the following `mv` command recursively moves all files under a specified directory to a specified bucket and prefix while excluding some files by using an `--exclude` parameter. In this example, the directory `myDir` has the files `test1.txt` and `test2.jpg`.

```
aws s3 mv myDir s3://mybucket/ \
  --recursive \
  --exclude "*.jpg"
```

Output:

```
move: myDir/test1.txt to s3://mybucket2/test1.txt
```

Example 7: Move all objects and prefixes in a bucket to the local directory, except specified prefix

When passed with the parameter `--recursive`, the following `mv` command recursively moves all objects under a specified bucket to another bucket while excluding some objects by using an `--exclude` parameter. In this example, the bucket `mybucket` has the objects `test1.txt` and `another/test1.txt`.

```
aws s3 mv s3://mybucket/ s3://mybucket2/ \
  --recursive \
  --exclude "mybucket/another/*"
```

Output:

```
move: s3://mybucket/test1.txt to s3://mybucket2/test1.txt
```

Example 8: Move an object to the specified bucket and set the ACL

The following `mv` command moves a single object to a specified bucket and key while setting the ACL to `public-read-write`.

```
aws s3 mv s3://mybucket/test.txt s3://mybucket/test2.txt \  
--acl public-read-write
```

Output:

```
move: s3://mybucket/test.txt to s3://mybucket/test2.txt
```

Example 9: Move a local file to the specified bucket and grant permissions

The following `mv` command illustrates the use of the `--grants` option to grant read access to all users and full control to a specific user identified by their email address.

```
aws s3 mv file.txt s3://mybucket/ \  
--grants read=uri=http://acs.amazonaws.com/groups/global/AllUsers  
full=emailaddress=user@example.com
```

Output:

```
move: file.txt to s3://mybucket/file.txt
```

Example 10: Move a file to an S3 access point

The following `mv` command moves a single file named `mydoc.txt` to the access point named `myaccesspoint` at the key named `mykey`.

```
aws s3 mv mydoc.txt s3://arn:aws:s3:us-west-2:123456789012:accesspoint/  
myaccesspoint/mykey
```

Output:

```
move: mydoc.txt to s3://arn:aws:s3:us-west-2:123456789012:accesspoint/myaccesspoint/  
mykey
```

- For API details, see [Mv](#) in *AWS CLI Command Reference*.

presign

The following code example shows how to use presign.

AWS CLI

Example 1: To create a pre-signed URL with the default one hour lifetime that links to an object in an S3 bucket

The following presign command generates a pre-signed URL for a specified bucket and key that is valid for one hour.

```
aws s3 presign s3://DOC-EXAMPLE-BUCKET/test2.txt
```

Output:

```
https://DOC-EXAMPLE-BUCKET.s3.us-west-2.amazonaws.com/key?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAEXAMPLE123456789%2F20210621%2Fus-west-2%2Fs3%2Faws4_request&X-Amz-Date=20210621T041609Z&X-Amz-Expires=3600&X-Amz-SignedHeaders=host&X-Amz-Signature=EXAMBLE1234494d5fba3fed607f98018e1dfc62e2529ae96d844123456
```

Example 2: To create a pre-signed URL with a custom lifetime that links to an object in an S3 bucket

The following presign command generates a pre-signed URL for a specified bucket and key that is valid for one week.

```
aws s3 presign s3://DOC-EXAMPLE-BUCKET/test2.txt \
  --expires-in 604800
```

Output:

```
https://DOC-EXAMPLE-BUCKET.s3.us-west-2.amazonaws.com/key?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAEXAMPLE123456789%2F20210621%2Fus-west-2%2Fs3%2Faws4_request&X-Amz-Date=20210621T041609Z&X-Amz-Expires=604800&X-Amz-SignedHeaders=host&X-Amz-Signature=EXAMBLE1234494d5fba3fed607f98018e1dfc62e2529ae96d844123456
```

For more information, see [Share an Object with Others](#) in the *S3 Developer Guide* guide.

- For API details, see [Presign](#) in *AWS CLI Command Reference*.

put-bucket-accelerate-configuration

The following code example shows how to use `put-bucket-accelerate-configuration`.

AWS CLI

To set the accelerate configuration of a bucket

The following `put-bucket-accelerate-configuration` example enables the accelerate configuration for the specified bucket.

```
aws s3api put-bucket-accelerate-configuration \  
  --bucket my-bucket \  
  --accelerate-configuration Status=Enabled
```

This command produces no output.

- For API details, see [PutBucketAccelerateConfiguration](#) in *AWS CLI Command Reference*.

put-bucket-acl

The following code example shows how to use `put-bucket-acl`.

AWS CLI

This example grants `full` control to two AWS users (`user1@example.com` and `user2@example.com`) and read permission to everyone:

```
aws s3api put-bucket-acl --bucket MyBucket --grant-full-control  
  emailaddress=user1@example.com,emailaddress=user2@example.com --grant-read  
  uri=http://acs.amazonaws.com/groups/global/AllUsers
```

See <http://docs.aws.amazon.com/AmazonS3/latest/API/RESTBucketPUTacl.html> for details on custom ACLs (the `s3api` ACL commands, such as `put-bucket-acl`, use the same shorthand argument notation).

- For API details, see [PutBucketAcl](#) in *AWS CLI Command Reference*.

put-bucket-analytics-configuration

The following code example shows how to use `put-bucket-analytics-configuration`.

AWS CLI

To sets an analytics configuration for the bucket

The following `put-bucket-analytics-configuration` example configures analytics for the specified bucket.

```
aws s3api put-bucket-analytics-configuration \  
  --bucket my-bucket --id 1 \  
  --analytics-configuration '{"Id": "1","StorageClassAnalysis": {}}'
```

This command produces no output.

- For API details, see [PutBucketAnalyticsConfiguration](#) in *AWS CLI Command Reference*.

put-bucket-cors

The following code example shows how to use `put-bucket-cors`.

AWS CLI

The following example enables PUT, POST, and DELETE requests from *www.example.com*, and enables GET requests from any domain:

```
aws s3api put-bucket-cors --bucket MyBucket --cors-configuration file://cors.json  
  
cors.json:  
{  
  "CORSRules": [  
    {  
      "AllowedOrigins": ["http://www.example.com"],  
      "AllowedHeaders": ["*"],  
      "AllowedMethods": ["PUT", "POST", "DELETE"],  
      "MaxAgeSeconds": 3000,  
      "ExposeHeaders": ["x-amz-server-side-encryption"]  
    },  
    {  
      "AllowedOrigins": ["*"],  
      "AllowedHeaders": ["Authorization"],  
      "AllowedMethods": ["GET"],  
      "MaxAgeSeconds": 3000  
    }  
  ]  
}
```

```
]
}
```

- For API details, see [PutBucketCors](#) in *AWS CLI Command Reference*.

put-bucket-encryption

The following code example shows how to use `put-bucket-encryption`.

AWS CLI

To configure server-side encryption for a bucket

The following `put-bucket-encryption` example sets AES256 encryption as the default for the specified bucket.

```
aws s3api put-bucket-encryption \
  --bucket my-bucket \
  --server-side-encryption-configuration '{"Rules":
  [{"ApplyServerSideEncryptionByDefault": {"SSEAlgorithm": "AES256"}}]}'
```

This command produces no output.

- For API details, see [PutBucketEncryption](#) in *AWS CLI Command Reference*.

put-bucket-intelligent-tiering-configuration

The following code example shows how to use `put-bucket-intelligent-tiering-configuration`.

AWS CLI

To update an S3 Intelligent-Tiering configuration on a bucket

The following `put-bucket-intelligent-tiering-configuration` example updates an S3 Intelligent-Tiering configuration, named `ExampleConfig`, on a bucket. The configuration will transition objects that have not been accessed under the prefix `images` to Archive Access after 90 days and Deep Archive Access after 180 days.

```
aws s3api put-bucket-intelligent-tiering-configuration \
```



```
--bucket DOC-EXAMPLE-BUCKET \  
--id "ExampleConfig" \  
--intelligent-tiering-configuration file://intelligent-tiering-  
configuration.json
```

Contents of `intelligent-tiering-configuration.json`:

```
{  
  "Id": "ExampleConfig",  
  "Status": "Enabled",  
  "Filter": {  
    "Prefix": "images"  
  },  
  "Tierings": [  
    {  
      "Days": 90,  
      "AccessTier": "ARCHIVE_ACCESS"  
    },  
    {  
      "Days": 180,  
      "AccessTier": "DEEP_ARCHIVE_ACCESS"  
    }  
  ]  
}
```

This command produces no output.

For more information, see [Setting Object Ownership on an existing bucket](#) in the *Amazon S3 User Guide*.

- For API details, see [PutBucketIntelligentTieringConfiguration](#) in *AWS CLI Command Reference*.

put-bucket-inventory-configuration

The following code example shows how to use `put-bucket-inventory-configuration`.

AWS CLI

Example 1: To set an inventory configuration for a bucket

The following `put-bucket-inventory-configuration` example sets a weekly ORC-formatted inventory report for the bucket `my-bucket`.

```
aws s3api put-bucket-inventory-configuration \  
  --bucket my-bucket \  
  --id 1 \  
  --inventory-configuration '{"Destination": { "S3BucketDestination":  
  { "AccountId": "123456789012", "Bucket": "arn:aws:s3:::my-bucket", "Format":  
  "ORC" }}, "IsEnabled": true, "Id": "1", "IncludedObjectVersions": "Current",  
  "Schedule": { "Frequency": "Weekly" } }'
```

This command produces no output.

Example 2: To set an inventory configuration for a bucket

The following `put-bucket-inventory-configuration` example sets a daily CSV-formatted inventory report for the bucket `my-bucket`.

```
aws s3api put-bucket-inventory-configuration \  
  --bucket my-bucket \  
  --id 2 \  
  --inventory-configuration '{"Destination": { "S3BucketDestination":  
  { "AccountId": "123456789012", "Bucket": "arn:aws:s3:::my-bucket", "Format":  
  "CSV" }}, "IsEnabled": true, "Id": "2", "IncludedObjectVersions": "Current",  
  "Schedule": { "Frequency": "Daily" } }'
```

This command produces no output.

- For API details, see [PutBucketInventoryConfiguration](#) in *AWS CLI Command Reference*.

put-bucket-lifecycle-configuration

The following code example shows how to use `put-bucket-lifecycle-configuration`.

AWS CLI

The following command applies a lifecycle configuration to a bucket named `my-bucket`:

```
aws s3api put-bucket-lifecycle-configuration --bucket my-bucket --lifecycle-  
configuration file://lifecycle.json
```

The file `lifecycle.json` is a JSON document in the current folder that specifies two rules:

```
{
```

```
"Rules": [
  {
    "ID": "Move rotated logs to Glacier",
    "Prefix": "rotated/",
    "Status": "Enabled",
    "Transitions": [
      {
        "Date": "2015-11-10T00:00:00.000Z",
        "StorageClass": "GLACIER"
      }
    ]
  },
  {
    "Status": "Enabled",
    "Prefix": "",
    "NoncurrentVersionTransitions": [
      {
        "NoncurrentDays": 2,
        "StorageClass": "GLACIER"
      }
    ],
    "ID": "Move old versions to Glacier"
  }
]
```

The first rule moves files with the prefix `rotated` to Glacier on the specified date. The second rule moves old object versions to Glacier when they are no longer current. For information on acceptable timestamp formats, see [Specifying Parameter Values in the AWS CLI User Guide](#).

- For API details, see [PutBucketLifecycleConfiguration](#) in *AWS CLI Command Reference*.

put-bucket-lifecycle

The following code example shows how to use `put-bucket-lifecycle`.

AWS CLI

The following command applies a lifecycle configuration to the bucket `my-bucket`:

```
aws s3api put-bucket-lifecycle --bucket my-bucket --lifecycle-configuration file://lifecycle.json
```

The file `lifecycle.json` is a JSON document in the current folder that specifies two rules:

```
{
  "Rules": [
    {
      "ID": "Move to Glacier after sixty days (objects in logs/2015/)",
      "Prefix": "logs/2015/",
      "Status": "Enabled",
      "Transition": {
        "Days": 60,
        "StorageClass": "GLACIER"
      }
    },
    {
      "Expiration": {
        "Date": "2016-01-01T00:00:00.000Z"
      },
      "ID": "Delete 2014 logs in 2016.",
      "Prefix": "logs/2014/",
      "Status": "Enabled"
    }
  ]
}
```

The first rule moves files to Amazon Glacier after sixty days. The second rule deletes files from Amazon S3 on the specified date. For information on acceptable timestamp formats, see *Specifying Parameter Values in the AWS CLI User Guide*.

Each rule in the above example specifies a policy (Transition or Expiration) and file prefix (folder name) to which it applies. You can also create a rule that applies to an entire bucket by specifying a blank prefix:

```
{
  "Rules": [
    {
      "ID": "Move to Glacier after sixty days (all objects in bucket)",
      "Prefix": "",
      "Status": "Enabled",
      "Transition": {
        "Days": 60,
        "StorageClass": "GLACIER"
      }
    }
  ]
}
```

```
    }  
  ]  
}
```

- For API details, see [PutBucketLifecycle](#) in *AWS CLI Command Reference*.

put-bucket-logging

The following code example shows how to use `put-bucket-logging`.

AWS CLI

Example 1: To set bucket policy logging

The following `put-bucket-logging` example sets the logging policy for *MyBucket*. First, grant the logging service principal permission in your bucket policy using the `put-bucket-policy` command.

```
aws s3api put-bucket-policy \  
  --bucket MyBucket \  
  --policy file://policy.json
```

Contents of `policy.json`:

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "S3ServerAccessLogsPolicy",  
      "Effect": "Allow",  
      "Principal": {"Service": "logging.s3.amazonaws.com"},  
      "Action": "s3:PutObject",  
      "Resource": "arn:aws:s3:::MyBucket/Logs/*",  
      "Condition": {  
        "ArnLike": {"aws:SourceARN": "arn:aws:s3:::SOURCE-BUCKET-NAME"},  
        "StringEquals": {"aws:SourceAccount": "SOURCE-AWS-ACCOUNT-ID"}  
      }  
    }  
  ]  
}
```

To apply the logging policy, use `put-bucket-logging`.

```
aws s3api put-bucket-logging \  
  --bucket MyBucket \  
  --bucket-logging-status file://logging.json
```

Contents of `logging.json`:

```
{  
  "LoggingEnabled": {  
    "TargetBucket": "MyBucket",  
    "TargetPrefix": "Logs/"  
  }  
}
```

The `put-bucket-policy` command is required to grant `s3:PutObject` permissions to the logging service principal.

For more information, see [Amazon S3 Server Access Logging](#) in the *Amazon S3 User Guide*.

Example 2: To set a bucket policy for logging access to only a single user

The following `put-bucket-logging` example sets the logging policy for *MyBucket*. The AWS user *bob@example.com* will have full control over the log files, and no one else has any access. First, grant S3 permission with `put-bucket-acl`.

```
aws s3api put-bucket-acl \  
  --bucket MyBucket \  
  --grant-write URI=http://acs.amazonaws.com/groups/s3/LogDelivery \  
  --grant-read-acp URI=http://acs.amazonaws.com/groups/s3/LogDelivery
```

Then apply the logging policy using `put-bucket-logging`.

```
aws s3api put-bucket-logging \  
  --bucket MyBucket \  
  --bucket-logging-status file://logging.json
```

Contents of `logging.json`:

```
{  
  "LoggingEnabled": {  
    "TargetBucket": "MyBucket",  
    "TargetPrefix": "MyBucketLogs/",  
  }  
}
```

```
    "TargetGrants": [
      {
        "Grantee": {
          "Type": "AmazonCustomerByEmail",
          "EmailAddress": "bob@example.com"
        },
        "Permission": "FULL_CONTROL"
      }
    ]
  }
}
```

the `put-bucket-acl` command is required to grant S3's log delivery system the necessary permissions (write and read-acp permissions).

For more information, see [Amazon S3 Server Access Logging](#) in the *Amazon S3 Developer Guide*.

- For API details, see [PutBucketLogging](#) in *AWS CLI Command Reference*.

put-bucket-metrics-configuration

The following code example shows how to use `put-bucket-metrics-configuration`.

AWS CLI

To set a metrics configuration for a bucket

The following `put-bucket-metrics-configuration` example sets a metric configuration with ID 123 for the specified bucket.

```
aws s3api put-bucket-metrics-configuration \
  --bucket my-bucket \
  --id 123 \
  --metrics-configuration '{"Id": "123", "Filter": {"Prefix": "logs"}}'
```

This command produces no output.

- For API details, see [PutBucketMetricsConfiguration](#) in *AWS CLI Command Reference*.

put-bucket-notification-configuration

The following code example shows how to use `put-bucket-notification-configuration`.

AWS CLI

To enable the specified notifications to a bucket

The following `put-bucket-notification-configuration` example applies a notification configuration to a bucket named `my-bucket`. The file `notification.json` is a JSON document in the current folder that specifies an SNS topic and an event type to monitor.

```
aws s3api put-bucket-notification-configuration \  
  --bucket my-bucket \  
  --notification-configuration file://notification.json
```

Contents of `notification.json`:

```
{  
  "TopicConfigurations": [  
    {  
      "TopicArn": "arn:aws:sns:us-west-2:123456789012:s3-notification-topic",  
      "Events": [  
        "s3:ObjectCreated:*"  
      ]  
    }  
  ]  
}
```

The SNS topic must have an IAM policy attached to it that allows Amazon S3 to publish to it.

```
{  
  "Version": "2008-10-17",  
  "Id": "example-ID",  
  "Statement": [  
    {  
      "Sid": "example-statement-ID",  
      "Effect": "Allow",  
      "Principal": {  
        "Service": "s3.amazonaws.com"  
      },  
      "Action": [  
        "SNS:Publish"  
      ],  
      "Resource": "arn:aws:sns:us-west-2:123456789012::s3-notification-topic",  
      "Condition": {
```



```

        "ArnLike": {
            "aws:SourceArn": "arn:aws:s3:*:*:my-bucket"
        }
    }
}
]
}

```

- For API details, see [PutBucketNotificationConfiguration](#) in *AWS CLI Command Reference*.

put-bucket-notification

The following code example shows how to use `put-bucket-notification`.

AWS CLI

The applies a notification configuration to a bucket named `my-bucket`:

```
aws s3api put-bucket-notification --bucket my-bucket --notification-configuration
file://notification.json
```

The file `notification.json` is a JSON document in the current folder that specifies an SNS topic and an event type to monitor:

```
{
  "TopicConfiguration": {
    "Event": "s3:ObjectCreated:*",
    "Topic": "arn:aws:sns:us-west-2:123456789012:s3-notification-topic"
  }
}
```

The SNS topic must have an IAM policy attached to it that allows Amazon S3 to publish to it:

```
{
  "Version": "2008-10-17",
  "Id": "example-ID",
  "Statement": [
    {
      "Sid": "example-statement-ID",
      "Effect": "Allow",
      "Principal": {
```

```
    "Service": "s3.amazonaws.com"
  },
  "Action": [
    "SNS:Publish"
  ],
  "Resource": "arn:aws:sns:us-west-2:123456789012:my-bucket",
  "Condition": {
    "ArnLike": {
      "aws:SourceArn": "arn:aws:s3:*:*:my-bucket"
    }
  }
}
```

- For API details, see [PutBucketNotification](#) in *AWS CLI Command Reference*.

put-bucket-ownership-controls

The following code example shows how to use `put-bucket-ownership-controls`.

AWS CLI

To update the bucket ownership settings of a bucket

The following `put-bucket-ownership-controls` example updates the bucket ownership settings of a bucket.

```
aws s3api put-bucket-ownership-controls \
  --bucket DOC-EXAMPLE-BUCKET \
  --ownership-controls="Rules=[{ObjectOwnership=BucketOwnerEnforced}]"
```

This command produces no output.

For more information, see [Setting Object Ownership on an existing bucket](#) in the *Amazon S3 User Guide*.

- For API details, see [PutBucketOwnershipControls](#) in *AWS CLI Command Reference*.

put-bucket-policy

The following code example shows how to use `put-bucket-policy`.

AWS CLI

This example allows all users to retrieve any object in *MyBucket* except those in the *MySecretFolder*. It also grants put and delete permission to the root user of the AWS account 1234-5678-9012:

```
aws s3api put-bucket-policy --bucket MyBucket --policy file://policy.json
```

```
policy.json:
```

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::MyBucket/*"
    },
    {
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::MyBucket/MySecretFolder/*"
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:root"
      },
      "Action": [
        "s3:DeleteObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::MyBucket/*"
    }
  ]
}
```

- For API details, see [PutBucketPolicy](#) in *AWS CLI Command Reference*.

put-bucket-replication

The following code example shows how to use put-bucket-replication.

AWS CLI

To configure replication for an S3 bucket

The following `put-bucket-replication` example applies a replication configuration to the specified S3 bucket.

```
aws s3api put-bucket-replication \  
  --bucket AWSDOC-EXAMPLE-BUCKET1 \  
  --replication-configuration file://replication.json
```

Contents of `replication.json`:

```
{  
  "Role": "arn:aws:iam::123456789012:role/s3-replication-role",  
  "Rules": [  
    {  
      "Status": "Enabled",  
      "Priority": 1,  
      "DeleteMarkerReplication": { "Status": "Disabled" },  
      "Filter" : { "Prefix": ""},  
      "Destination": {  
        "Bucket": "arn:aws:s3:::AWSDOC-EXAMPLE-BUCKET2"  
      }  
    }  
  ]  
}
```

The destination bucket must have versioning enabled. The specified role must have permission to write to the destination bucket and have a trust relationship that allows Amazon S3 to assume the role.

Example role permission policy:

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "s3:GetReplicationConfiguration",
```

```

        "s3:ListBucket"
    ],
    "Resource": [
        "arn:aws:s3:::AWSDOC-EXAMPLE-BUCKET1"
    ]
},
{
    "Effect": "Allow",
    "Action": [
        "s3:GetObjectVersion",
        "s3:GetObjectVersionAcl",
        "s3:GetObjectVersionTagging"
    ],
    "Resource": [
        "arn:aws:s3:::AWSDOC-EXAMPLE-BUCKET1/*"
    ]
},
{
    "Effect": "Allow",
    "Action": [
        "s3:ReplicateObject",
        "s3:ReplicateDelete",
        "s3:ReplicateTags"
    ],
    "Resource": "arn:aws:s3:::AWSDOC-EXAMPLE-BUCKET2/*"
}
]
}

```

Example trust relationship policy:

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}

```

This command produces no output.

For more information, see [This is the topic title](#) in the *Amazon Simple Storage Service Console User Guide*.

- For API details, see [PutBucketReplication](#) in *AWS CLI Command Reference*.

put-bucket-request-payment

The following code example shows how to use `put-bucket-request-payment`.

AWS CLI

Example 1: To enable `requester pays` configuration for a bucket

The following `put-bucket-request-payment` example enables `requester pays` for the specified bucket.

```
aws s3api put-bucket-request-payment \  
  --bucket my-bucket \  
  --request-payment-configuration '{"Payer":"Requester}"'
```

This command produces no output.

Example 2: To disable `requester pays` configuration for a bucket

The following `put-bucket-request-payment` example disables `requester pays` for the specified bucket.

```
aws s3api put-bucket-request-payment \  
  --bucket my-bucket \  
  --request-payment-configuration '{"Payer":"BucketOwner}"'
```

This command produces no output.

- For API details, see [PutBucketRequestPayment](#) in *AWS CLI Command Reference*.

put-bucket-tagging

The following code example shows how to use `put-bucket-tagging`.

AWS CLI

The following command applies a tagging configuration to a bucket named `my-bucket`:

```
aws s3api put-bucket-tagging --bucket my-bucket --tagging file://tagging.json
```

The file `tagging.json` is a JSON document in the current folder that specifies tags:

```
{
  "TagSet": [
    {
      "Key": "organization",
      "Value": "marketing"
    }
  ]
}
```

Or apply a tagging configuration to `my-bucket` directly from the command line:

```
aws s3api put-bucket-tagging --bucket my-bucket --tagging
'TagSet=[{Key=organization,Value=marketing}]'
```

- For API details, see [PutBucketTagging](#) in *AWS CLI Command Reference*.

put-bucket-versioning

The following code example shows how to use `put-bucket-versioning`.

AWS CLI

The following command enables versioning on a bucket named `my-bucket`:

```
aws s3api put-bucket-versioning --bucket my-bucket --versioning-configuration
Status=Enabled
```

The following command enables versioning, and uses an mfa code

```
aws s3api put-bucket-versioning --bucket my-bucket --versioning-configuration
Status=Enabled --mfa "SERIAL 123456"
```

- For API details, see [PutBucketVersioning](#) in *AWS CLI Command Reference*.

put-bucket-website

The following code example shows how to use `put-bucket-website`.

AWS CLI

The applies a static website configuration to a bucket named `my-bucket`:

```
aws s3api put-bucket-website --bucket my-bucket --website-configuration file://
website.json
```

The file `website.json` is a JSON document in the current folder that specifies index and error pages for the website:

```
{
  "IndexDocument": {
    "Suffix": "index.html"
  },
  "ErrorDocument": {
    "Key": "error.html"
  }
}
```

- For API details, see [PutBucketWebsite](#) in *AWS CLI Command Reference*.

put-object-acl

The following code example shows how to use `put-object-acl`.

AWS CLI

The following command grants `full control` to two AWS users (`user1@example.com` and `user2@example.com`) and `read` permission to everyone:

```
aws s3api put-object-acl --bucket MyBucket --key file.txt --grant-full-control
emailaddress=user1@example.com,emailaddress=user2@example.com --grant-read
uri=http://acs.amazonaws.com/groups/global/AllUsers
```


See <http://docs.aws.amazon.com/AmazonS3/latest/API/RESTBucketPUTacl.html> for details on custom ACLs (the s3api ACL commands, such as `put-object-acl`, use the same shorthand argument notation).

- For API details, see [PutObjectAcl](#) in *AWS CLI Command Reference*.

put-object-legal-hold

The following code example shows how to use `put-object-legal-hold`.

AWS CLI

To apply a Legal Hold to an object

The following `put-object-legal-hold` example sets a Legal Hold on the object `doc1.rtf`.

```
aws s3api put-object-legal-hold \  
  --bucket my-bucket-with-object-lock \  
  --key doc1.rtf \  
  --legal-hold Status=ON
```

This command produces no output.

- For API details, see [PutObjectLegalHold](#) in *AWS CLI Command Reference*.

put-object-lock-configuration

The following code example shows how to use `put-object-lock-configuration`.

AWS CLI

To set an object lock configuration on a bucket

The following `put-object-lock-configuration` example sets a 50-day object lock on the specified bucket.

```
aws s3api put-object-lock-configuration \  
  --bucket my-bucket-with-object-lock \  
  --object-lock-configuration '{ "ObjectLockEnabled": "Enabled", "Rule":  
  { "DefaultRetention": { "Mode": "COMPLIANCE", "Days": 50 } } }'
```

This command produces no output.

- For API details, see [PutObjectLockConfiguration](#) in *AWS CLI Command Reference*.

put-object-retention

The following code example shows how to use `put-object-retention`.

AWS CLI

To set an object retention configuration for an object

The following `put-object-retention` example sets an object retention configuration for the specified object until 2025-01-01.

```
aws s3api put-object-retention \  
  --bucket my-bucket-with-object-lock \  
  --key doc1.rtf \  
  --retention '{"Mode": "GOVERNANCE", "RetainUntilDate": "2025-01-01T00:00:00" }'
```

This command produces no output.

- For API details, see [PutObjectRetention](#) in *AWS CLI Command Reference*.

put-object-tagging

The following code example shows how to use `put-object-tagging`.

AWS CLI

To set a tag on an object

The following `put-object-tagging` example sets a tag with the key `designation` and the value `confidential` on the specified object.

```
aws s3api put-object-tagging \  
  --bucket my-bucket \  
  --key doc1.rtf \  
  --tagging '{"TagSet": [{ "Key": "designation", "Value": "confidential" }]}'
```

This command produces no output.

The following `put-object-tagging` example sets multiple tags sets on the specified object.

```
aws s3api put-object-tagging \  
  --bucket my-bucket-example \  
  --key doc3.rtf \  
  --tagging '{"TagSet": [{ "Key": "designation", "Value": "confidential" },  
  { "Key": "department", "Value": "finance" }, { "Key": "team", "Value":  
  "payroll" } ]}'
```

This command produces no output.

- For API details, see [PutObjectTagging](#) in *AWS CLI Command Reference*.

put-object

The following code example shows how to use `put-object`.

AWS CLI

The following example uses the `put-object` command to upload an object to Amazon S3:

```
aws s3api put-object --bucket text-content --key dir-1/my_images.tar.bz2 --body  
my_images.tar.bz2
```

The following example shows an upload of a video file (The video file is specified using Windows file system syntax.):

```
aws s3api put-object --bucket text-content --key dir-1/big-video-file.mp4 --body e:  
\media\videos\f-sharp-3-data-services.mp4
```

For more information about uploading objects, see *Uploading Objects in the Amazon S3 Developer Guide*.

- For API details, see [PutObject](#) in *AWS CLI Command Reference*.

put-public-access-block

The following code example shows how to use `put-public-access-block`.

AWS CLI

To set the block public access configuration for a bucket

The following `put-public-access-block` example sets a restrictive block public access configuration for the specified bucket.

```
aws s3api put-public-access-block \  
  --bucket my-bucket \  
  --public-access-block-configuration  
  "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets=true"
```

This command produces no output.

- For API details, see [PutPublicAccessBlock](#) in *AWS CLI Command Reference*.

rb

The following code example shows how to use `rb`.

AWS CLI

Example 1: Delete a bucket

The following `rb` command removes a bucket. In this example, the user's bucket is `mybucket`. Note that the bucket must be empty in order to remove:

```
aws s3 rb s3://mybucket
```

Output:

```
remove_bucket: mybucket
```

Example 2: Force delete a bucket

The following `rb` command uses the `--force` parameter to first remove all of the objects in the bucket and then remove the bucket itself. In this example, the user's bucket is `mybucket` and the objects in `mybucket` are `test1.txt` and `test2.txt`:

```
aws s3 rb s3://mybucket \  
  --force
```

Output:

```
delete: s3://mybucket/test1.txt
```

```
delete: s3://mybucket/test2.txt
remove_bucket: mybucket
```

- For API details, see [Rb](#) in *AWS CLI Command Reference*.

restore-object

The following code example shows how to use `restore-object`.

AWS CLI

To create a restore request for an object

The following `restore-object` example restores the specified Amazon S3 Glacier object for the bucket `my-glacier-bucket` for 10 days.

```
aws s3api restore-object \
  --bucket my-glacier-bucket \
  --key doc1.rtf \
  --restore-request Days=10
```

This command produces no output.

- For API details, see [RestoreObject](#) in *AWS CLI Command Reference*.

rm

The following code example shows how to use `rm`.

AWS CLI

Example 1: Delete an S3 object

The following `rm` command deletes a single S3 object:

```
aws s3 rm s3://mybucket/test2.txt
```

Output:

```
delete: s3://mybucket/test2.txt
```

Example 2: Delete all contents in a bucket

The following `rm` command recursively deletes all objects under a specified bucket and prefix when passed with the parameter `--recursive`. In this example, the bucket `mybucket` contains the objects `test1.txt` and `test2.txt`:

```
aws s3 rm s3://mybucket \  
  --recursive
```

Output:

```
delete: s3://mybucket/test1.txt  
delete: s3://mybucket/test2.txt
```

Example 3: Delete all contents in a bucket, except ``.jpg`` files

The following `rm` command recursively deletes all objects under a specified bucket and prefix when passed with the parameter `--recursive` while excluding some objects by using an `--exclude` parameter. In this example, the bucket `mybucket` has the objects `test1.txt` and `test2.jpg`:

```
aws s3 rm s3://mybucket/ \  
  --recursive \  
  --exclude "*.jpg"
```

Output:

```
delete: s3://mybucket/test1.txt
```

Example 4: Delete all contents in a bucket, except objects under the specified prefix

The following `rm` command recursively deletes all objects under a specified bucket and prefix when passed with the parameter `--recursive` while excluding all objects under a particular prefix by using an `--exclude` parameter. In this example, the bucket `mybucket` has the objects `test1.txt` and `another/test.txt`:

```
aws s3 rm s3://mybucket/ \  
  --recursive \  
  --exclude another/
```

```
--exclude "another/*"
```

Output:

```
delete: s3://mybucket/test1.txt
```

Example 5: Delete an object from an S3 access point

The following `rm` command deletes a single object (`mykey`) from the access point (`myaccesspoint`). :: The following `rm` command deletes a single object (`mykey`) from the access point (`myaccesspoint`).

```
aws s3 rm s3://arn:aws:s3:us-west-2:123456789012:accesspoint/myaccesspoint/mykey
```

Output:

```
delete: s3://arn:aws:s3:us-west-2:123456789012:accesspoint/myaccesspoint/mykey
```

- For API details, see [Rm](#) in *AWS CLI Command Reference*.

select-object-content

The following code example shows how to use `select-object-content`.

AWS CLI

To filter the contents of an Amazon S3 object based on an SQL statement

The following `select-object-content` example filters the object `my-data-file.csv` with the specified SQL statement and sends output to a file.

```
aws s3api select-object-content \  
  --bucket my-bucket \  
  --key my-data-file.csv \  
  --expression "select * from s3object limit 100" \  
  --expression-type 'SQL' \  
  --input-serialization '{"CSV": {}, "CompressionType": "NONE"}' \  
  --output-serialization '{"CSV": {}}' "output.csv"
```

This command produces no output.

- For API details, see [SelectObjectContent](#) in *AWS CLI Command Reference*.

sync

The following code example shows how to use sync.

AWS CLI

Example 1: Sync all local objects to the specified bucket

The following sync command syncs objects from a local directory to the specified prefix and bucket by uploading the local files to S3. A local file will require uploading if the size of the local file is different than the size of the S3 object, the last modified time of the local file is newer than the last modified time of the S3 object, or the local file does not exist under the specified bucket and prefix. In this example, the user syncs the bucket mybucket to the local current directory. The local current directory contains the files test.txt and test2.txt. The bucket mybucket contains no objects.

```
aws s3 sync . s3://mybucket
```

Output:

```
upload: test.txt to s3://mybucket/test.txt
upload: test2.txt to s3://mybucket/test2.txt
```

Example 2: Sync all S3 objects from the specified S3 bucket to another bucket

The following sync command syncs objects under a specified prefix and bucket to objects under another specified prefix and bucket by copying S3 objects. An S3 object will require copying if the sizes of the two S3 objects differ, the last modified time of the source is newer than the last modified time of the destination, or the S3 object does not exist under the specified bucket and prefix destination.

In this example, the user syncs the bucket mybucket to the bucket mybucket2. The bucket mybucket contains the objects test.txt and test2.txt. The bucket mybucket2 contains no objects:

```
aws s3 sync s3://mybucket s3://mybucket2
```


Output:

```
copy: s3://mybucket/test.txt to s3://mybucket2/test.txt
copy: s3://mybucket/test2.txt to s3://mybucket2/test2.txt
```

Example 3: Sync all S3 objects from the specified S3 bucket to the local directory

The following sync command syncs files from the specified S3 bucket to the local directory by downloading S3 objects. An S3 object will require downloading if the size of the S3 object differs from the size of the local file, the last modified time of the S3 object is newer than the last modified time of the local file, or the S3 object does not exist in the local directory. Take note that when objects are downloaded from S3, the last modified time of the local file is changed to the last modified time of the S3 object. In this example, the user syncs the bucket mybucket to the current local directory. The bucket mybucket contains the objects test.txt and test2.txt. The current local directory has no files:

```
aws s3 sync s3://mybucket .
```

Output:

```
download: s3://mybucket/test.txt to test.txt
download: s3://mybucket/test2.txt to test2.txt
```

Example 4: Sync all local objects to the specified bucket and delete all files that do not match

The following sync command syncs objects under a specified prefix and bucket to files in a local directory by uploading the local files to S3. Because of the --delete parameter, any files existing under the specified prefix and bucket but not existing in the local directory will be deleted. In this example, the user syncs the bucket mybucket to the local current directory. The local current directory contains the files test.txt and test2.txt. The bucket mybucket contains the object test3.txt:

```
aws s3 sync . s3://mybucket \
  --delete
```

Output:

```
upload: test.txt to s3://mybucket/test.txt
```

```
upload: test2.txt to s3://mybucket/test2.txt
delete: s3://mybucket/test3.txt
```

Example 5: Sync all local objects to the specified bucket except ``.jpg`` files

The following sync command syncs objects under a specified prefix and bucket to files in a local directory by uploading the local files to S3. Because of the `--exclude` parameter, all files matching the pattern existing both in S3 and locally will be excluded from the sync. In this example, the user syncs the bucket `mybucket` to the local current directory. The local current directory contains the files `test.jpg` and `test2.txt`. The bucket `mybucket` contains the object `test.jpg` of a different size than the local `test.jpg`:

```
aws s3 sync . s3://mybucket \
  --exclude "*.jpg"
```

Output:

```
upload: test2.txt to s3://mybucket/test2.txt
```

Example 6: Sync all local objects to the specified bucket except ``.jpg`` files

The following sync command syncs files under a local directory to objects under a specified prefix and bucket by downloading S3 objects. This example uses the `--exclude` parameter flag to exclude a specified directory and S3 prefix from the sync command. In this example, the user syncs the local current directory to the bucket `mybucket`. The local current directory contains the files `test.txt` and `another/test2.txt`. The bucket `mybucket` contains the objects `another/test5.txt` and `test1.txt`:

```
aws s3 sync s3://mybucket/ . \
  --exclude "*another/*"
```

Output:

```
download: s3://mybucket/test1.txt to test1.txt
```

Example 7: Sync all objects between buckets in different regions

The following sync command syncs files between two buckets in different regions:

```
aws s3 sync s3://my-us-west-2-bucket s3://my-us-east-1-bucket \  
  --source-region us-west-2 \  
  --region us-east-1
```

Output:

```
download: s3://my-us-west-2-bucket/test1.txt to s3://my-us-east-1-bucket/test1.txt
```

Example 8: Sync to an S3 access point

The following sync command syncs the current directory to the access point (myaccesspoint):

```
aws s3 sync . s3://arn:aws:s3:us-west-2:123456789012:accesspoint/myaccesspoint/
```

Output:

```
upload: test.txt to s3://arn:aws:s3:us-west-2:123456789012:accesspoint/  
myaccesspoint/test.txt  
upload: test2.txt to s3://arn:aws:s3:us-west-2:123456789012:accesspoint/  
myaccesspoint/test2.txt
```

- For API details, see [Sync](#) in *AWS CLI Command Reference*.

upload-part-copy

The following code example shows how to use upload-part-copy.

AWS CLI

To upload part of an object by copying data from an existing object as the data source

The following upload-part-copy example uploads a part by copying data from an existing object as a data source.

```
aws s3api upload-part-copy \  
  --bucket my-bucket \  
  --key "Map_Data_June.mp4" \  
  --copy-source "my-bucket/copy_of_Map_Data_June.mp4" \  
  --part-number 1 \  
  --
```

```
--upload-id
"bq0tdE1CDpWQYRPLHuNG50xAT6pA5D.m_RiBy0gg0H6b13pVRY7QjvL1f75iFdJqp_2wztk5hvpUM2SesXgrzbehG5"
```

Output:

```
{
  "CopyPartResult": {
    "LastModified": "2019-12-13T23:16:03.000Z",
    "ETag": "\"711470fc377698c393d94aed6305e245\""
  }
}
```

- For API details, see [UploadPartCopy](#) in *AWS CLI Command Reference*.

upload-part

The following code example shows how to use `upload-part`.

AWS CLI

The following command uploads the first part in a multipart upload initiated with the `create-multipart-upload` command:

```
aws s3api upload-part --bucket my-bucket --key 'multipart/01' --part-number 1 --body
part01 --upload-id
"dfRtDYU0WWCCcH43C3WFbkR0NycyCpTJJvxu2i5GYkZ1jF.Yxwh6XG7WfS2vC4to6HiV6Yj1x.cph0gtNBtJ8P3URC"
```

The `body` option takes the name or path of a local file for upload (do not use the `file://` prefix). The minimum part size is 5 MB. Upload ID is returned by `create-multipart-upload` and can also be retrieved with `list-multipart-uploads`. Bucket and key are specified when you create the multipart upload.

Output:

```
{
  "ETag": "\"e868e0f4719e394144ef36531ee6824c\""
}
```

Save the ETag value of each part for later. They are required to complete the multipart upload.

- For API details, see [UploadPart](#) in *AWS CLI Command Reference*.

website

The following code example shows how to use `website`.

AWS CLI

Configure an S3 bucket as a static website

The following command configures a bucket named `my-bucket` as a static website. The `index document` option specifies the file in `my-bucket` that visitors will be directed to when they navigate to the website URL. In this case, the bucket is in the `us-west-2` region, so the site would appear at `http://my-bucket.s3-website-us-west-2.amazonaws.com`.

All files in the bucket that appear on the static site must be configured to allow visitors to open them. File permissions are configured separately from the bucket website configuration.

```
aws s3 website s3://my-bucket/ \  
  --index-document index.html \  
  --error-document error.html
```

For information on hosting a static website in Amazon S3, see [Hosting a Static Website](#) in the *Amazon Simple Storage Service Developer Guide*.

- For API details, see [Website](#) in *AWS CLI Command Reference*.

Amazon S3 Control examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon S3 Control.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-access-point

The following code example shows how to use create-access-point.

AWS CLI

To create an access point

The following create-access-point example creates an access point named finance-ap for the bucket business-records in account 123456789012. Before running this example, replace the access point name, bucket name, and account number with appropriate values for your use case.

```
aws s3control create-access-point \  
  --account-id 123456789012 \  
  --bucket business-records \  
  --name finance-ap
```

This command produces no output.

For more information, see [Creating Access Points](#) in the *Amazon Simple Storage Service Developer Guide*.

- For API details, see [CreateAccessPoint](#) in *AWS CLI Command Reference*.

create-job

The following code example shows how to use create-job.

AWS CLI

To create an Amazon S3 batch operations job

The following create-job example creates an Amazon S3 batch operations job to tag objects as confidential in the bucket employee-records.

```
aws s3control create-job \  
  --account-id 123456789012 \  
  --bucket employee-records \  
  --name my-job
```

```
--account-id 123456789012 \  
--operation '{"S3PutObjectTagging": { "TagSet": [{"Key":"confidential",  
"Value":"true"}] }}' \  
--report '{"Bucket":"arn:aws:s3:::employee-records-logs","Prefix":"batch-op-  
create-job",  
"Format":"Report_CSV_20180820","Enabled":true,"ReportScope":"AllTasks"}' \  
--manifest '{"Spec":{"Format":"S3BatchOperations_CSV_20180820","Fields":  
["Bucket","Key"]},"Location":{"ObjectArn":"arn:aws:s3:::employee-records-logs/inv-  
report/7a6a9be4-072c-407e-85a2-  
ec3e982f773e.csv","ETag":"69f52a4e9f797e987155d9c8f5880897"}}' \  
--priority 42 \  
--role-arn arn:aws:iam::123456789012:role/S3BatchJobRole
```

Output:

```
{  
  "JobId": "93735294-df46-44d5-8638-6356f335324e"  
}
```

- For API details, see [CreateJob](#) in *AWS CLI Command Reference*.

delete-access-point-policy

The following code example shows how to use `delete-access-point-policy`.

AWS CLI

To delete an access point policy

The following `delete-access-point-policy` example deletes the access point policy from the access point named `finance-ap` in account `123456789012`. Before running this example, replace the access point name and account number with appropriate values for your use case.

```
aws s3control delete-access-point-policy \  
--account-id 123456789012 \  
--name finance-ap
```

This command produces no output.

For more information, see [Managing Data Access with Amazon S3 Access Points](#) in the *Amazon Simple Storage Service Developer Guide*.

- For API details, see [DeleteAccessPointPolicy](#) in *AWS CLI Command Reference*.

delete-access-point

The following code example shows how to use `delete-access-point`.

AWS CLI

To delete an access point

The following `delete-access-point` example deletes an access point named `finance-ap` in account `123456789012`. Before running this example, replace the access point name and account number with appropriate values for your use case.

```
aws s3control delete-access-point \  
  --account-id 123456789012 \  
  --name finance-ap
```

This command produces no output.

For more information, see [Managing Data Access with Amazon S3 Access Points](#) in the *Amazon Simple Storage Service Developer Guide*.

- For API details, see [DeleteAccessPoint](#) in *AWS CLI Command Reference*.

delete-public-access-block

The following code example shows how to use `delete-public-access-block`.

AWS CLI

To delete block public access settings for an account

The following `delete-public-access-block` example deletes block public access settings for the specified account.

```
aws s3control delete-public-access-block \  
  --account-id 123456789012
```

This command produces no output.

- For API details, see [DeletePublicAccessBlock](#) in *AWS CLI Command Reference*.

describe-job

The following code example shows how to use `describe-job`.

AWS CLI

To describe an Amazon S3 batch operations job

The following `describe-job` provides configuration parameters and status for the specified batch operations job.

```
aws s3control describe-job \
  --account-id 123456789012 \
  --job-id 93735294-df46-44d5-8638-6356f335324e
```

Output:

```
{
  "Job": {
    "TerminationDate": "2019-10-03T21:49:53.944Z",
    "JobId": "93735294-df46-44d5-8638-6356f335324e",
    "FailureReasons": [],
    "Manifest": {
      "Spec": {
        "Fields": [
          "Bucket",
          "Key"
        ],
        "Format": "S3BatchOperations_CSV_20180820"
      },
      "Location": {
        "ETag": "69f52a4e9f797e987155d9c8f5880897",
        "ObjectArn": "arn:aws:s3:::employee-records-logs/inv-report/7a6a9be4-072c-407e-85a2-ec3e982f773e.csv"
      }
    },
    "Operation": {
      "S3PutObjectTagging": {
        "TagSet": [
          {
```

```

        "Value": "true",
        "Key": "confidential"
      }
    ]
  }
},
"RoleArn": "arn:aws:iam::123456789012:role/S3BatchJobRole",
"ProgressSummary": {
  "TotalNumberOfTasks": 8,
  "NumberOfTasksFailed": 0,
  "NumberOfTasksSucceeded": 8
},
"Priority": 42,
"Report": {
  "ReportScope": "AllTasks",
  "Format": "Report_CSV_20180820",
  "Enabled": true,
  "Prefix": "batch-op-create-job",
  "Bucket": "arn:aws:s3:::employee-records-logs"
},
"JobArn": "arn:aws:s3:us-west-2:123456789012:job/93735294-
df46-44d5-8638-6356f335324e",
"CreationTime": "2019-10-03T21:48:48.048Z",
"Status": "Complete"
}
}

```

- For API details, see [DescribeJob](#) in *AWS CLI Command Reference*.

get-access-point-policy-status

The following code example shows how to use `get-access-point-policy-status`.

AWS CLI

To retrieve the access point policy status

The following `get-access-point-policy-status` example retrieves the access point policy status for the access point named `finance-ap` in account `123456789012`. The access point policy status indicates whether the access point's policy allows public access. Before running this example, replace the access point name and account number with appropriate values for your use case.

```
aws s3control get-access-point-policy-status \  
  --account-id 123456789012 \  
  --name finance-ap
```

Output:

```
{  
  "PolicyStatus": {  
    "IsPublic": false  
  }  
}
```

For more information about when an access point policy is considered public, see [The Meaning of "Public"](#) in the *Amazon Simple Storage Service Developer Guide*.

- For API details, see [GetAccessPointPolicyStatus](#) in *AWS CLI Command Reference*.

get-access-point-policy

The following code example shows how to use `get-access-point-policy`.

AWS CLI

To retrieve an access point policy

The following `get-access-point-policy` example retrieves the access point policy from the access point named `finance-ap` in account `123456789012`. Before running this example, replace the access point name and account number with appropriate values for your use case.

```
aws s3control get-access-point-policy \  
  --account-id 123456789012 \  
  --name finance-ap
```

Output:

```
{  
  "Policy": "{\"Version\":\"2012-10-17\", \"Statement\": [{\"Effect\":\"Allow\", \"Principal\": {\"AWS\": \"arn:aws:iam:123456789012:role/Admin\"}, \"Action\": \"s3:GetObject\", \"Resource\": \"arn:aws:s3:us-west-2:123456789012:accesspoint/finance-ap/object/records/*\"}]}"
```

```
}
```

For more information, see [Managing Data Access with Amazon S3 Access Points](#) in the *Amazon Simple Storage Service Developer Guide*.

- For API details, see [GetAccessPointPolicy](#) in *AWS CLI Command Reference*.

get-access-point

The following code example shows how to use `get-access-point`.

AWS CLI

To retrieve access point configuration details

The following `get-access-point` example retrieves the configuration details for the access point named `finance-ap` in account `123456789012`. Before running this example, replace the access point name and account number with appropriate values for your use case.

```
aws s3control get-access-point \  
  --account-id 123456789012 \  
  --name finance-ap
```

Output:

```
{  
  "Name": "finance-ap",  
  "Bucket": "business-records",  
  "NetworkOrigin": "Internet",  
  "PublicAccessBlockConfiguration": {  
    "BlockPublicAcls": false,  
    "IgnorePublicAcls": false,  
    "BlockPublicPolicy": false,  
    "RestrictPublicBuckets": false  
  },  
  "CreationDate": "2020-01-01T00:00:00Z"  
}
```

For more information, see [Managing Data Access with Amazon S3 Access Points](#) in the *Amazon Simple Storage Service Developer Guide*.

- For API details, see [GetAccessPoint](#) in *AWS CLI Command Reference*.

get-multi-region-access-point-routes

The following code example shows how to use `get-multi-region-access-point-routes`.

AWS CLI

To query the current Multi-Region Access Point route configuration

The following `get-multi-region-access-point-routes` example returns the current routing configuration for the specified Multi-Region Access Point.

```
aws s3control get-multi-region-access-point-routes \
  --region Region \
  --account-id 111122223333 \
  --mrap MultiRegionAccessPoint_ARN
```

Output:

```
{
  "Mrap": "arn:aws:s3::111122223333:accesspoint/0000000000000000.mrap",
  "Routes": [
    {
      "Bucket": "DOC-EXAMPLE-BUCKET-1",
      "Region": "ap-southeast-2",
      "TrafficDialPercentage": 100
    },
    {
      "Bucket": "DOC-EXAMPLE-BUCKET-2",
      "Region": "us-west-1",
      "TrafficDialPercentage": 0
    }
  ]
}
```

- For API details, see [GetMultiRegionAccessPointRoutes](#) in *AWS CLI Command Reference*.

get-public-access-block

The following code example shows how to use `get-public-access-block`.

AWS CLI

To list public block access settings for an account

The following `get-public-access-block` example displays the block public access settings for the specified account.

```
aws s3control get-public-access-block \  
  --account-id 123456789012
```

Output:

```
{  
  "PublicAccessBlockConfiguration": {  
    "BlockPublicPolicy": true,  
    "RestrictPublicBuckets": true,  
    "IgnorePublicAcls": true,  
    "BlockPublicAcls": true  
  }  
}
```

- For API details, see [GetPublicAccessBlock](#) in *AWS CLI Command Reference*.

list-access-points

The following code example shows how to use `list-access-points`.

AWS CLI

Example 1: To retrieve a list of all access points for an account

The following `list-access-points` example displays a list of all access points attached to buckets owned by account 123456789012.

```
aws s3control list-access-points \  
  --account-id 123456789012
```

Output:

```
{  
  "AccessPointList": [  
    {  
      "Name": "finance-ap",  
      "NetworkOrigin": "Internet",
```

```

    "Bucket": "business-records"
  },
  {
    "Name": "managers-ap",
    "NetworkOrigin": "Internet",
    "Bucket": "business-records"
  },
  {
    "Name": "private-network-ap",
    "NetworkOrigin": "VPC",
    "VpcConfiguration": {
      "VpcId": "1a2b3c"
    },
    "Bucket": "business-records"
  },
  {
    "Name": "customer-ap",
    "NetworkOrigin": "Internet",
    "Bucket": "external-docs"
  },
  {
    "Name": "public-ap",
    "NetworkOrigin": "Internet",
    "Bucket": "external-docs"
  }
]
}

```

Example 2: To retrieve a list of all access points for a bucket

The following `list-access-points` example retrieves a list of all access points attached to the bucket `external-docs` owned by account `123456789012`.

```

aws s3control list-access-points \
  --account-id 123456789012 \
  --bucket external-docs

```

Output:

```

{
  "AccessPointList": [
    {
      "Name": "customer-ap",

```

```

        "NetworkOrigin": "Internet",
        "Bucket": "external-docs"
    },
    {
        "Name": "public-ap",
        "NetworkOrigin": "Internet",
        "Bucket": "external-docs"
    }
]
}

```

For more information, see [Managing Data Access with Amazon S3 Access Points](#) in the *Amazon Simple Storage Service Developer Guide*.

- For API details, see [ListAccessPoints](#) in *AWS CLI Command Reference*.

list-jobs

The following code example shows how to use `list-jobs`.

AWS CLI

To list an accounts Amazon S3 batch operations jobs

The following `list-jobs` example lists all recent batch operations jobs for the specified account.

```

aws s3control list-jobs \
  --account-id 123456789012

```

Output:

```

{
  "Jobs": [
    {
      "Operation": "S3PutObjectTagging",
      "ProgressSummary": {
        "NumberOfTasksFailed": 0,
        "NumberOfTasksSucceeded": 8,
        "TotalNumberOfTasks": 8
      },
      "CreationTime": "2019-10-03T21:48:48.048Z",
    }
  ]
}

```



```

        "Status": "Complete",
        "JobId": "93735294-df46-44d5-8638-6356f335324e",
        "Priority": 42
    },
    {
        "Operation": "S3PutObjectTagging",
        "ProgressSummary": {
            "NumberOfTasksFailed": 0,
            "NumberOfTasksSucceeded": 0,
            "TotalNumberOfTasks": 0
        },
        "CreationTime": "2019-10-03T21:46:07.084Z",
        "Status": "Failed",
        "JobId": "3f3c7619-02d3-4779-97f6-1d98dd313108",
        "Priority": 42
    },
]
}

```

- For API details, see [ListJobs](#) in *AWS CLI Command Reference*.

put-access-point-policy

The following code example shows how to use `put-access-point-policy`.

AWS CLI

To set an access point policy

The following `put-access-point-policy` example places the specified access point policy for the access point `finance-ap` in account `123456789012`. If the access point `finance-ap` already has a policy, this command replaces the existing policy with the one specified in this command. Before running this example, replace the account number, access point name, and policy statements with appropriate values for your use case.

```

aws s3control put-access-point-policy \
  --account-id 123456789012 \
  --name finance-ap \
  --policy file://ap-policy.json

```

Contents of `ap-policy.json`:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:user/Alice"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/finance-ap/
object/Alice/*"
    }
  ]
}
```

This command produces no output.

For more information, see [Managing Data Access with Amazon S3 Access Points](#) in the *Amazon Simple Storage Service Developer Guide*.

- For API details, see [PutAccessPointPolicy](#) in *AWS CLI Command Reference*.

put-public-access-block

The following code example shows how to use `put-public-access-block`.

AWS CLI

To edit block public access settings for an account

The following `put-public-access-block` example toggles all block public access settings to `true` for the specified account.

```
aws s3control put-public-access-block \
  --account-id 123456789012 \
  --public-access-block-configuration '{"BlockPublicAcls": true,
  "IgnorePublicAcls": true, "BlockPublicPolicy": true, "RestrictPublicBuckets":
  true}'
```

This command produces no output.

- For API details, see [PutPublicAccessBlock](#) in *AWS CLI Command Reference*.

submit-multi-region-access-point-routes

The following code example shows how to use `submit-multi-region-access-point-routes`.

AWS CLI

To update your Multi-Region Access Point routing configuration

The following `submit-multi-region-access-point-routes` example updates the routing statuses of `DOC-EXAMPLE-BUCKET-1` and `DOC-EXAMPLE-BUCKET-2` in the `ap-southeast-2` Region for your Multi-Region Access Point.

```
aws s3control submit-multi-region-access-point-routes \  
  --region ap-southeast-2 \  
  --account-id 111122223333 \  
  --mrap MultiRegionAccessPoint_ARN \  
  --route-updates Bucket=DOC-EXAMPLE-BUCKET-1,TrafficDialPercentage=100  
  Bucket=DOC-EXAMPLE-BUCKET-2,TrafficDialPercentage=0
```

This command produces no output.

- For API details, see [SubmitMultiRegionAccessPointRoutes](#) in *AWS CLI Command Reference*.

update-job-priority

The following code example shows how to use `update-job-priority`.

AWS CLI

To update the job priority of an Amazon S3 batch operations job

The following `update-job-priority` example updates the specified job to a new priority.

```
aws s3control update-job-priority \  
  --account-id 123456789012 \  
  --job-id 8d9a18fe-c303-4d39-8ccc-860d372da386 \  
  --priority 52
```

Output:

```
{  
  "JobId": "8d9a18fe-c303-4d39-8ccc-860d372da386",
```

```
"Priority": 52
}
```

- For API details, see [UpdateJobPriority](#) in *AWS CLI Command Reference*.

update-job-status

The following code example shows how to use `update-job-status`.

AWS CLI

To update the status of an Amazon S3 batch operations job

The following `update-job-status` example cancels the specified job which is awaiting approval.

```
aws s3control update-job-status \
  --account-id 123456789012 \
  --job-id 8d9a18fe-c303-4d39-8ccc-860d372da386 \
  --requested-job-status Cancelled
```

Output:

```
{
  "Status": "Cancelled",
  "JobId": "8d9a18fe-c303-4d39-8ccc-860d372da386"
}
```

The following `update-job-status` example confirms and runs the specified which is awaiting approval.

```
aws s3control update-job-status \
  --account-id 123456789012 \
  --job-id 5782949f-3301-4fb3-be34-8d5bab54dbca \
  --requested-job-status Ready
```

Output::

```
{
  "Status": "Ready",
  "JobId": "5782949f-3301-4fb3-be34-8d5bab54dbca"
}
```

```
}
```

The following `update-job-status` example cancels the specified job which is running.

```
aws s3control update-job-status \  
  --account-id 123456789012 \  
  --job-id 5782949f-3301-4fb3-be34-8d5bab54dbca \  
  --requested-job-status Cancelled  
  
Output::  
{  
    "Status": "Cancelling",  
    "JobId": "5782949f-3301-4fb3-be34-8d5bab54dbca"  
}
```

- For API details, see [UpdateJobStatus](#) in *AWS CLI Command Reference*.

S3 Glacier examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with S3 Glacier.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

abort-multipart-upload

The following code example shows how to use `abort-multipart-upload`.

AWS CLI

The following command deletes an in-progress multipart upload to a vault named `my-vault`:

```
aws glacier abort-multipart-upload --account-id - --vault-name my-vault
--upload-id 19gaRezEXAMPLES6Ry5YYdqthH0C_kGRCT03L9yetr220UmPtBYKk-
0ssZtLqyFu7sY1_lR7vgFuJV6NtcV5zpsJ
```

This command does not produce any output. Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account. The upload ID is returned by the `aws glacier initiate-multipart-upload` command and can also be obtained by using `aws glacier list-multipart-uploads`.

For more information on multipart uploads to Amazon Glacier using the AWS CLI, see *Using Amazon Glacier* in the *AWS CLI User Guide*.

- For API details, see [AbortMultipartUpload](#) in *AWS CLI Command Reference*.

`abort-vault-lock`

The following code example shows how to use `abort-vault-lock`.

AWS CLI

To abort an in-progress vault lock process

The following `abort-vault-lock` example deletes a vault lock policy from the specified vault and resets the lock state of the vault lock to unlocked.

```
aws glacier abort-vault-lock \
--account-id - \
--vault-name MyVaultName
```

This command produces no output.

For more information, see [Abort Vault Lock \(DELETE lock-policy\)](#) in the *Amazon Glacier API Developer Guide*.

- For API details, see [AbortVaultLock](#) in *AWS CLI Command Reference*.

add-tags-to-vault

The following code example shows how to use `add-tags-to-vault`.

AWS CLI

The following command adds two tags to a vault named `my-vault`:

```
aws glacier add-tags-to-vault --account-id - --vault-name my-vault --tags
id=1234,date=july2015
```

Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

- For API details, see [AddTagsToVault](#) in *AWS CLI Command Reference*.

complete-multipart-upload

The following code example shows how to use `complete-multipart-upload`.

AWS CLI

The following command completes multipart upload for a 3 MiB archive:

```
aws glacier complete-multipart-upload --archive-size 3145728 --checksum
9628195fcdcbcbbe76cdde456d4646fa7de5f219fb39823836d81f0cc0e18aa67
--upload-id 19gaRezEXAMPLES6Ry5YYdqthH0C_kGRCT03L9yetr220UmPtBYKk-
0ssZtLqyFu7sY1_lR7vgFuJV6NtcV5zpsJ --account-id - --vault-name my-vault
```

Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

The upload ID is returned by the `aws glacier initiate-multipart-upload` command and can also be obtained by using `aws glacier list-multipart-uploads`. The checksum parameter takes a SHA-256 tree hash of the archive in hexadecimal.

For more information on multipart uploads to Amazon Glacier using the AWS CLI, including instructions on calculating a tree hash, see *Using Amazon Glacier* in the *AWS CLI User Guide*.

- For API details, see [CompleteMultipartUpload](#) in *AWS CLI Command Reference*.

complete-vault-lock

The following code example shows how to use `complete-vault-lock`.

AWS CLI

To complete an in-progress vault lock process

The following `complete-vault-lock` example completes the in-progress locking progress for the specified vault and sets the lock state of the vault lock to `Locked`. You get the value for the `lock-id` parameter when you run `initiate-lock-process`.

```
aws glacier complete-vault-lock \  
  --account-id - \  
  --vault-name MyVaultName \  
  --lock-id 9QZgEXAMPLEPhvL6xEXAMPLE
```

This command produces no output.

For more information, see [Complete Vault Lock \(POST lockId\)](#) in the *Amazon Glacier API Developer Guide*.

- For API details, see [CompleteVaultLock](#) in *AWS CLI Command Reference*.

create-vault

The following code example shows how to use `create-vault`.

AWS CLI

The following command creates a new vault named `my-vault`:

```
aws glacier create-vault --vault-name my-vault --account-id -
```

Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

- For API details, see [CreateVault](#) in *AWS CLI Command Reference*.

delete-archive

The following code example shows how to use `delete-archive`.

AWS CLI

To delete an archive from a vault

The following `delete-archive` example removes the specified archive from `example_vault`.

```
aws glacier delete-archive \  
  --account-id 111122223333 \  
  --vault-name example_vault \  
  --archive-id Sc0u9ZP8yaWkmh-XG1IvAVprtLhaLCGnNwN15I5x9HqPIkX5mjc0DrId3Ln-  
  Gi_k2HzmlIDZUz117KSdVMdMXLuFwi9PJUitxw073edQ43eT1MwkH0pd9zVSAuV_XXZBVhKhyGhJ7w
```

This command produces no output.

- For API details, see [DeleteArchive](#) in *AWS CLI Command Reference*.

`delete-vault-access-policy`

The following code example shows how to use `delete-vault-access-policy`.

AWS CLI

To remove the access policy of a vault

The following `delete-vault-access-policy` example removes the access policy for the specified vault.

```
aws glacier delete-vault-access-policy \  
  --account-id 111122223333 \  
  --vault-name example_vault
```

This command produces no output.

- For API details, see [DeleteVaultAccessPolicy](#) in *AWS CLI Command Reference*.

`delete-vault-notifications`

The following code example shows how to use `delete-vault-notifications`.

AWS CLI

To remove the SNS notifications for a vault

The following `delete-vault-notifications` example removes notifications sent by Amazon Simple Notification Service (Amazon SNS) for the specified vault.

```
aws glacier delete-vault-notifications \  
  --account-id 111122223333 \  
  --vault-name example_vault
```

This command produces no output.

- For API details, see [DeleteVaultNotifications](#) in *AWS CLI Command Reference*.

delete-vault

The following code example shows how to use `delete-vault`.

AWS CLI

The following command deletes a vault named `my-vault`:

```
aws glacier delete-vault --vault-name my-vault --account-id -
```

This command does not produce any output. Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

- For API details, see [DeleteVault](#) in *AWS CLI Command Reference*.

describe-job

The following code example shows how to use `describe-job`.

AWS CLI

The following command retrieves information about an inventory retrieval job on a vault named `my-vault`:

```
aws glacier describe-job --account-id - --vault-name my-  
vault --job-id zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-  
R047Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_Xq1NHS61ds04CnMW
```

Output:

```
{
```

```
"InventoryRetrievalParameters": {
  "Format": "JSON"
},
"VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-vault",
"Completed": false,
"JobId": "zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
R047Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_Xq1NHS61ds04CnMW",
"Action": "InventoryRetrieval",
"CreationDate": "2015-07-17T20:23:41.616Z",
"StatusCode": "InProgress"
}
```

The job ID can be found in the output of `aws glacier initiate-job` and `aws glacier list-jobs`. Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

- For API details, see [DescribeJob](#) in *AWS CLI Command Reference*.

describe-vault

The following code example shows how to use `describe-vault`.

AWS CLI

The following command retrieves data about a vault named `my-vault`:

```
aws glacier describe-vault --vault-name my-vault --account-id -
```

Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

- For API details, see [DescribeVault](#) in *AWS CLI Command Reference*.

get-data-retrieval-policy

The following code example shows how to use `get-data-retrieval-policy`.

AWS CLI

The following command gets the data retrieval policy for the in-use account:

```
aws glacier get-data-retrieval-policy --account-id -
```

Output:

```
{
  "Policy": {
    "Rules": [
      {
        "BytesPerHour": 10737418240,
        "Strategy": "BytesPerHour"
      }
    ]
  }
}
```

Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

- For API details, see [GetDataRetrievalPolicy](#) in *AWS CLI Command Reference*.

get-job-output

The following code example shows how to use `get-job-output`.

AWS CLI

The following command saves the output from a vault inventory job to a file in the current directory named `output.json`:

```
aws glacier get-job-output --account-id - --vault-name my-
vault --job-id zbxcm3Z_3z5UkooroF7SuZKrxgGoDc3RloGduS7Eg-
R047Yc6FxsdsGBgf_Q2DK5Ejh18CnTS5XW4_Xq1NHS61ds04CnMW output.json
```

The `job-id` is available in the output of `aws glacier list-jobs`. Note that the output file name is a positional argument that is not prefixed by an option name. Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

Output:

```
{
  "status": 200,
  "acceptRanges": "bytes",
  "contentType": "application/json"
```

```
}

```

output.json:

```
{
  "VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-vault",
  "InventoryDate": "2015-04-07T00:26:18Z",
  "ArchiveList": [
    {
      "ArchiveId": "kKB7ymWJVpPSwhGP6ycS0Aekp9ZYe_--zM_mw6k76ZFGEIWQX-ybtRDvc2VkpSDtfKmqRj0IRQLSGsNuDp-AJVlu2ccmDSyDUMzWkbwbpAdGATGDiB3hH00bjbGehXTcApVud_wyDw",
      "ArchiveDescription": "multipart upload test",
      "CreationDate": "2015-04-06T22:24:34Z",
      "Size": 3145728,
      "SHA256TreeHash": "9628195fcdbcb"
    }
  ]
}
```

- For API details, see [GetJobOutput](#) in *AWS CLI Command Reference*.

get-vault-access-policy

The following code example shows how to use `get-vault-access-policy`.

AWS CLI

To retrieve the access policy of a vault

The following `get-vault-access-policy` example retrieves the access policy for the specified vault.

```
aws glacier get-vault-access-policy \
  --account-id 111122223333 \
  --vault-name example_vault
```

Output:

```
{
  "policy": {
    "Policy": "{\n\"Version\": \"2012-10-17\", \"Statement\": [\n\"Effect\": \"Allow\", \"Principal\": {\n\"AWS\": \"arn:aws:iam::444455556666:root\"}, \"Action\": \"glacier:ListJobs\", \"Resource\": \"arn:aws:glacier:us-east-1:111122223333:vaults/example_vault\"}, {\n\"Effect\": \"Allow\", \"Principal\": {\n\"AWS\": \"arn:aws:iam::444455556666:root\"}, \"Action\": \"glacier:UploadArchive\", \"Resource\": \"arn:aws:glacier:us-east-1:111122223333:vaults/example_vault\"}]\n}"
  }
}
```

- For API details, see [GetVaultAccessPolicy](#) in *AWS CLI Command Reference*.

get-vault-lock

The following code example shows how to use `get-vault-lock`.

AWS CLI

To get the details of a vault lock

The following `get-vault-lock` example retrieved the details about the lock for the specified vault.

```
aws glacier get-vault-lock \
  --account-id - \
  --vault-name MyVaultName
```

Output:

```
{
  "Policy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Sid\":\"Define-vault-lock\",\"Effect\":\"Deny\",\"Principal\":{\"AWS\":\"arn:aws:iam:999999999999:root\"},\"Action\":\"glacier:DeleteArchive\",\"Resource\":\"arn:aws:glacier:us-west-2:999999999999:vaults/MyVaultName\",\"Condition\":{\"NumericLessThanEquals\":{\"glacier:ArchiveAgeinDays\":\"365\"}}}]}\",
  "State": "Locked",
  "CreationDate": "2019-07-29T22:25:28.640Z"
}
```

For more information, see [Get Vault Lock \(GET lock-policy\)](#) in the *Amazon Glacier API Developer Guide*.

- For API details, see [GetVaultLock](#) in *AWS CLI Command Reference*.

get-vault-notifications

The following code example shows how to use `get-vault-notifications`.

AWS CLI

The following command gets a description of the notification configuration for a vault named `my-vault`:

```
aws glacier get-vault-notifications --account-id - --vault-name my-vault
```

Output:

```
{
  "vaultNotificationConfig": {
    "Events": [
      "InventoryRetrievalCompleted",
      "ArchiveRetrievalCompleted"
    ],
    "SNSTopic": "arn:aws:sns:us-west-2:0123456789012:my-vault"
  }
}
```

If no notifications have been configured for the vault, an error is returned. Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

- For API details, see [GetVaultNotifications](#) in *AWS CLI Command Reference*.

initiate-job

The following code example shows how to use `initiate-job`.

AWS CLI

The following command initiates a job to get an inventory of the vault `my-vault`:

```
aws glacier initiate-job --account-id - --vault-name my-vault --job-parameters
'{"Type": "inventory-retrieval"}'
```

Output:

```
{
  "location": "/0123456789012/vaults/my-vault/jobs/
zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
R047Yc6FxsdBgf_Q2DK5Ejh18CnTS5XW4_Xq1NHS61ds04CnMW",
  "jobId": "zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
R047Yc6FxsdBgf_Q2DK5Ejh18CnTS5XW4_Xq1NHS61ds04CnMW"
}
```

Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

The following command initiates a job to retrieve an archive from the vault `my-vault`:

```
aws glacier initiate-job --account-id - --vault-name my-vault --job-parameters
file://job-archive-retrieval.json
```

`job-archive-retrieval.json` is a JSON file in the local folder that specifies the type of job, archive ID, and some optional parameters:

```
{
  "Type": "archive-retrieval",
  "ArchiveId": "kKB7ymWJVpPSwhGP6ycS0Aekp9ZYe_--zM_mw6k76ZFGIEWQX-
ybtRDvc2VkpSDtFkmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hH00bjbGehXTcApVud_wyDw",
  "Description": "Retrieve archive on 2015-07-17",
  "SNSTopic": "arn:aws:sns:us-west-2:0123456789012:my-topic"
}
```

Archive IDs are available in the output of `aws glacier upload-archive` and `aws glacier get-job-output`.

Output:

```
{
  "location": "/011685312445/vaults/mwunderl/jobs/l7IL5-
EkXyEY9Ws95fClzIbk205uLYaFdAY0i-
azsX_Z8V6NH4yERHzars8wTKYQMX6nBDI9cMNHzyZJ059-8N9aHWav",
  "jobId": "l7IL5-EkXy205uLYaFdAY0iEY9Ws95fClzIbk-
azsX_Z8V6NH4yERHzars8wTKYQMX6nBDI9cMNHzyZJ059-8N9aHWav"
}
```

See *Initiate Job* in the *Amazon Glacier API Reference* for details on the job parameters format.

- For API details, see [InitiateJob](#) in *AWS CLI Command Reference*.

initiate-multipart-upload

The following code example shows how to use `initiate-multipart-upload`.

AWS CLI

The following command initiates a multipart upload to a vault named `my-vault` with a part size of 1 MiB (1024 x 1024 bytes) per file:

```
aws glacier initiate-multipart-upload --account-id - --part-size 1048576 --vault-name my-vault --archive-description "multipart upload test"
```

The archive description parameter is optional. Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

This command outputs an upload ID when successful. Use the upload ID when uploading each part of your archive with `aws glacier upload-multipart-part`. For more information on multipart uploads to Amazon Glacier using the AWS CLI, see *Using Amazon Glacier in the AWS CLI User Guide*.

- For API details, see [InitiateMultipartUpload](#) in *AWS CLI Command Reference*.

initiate-vault-lock

The following code example shows how to use `initiate-vault-lock`.

AWS CLI

To initiate the vault locking process

The following `initiate-vault-lock` example installs a vault lock policy on the specified vault and sets the lock state of the vault lock to `InProgress`. You must complete the process by calling `complete-vault-lock` within 24 hours to set the state of the vault lock to `Locked`.

```
aws glacier initiate-vault-lock \  
  --account-id - \  
  --vault-name MyVaultName \  
  --policy file://vault_lock_policy.json
```

Contents of `vault_lock_policy.json`:

```
{"Policy":{"Version\":\"2012-10-17\",\"Statement\":[{\"Sid\":\"Define-vault-lock\",\"Effect\":\"Deny\",\"Principal\":{\"AWS\":\"arn:aws:iam:999999999999:root\"},\"Action\":\"glacier:DeleteArchive\",\"Resource\":\"arn:aws:glacier:us-
```

```
west-2:999999999999:vaults/examplevault\", \"Condition\": { \"NumericLessThanEquals\": { \"glacier:ArchiveAgeInDays\": \"365\" } } } ] ] ] }
```

The output is the vault lock ID that you can use to complete the vault lock process.

```
{
  "lockId": "9QZgEXAMPLEPhvL6xEXAMPLE"
}
```

For more information, see [Initiate Vault Lock \(POST lock-policy\)](#) in the *Amazon Glacier API Developer Guide*.

- For API details, see [InitiateVaultLock](#) in *AWS CLI Command Reference*.

list-jobs

The following code example shows how to use `list-jobs`.

AWS CLI

The following command lists in-progress and recently completed jobs for a vault named `my-vault`:

```
aws glacier list-jobs --account-id - --vault-name my-vault
```

Output:

```
{
  "JobList": [
    {
      "VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-vault",
      "RetrievalByteRange": "0-3145727",
      "SNSTopic": "arn:aws:sns:us-west-2:0123456789012:my-vault",
      "Completed": false,
      "SHA256TreeHash":
"9628195fcdbcbbe76cdde932d4646fa7de5f219fb39823836d81f0cc0e18aa67",
      "JobId": "17IL5-EkXyEY9Ws95fClzIbk205uLYaFdAY0i-
azsX_Z8V6NH4yERHzars8wTKYQMX6nBDI9cMNHzyZJ059-8N9aHWav",
      "ArchiveId": "kKB7ymWJVpPSwhGP6ycS0Aekp9ZYe_--zM_mw6k76ZFGIEWQX-
ybtRDvc2VkPSDtFkMqrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKwbpbAdGATGDiB3hH00bjbGehXTcApVud_wyDw",
```

```

        "JobDescription": "Retrieve archive on 2015-07-17",
        "ArchiveSizeInBytes": 3145728,
        "Action": "ArchiveRetrieval",
        "ArchiveSHA256TreeHash":
"9628195fcdcbbbe76cdde932d4646fa7de5f219fb39823836d81f0cc0e18aa67",
        "CreationDate": "2015-07-17T21:16:13.840Z",
        "StatusCode": "InProgress"
    },
    {
        "InventoryRetrievalParameters": {
            "Format": "JSON"
        },
        "VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-vault",
        "Completed": false,
        "JobId": "zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
R047Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61ds04CnMW",
        "Action": "InventoryRetrieval",
        "CreationDate": "2015-07-17T20:23:41.616Z",
        "StatusCode": ""InProgress""
    }
]
}

```

Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

- For API details, see [ListJobs](#) in *AWS CLI Command Reference*.

list-multipart-uploads

The following code example shows how to use `list-multipart-uploads`.

AWS CLI

The following command shows all of the in-progress multipart uploads for a vault named `my-vault`:

```
aws glacier list-multipart-uploads --account-id - --vault-name my-vault
```

Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

For more information on multipart uploads to Amazon Glacier using the AWS CLI, see *Using Amazon Glacier in the AWS CLI User Guide*.

- For API details, see [ListMultipartUploads](#) in *AWS CLI Command Reference*.

list-parts

The following code example shows how to use `list-parts`.

AWS CLI

The following command lists the uploaded parts for a multipart upload to a vault named `my-vault`:

```
aws glacier list-parts --account-id - --vault-name my-vault --upload-id "SYZi7qnL-
YGqGwAm8Kn3BLP2E1NCvnB-5961R09CSaPmPwkYGH0qeN_nX3-Vhnd2yF0KfB5FkmbnBU9Gubbd1Cs8ut-D"
```

Output:

```
{
  "MultipartUploadId": "SYZi7qnL-
YGqGwAm8Kn3BLP2E1NCvnB-5961R09CSaPmPwkYGH0qeN_nX3-Vhnd2yF0KfB5FkmbnBU9Gubbd1Cs8ut-
D",
  "Parts": [
    {
      "RangeInBytes": "0-1048575",
      "SHA256TreeHash":
"e1f2a7cd6e047350f69b9f8cfa60fa606fe2f02802097a9a026360a7edc1f553"
    },
    {
      "RangeInBytes": "1048576-2097151",
      "SHA256TreeHash":
"43cf3061fb95796aed99a11a6aa3cd8f839eed15e655ab0a597126210636aee6"
    }
  ],
  "VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-vault",
  "CreationDate": "2015-07-18T00:05:23.830Z",
  "PartSizeInBytes": 1048576
}
```

Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

For more information on multipart uploads to Amazon Glacier using the AWS CLI, see [Using Amazon Glacier in the AWS CLI User Guide](#).

- For API details, see [ListParts](#) in *AWS CLI Command Reference*.

list-provisioned-capacity

The following code example shows how to use `list-provisioned-capacity`.

AWS CLI

To retrieve the provisioned capacity units

The following `list-provisioned-capacity` example retrieves details for any provisioned capacity units for the specified account.

```
aws glacier list-provisioned-capacity \  
  --account-id 111122223333
```

Output:

```
{  
  "ProvisionedCapacityList": [  
    {  
      "CapacityId": "HpASAUvfRFiVDb0jMfEIcr8K",  
      "ExpirationDate": "2020-03-18T19:59:24.000Z",  
      "StartDate": "2020-02-18T19:59:24.912Z"  
    }  
  ]  
}
```

- For API details, see [ListProvisionedCapacity](#) in *AWS CLI Command Reference*.

list-tags-for-vault

The following code example shows how to use `list-tags-for-vault`.

AWS CLI

The following command lists the tags applied to a vault named `my-vault`:

```
aws glacier list-tags-for-vault --account-id - --vault-name my-vault
```

Output:

```
{
  "Tags": {
    "date": "july2015",
    "id": "1234"
  }
}
```

Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

- For API details, see [ListTagsForVault](#) in *AWS CLI Command Reference*.

list-vaults

The following code example shows how to use `list-vaults`.

AWS CLI

The following command lists the vaults in the default account and region:

```
aws glacier list-vaults --account-id -
```

Output:

```
{
  "VaultList": [
    {
      "SizeInBytes": 3178496,
      "VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-vault",
      "LastInventoryDate": "2015-04-07T00:26:19.028Z",
      "VaultName": "my-vault",
      "NumberOfArchives": 1,
      "CreationDate": "2015-04-06T21:23:45.708Z"
    }
  ]
}
```

Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

- For API details, see [ListVaults](#) in *AWS CLI Command Reference*.

purchase-provisioned-capacity

The following code example shows how to use `purchase-provisioned-capacity`.

AWS CLI

To purchase a provisioned capacity unit

The following `purchase-provisioned-capacity` example purchases a provisioned capacity unit.

```
aws glacier purchase-provisioned-capacity \  
  --account-id 111122223333
```

Output:

```
{  
  "capacityId": "HpASAUvfRFiVDb0jMfEIcr8K"  
}
```

- For API details, see [PurchaseProvisionedCapacity](#) in *AWS CLI Command Reference*.

remove-tags-from-vault

The following code example shows how to use `remove-tags-from-vault`.

AWS CLI

The following command removes a tag with the key `date` from a vault named `my-vault`:

```
aws glacier remove-tags-from-vault --account-id - --vault-name my-vault --tag-keys  
  date
```

Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

- For API details, see [RemoveTagsFromVault](#) in *AWS CLI Command Reference*.

set-data-retrieval-policy

The following code example shows how to use `set-data-retrieval-policy`.

AWS CLI

The following command configures a data retrieval policy for the in-use account:

```
aws glacier set-data-retrieval-policy --account-id - --policy file://data-retrieval-policy.json
```

`data-retrieval-policy.json` is a JSON file in the current folder that specifies a data retrieval policy:

```
{
  "Rules":[
    {
      "Strategy":"BytesPerHour",
      "BytesPerHour":10737418240
    }
  ]
}
```

Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

The following command sets the data retrieval policy to `FreeTier` using inline JSON:

```
aws glacier set-data-retrieval-policy --account-id - --policy '{"Rules": [{"Strategy":"FreeTier"}]}'
```

See `Set Data Retrieval Policy` in the *Amazon Glacier API Reference* for details on the policy format.

- For API details, see [SetDataRetrievalPolicy](#) in *AWS CLI Command Reference*.

set-vault-access-policy

The following code example shows how to use `set-vault-access-policy`.

AWS CLI

To set the access policy of a vault

The following `set-vault-access-policy` example attaches a permission policy to the specified vault.

```
aws glacier set-vault-access-policy \  
  --account-id 111122223333 \  
  --vault-name example_vault \  
  --policy '{"Policy": "{\"Version\":\"2012-10-17\",\"Statement\":  
[{\n\"Effect\":\n\"Allow\", \n\"Principal\":{\n\"AWS\":\n\"arn:aws:iam:444455556666:root  
\n\"}, \n\"Action\":\n\"glacier:ListJobs\", \n\"Resource\":\n\"arn:aws:glacier:us-  
east-1:111122223333:vaults/example_vault\"}, {\n\"Effect\":\n\"Allow\", \n\"Principal\":  
{\n\"AWS\":\n\"arn:aws:iam:444455556666:root\", \n\"Action\":\n\"glacier:UploadArchive\",  
\n\"Resource\":\n\"arn:aws:glacier:us-east-1:111122223333:vaults/example_vault\"}}]}'
```

This command produces no output.

- For API details, see [SetVaultAccessPolicy](#) in *AWS CLI Command Reference*.

set-vault-notifications

The following code example shows how to use `set-vault-notifications`.

AWS CLI

The following command configures SNS notifications for a vault named `my-vault`:

```
aws glacier set-vault-notifications --account-id - --vault-name my-vault --vault-  
notification-config file://notificationconfig.json
```

`notificationconfig.json` is a JSON file in the current folder that specifies an SNS topic and the events to publish:

```
{  
  "SNSTopic": "arn:aws:sns:us-west-2:0123456789012:my-vault",  
  "Events": ["ArchiveRetrievalCompleted", "InventoryRetrievalCompleted"]  
}
```

Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

- For API details, see [SetVaultNotifications](#) in *AWS CLI Command Reference*.

upload-archive

The following code example shows how to use `upload-archive`.

AWS CLI

The following command uploads an archive in the current folder named `archive.zip` to a vault named `my-vault`:

```
aws glacier upload-archive --account-id - --vault-name my-vault --body archive.zip
```

Output:

```
{
  "archiveId": "kKB7ymWJVpPSwhGP6ycS0Aekp9ZYe_--zM_mw6k76ZFGIEWQX-
ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJV1u2ccmDSyDUmZwKbwbpAdGATGDiB3hH00bjbGehXTcApVud_wyDw",
  "checksum": "969fb39823836d81f0cc028195fcdcbbbe76cdde932d4646fa7de5f21e18aa67",
  "location": "/0123456789012/vaults/my-vault/archives/
kKB7ymWJVpPSwhGP6ycS0Aekp9ZYe_--zM_mw6k76ZFGIEWQX-ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJV1u2ccmDSyDUmZwKbwbpAdGATGDiB3hH00bjbGehXTcApVud_wyDw"
}
```

Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

To retrieve an uploaded archive, initiate a retrieval job with the `aws glacier initiate-job` command.

- For API details, see [UploadArchive](#) in *AWS CLI Command Reference*.

upload-multipart-part

The following code example shows how to use `upload-multipart-part`.

AWS CLI

The following command uploads the first 1 MiB (1024 x 1024 bytes) part of an archive:

```
aws glacier upload-multipart-part --body part1 --range 'bytes
0-1048575/*' --account-id - --vault-name my-vault --upload-
id 19gaRezEXAMPLES6Ry5YYdqthH0C_kGRCT03L9yetr220UmPtBYKk-
0ssZtLqyFu7sY1_1R7vgFuJV6NtcV5zpsJ
```

Amazon Glacier requires an account ID argument when performing operations, but you can use a hyphen to specify the in-use account.

The body parameter takes a path to a part file on the local filesystem. The range parameter takes an HTTP content range indicating the bytes that the part occupies in the completed archive. The upload ID is returned by the `aws glacier initiate-multipart-upload` command and can also be obtained by using `aws glacier list-multipart-uploads`.

For more information on multipart uploads to Amazon Glacier using the AWS CLI, see *Using Amazon Glacier* in the *AWS CLI User Guide*.

- For API details, see [UploadMultipartPart](#) in *AWS CLI Command Reference*.

Secrets Manager examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Secrets Manager.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

batch-get-secret-value

The following code example shows how to use `batch-get-secret-value`.

AWS CLI

Example 1: To retrieve the secret value for a group of secrets listed by name

The following `batch-get-secret-value` example gets the secret value secrets for three secrets.

```
aws secretsmanager batch-get-secret-value \  
  --secret-id-list MySecret1 MySecret2 MySecret3
```

Output:

```
{  
  "SecretValues": [  
    {  
      "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MySecret1-  
a1b2c3",  
      "Name": "MySecret1",  
      "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",  
      "SecretString": "{\"username\":\"diego_ramirez\",\"password\":\"EXAMPLE-  
PASSWORD\",\"engine\":\"mysql\",\"host\":\"secretsmanagertutorial.cluster.us-  
west-2.rds.amazonaws.com\",\"port\":3306,\"dbClusterIdentifier\":  
\"secretsmanagertutorial\"}",  
      "VersionStages": [  
        "AWSCURRENT"  
      ],  
      "CreateDate": "1523477145.729"  
    },  
    {  
      "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MySecret2-  
a1b2c3",  
      "Name": "MySecret2",  
      "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLEebbbb",  
      "SecretString": "{\"username\":\"akua_mansa\",\"password\":\"EXAMPLE-  
PASSWORD\"}",  
      "VersionStages": [  

```

```

        "AWSCURRENT"
    ],
    "CreateDate": "1673477781.275"
  },
  {
    "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MySecret3-
a1b2c3",
    "Name": "MySecret3",
    "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLEecccc",
    "SecretString": "{\"username\": \"jie_liu\", \"password\": \"EXAMPLE-
PASSWORD\"",
    "VersionStages": [
      "AWSCURRENT"
    ],
    "CreateDate": "1373477721.124"
  }
],
"Errors": []
}

```

For more information, see [Retrieve a group of secrets in a batch](#) in the *AWS Secrets Manager User Guide*.

Example 2: To retrieve the secret value for a group of secrets selected by filter

The following `batch-get-secret-value` example gets the secret value secrets in your account that have `MySecret` in the name. Filtering by name is case sensitive.

```
aws secretsmanager batch-get-secret-value \
  --filters Key="name",Values="MySecret"
```

Output:

```

{
  "SecretValues": [
    {
      "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MySecret1-
a1b2c3",
      "Name": "MySecret1",
      "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLEeaaaa",
      "SecretString": "{\"username\": \"diego_ramirez\", \"password\": \"EXAMPLE-
PASSWORD\", \"engine\": \"mysql\", \"host\": \"secretsmanagertutorial.cluster.us-

```

```

west-2.rds.amazonaws.com\", \"port\": 3306, \"dbClusterIdentifier\":
\"secretsmanagertutorial\"}],
  \"VersionStages\": [
    \"AWSCURRENT\"
  ],
  \"CreateDate\": \"1523477145.729\"
},
{
  \"ARN\": \"arn:aws:secretsmanager:us-west-2:123456789012:secret:MySecret2-
a1b2c3\",
  \"Name\": \"MySecret2\",
  \"VersionId\": \"a1b2c3d4-5678-90ab-cdef-EXAMPLEebbbb\",
  \"SecretString\": \"{\\\"username\\\":\\\"akua_mansa\\\",\\\"password\\\":\\\"EXAMPLE-
PASSWORD\\\"\",
  \"VersionStages\": [
    \"AWSCURRENT\"
  ],
  \"CreateDate\": \"1673477781.275\"
},
{
  \"ARN\": \"arn:aws:secretsmanager:us-west-2:123456789012:secret:MySecret3-
a1b2c3\",
  \"Name\": \"MySecret3\",
  \"VersionId\": \"a1b2c3d4-5678-90ab-cdef-EXAMPLEecccc\",
  \"SecretString\": \"{\\\"username\\\":\\\"jie_liu\\\",\\\"password\\\":\\\"EXAMPLE-
PASSWORD\\\"\",
  \"VersionStages\": [
    \"AWSCURRENT\"
  ],
  \"CreateDate\": \"1373477721.124\"
}
],
\"Errors\": []
}

```

For more information, see [Retrieve a group of secrets in a batch](#) in the *AWS Secrets Manager User Guide*.

- For API details, see [BatchGetSecretValue](#) in *AWS CLI Command Reference*.

cancel-rotate-secret

The following code example shows how to use `cancel-rotate-secret`.

AWS CLI

To turn off automatic rotation for a secret

The following `cancel-rotate-secret` example turns off automatic rotation for a secret. To resume rotation, call `rotate-secret`.

```
aws secretsmanager cancel-rotate-secret \  
  --secret-id MyTestSecret
```

Output:

```
{  
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-a1b2c3",  
  "Name": "MyTestSecret"  
}
```

For more information, see [Rotate a secret](#) in the *Secrets Manager User Guide*.

- For API details, see [CancelRotateSecret](#) in *AWS CLI Command Reference*.

create-secret

The following code example shows how to use `create-secret`.

AWS CLI

Example 1: To create a secret

The following `create-secret` example creates a secret with two key-value pairs.

```
aws secretsmanager create-secret \  
  --name MyTestSecret \  
  --description "My test secret created with the CLI." \  
  --secret-string "{\"user\": \"diegor\", \"password\": \"EXAMPLE-PASSWORD\"}"
```

Output:

```
{  
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-a1b2c3",
```

```
"Name": "MyTestSecret",
"VersionId": "EXAMPLE1-90ab-cdef-fedc-ba987EXAMPLE"
}
```

For more information, see [Create a secret](#) in the *Secrets Manager User Guide*.

Example 2: To create a secret from credentials in a JSON file

The following `create-secret` example creates a secret from credentials in a file. For more information, see [Loading AWS CLI parameters from a file](#) in the *AWS CLI User Guide*.

```
aws secretsmanager create-secret \
  --name MyTestSecret \
  --secret-string file://mycreds.json
```

Contents of `mycreds.json`:

```
{
  "engine": "mysql",
  "username": "saanvis",
  "password": "EXAMPLE-PASSWORD",
  "host": "my-database-endpoint.us-west-2.rds.amazonaws.com",
  "dbname": "myDatabase",
  "port": "3306"
}
```

Output:

```
{
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-a1b2c3",
  "Name": "MyTestSecret",
  "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

For more information, see [Create a secret](#) in the *Secrets Manager User Guide*.

- For API details, see [CreateSecret](#) in *AWS CLI Command Reference*.

delete-resource-policy

The following code example shows how to use `delete-resource-policy`.

AWS CLI

To delete the resource-based policy attached to a secret

The following `delete-resource-policy` example deletes the resource-based policy attached to a secret.

```
aws secretsmanager delete-resource-policy \  
  --secret-id MyTestSecret
```

Output:

```
{  
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-  
a1b2c3",  
  "Name": "MyTestSecret"  
}
```

For more information, see [Authentication and access control](#) in the *Secrets Manager User Guide*.

- For API details, see [DeleteResourcePolicy](#) in *AWS CLI Command Reference*.

delete-secret

The following code example shows how to use `delete-secret`.

AWS CLI

Example 1: To delete a secret

The following `delete-secret` example deletes a secret. You can recover the secret with `restore-secret` until the date and time in the `DeletionDate` response field. To delete a secret that is replicated to other regions, first remove its replicas with `remove-regions-from-replication`, and then call `delete-secret`.

```
aws secretsmanager delete-secret \  
  --secret-id MyTestSecret \  
  --recovery-window-in-days 7
```

Output:

```
{
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
  "Name": "MyTestSecret",
  "DeletionDate": 1524085349.095
}
```

For more information, see [Delete a secret](#) in the *Secrets Manager User Guide*.

Example 2: To delete a secret immediately

The following `delete-secret` example deletes a secret immediately without a recovery window. You can't recover this secret.

```
aws secretsmanager delete-secret \
  --secret-id MyTestSecret \
  --force-delete-without-recovery
```

Output:

```
{
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
  "Name": "MyTestSecret",
  "DeletionDate": 1508750180.309
}
```

For more information, see [Delete a secret](#) in the *Secrets Manager User Guide*.

- For API details, see [DeleteSecret](#) in *AWS CLI Command Reference*.

describe-secret

The following code example shows how to use `describe-secret`.

AWS CLI

To retrieve the details of a secret

The following `describe-secret` example shows the details of a secret.

```
aws secretsmanager describe-secret \  
--secret-id MyTestSecret
```

Output:

```
{  
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-  
Ca8JGt",  
  "Name": "MyTestSecret",  
  "Description": "My test secret",  
  "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/EXAMPLE1-90ab-cdef-fedc-  
ba987EXAMPLE",  
  "RotationEnabled": true,  
  "RotationLambdaARN": "arn:aws:lambda:us-  
west-2:123456789012:function:MyTestRotationLambda",  
  "RotationRules": {  
    "AutomaticallyAfterDays": 2,  
    "Duration": "2h",  
    "ScheduleExpression": "cron(0 16 1,15 * ? *)"  
  },  
  "LastRotatedDate": 1525747253.72,  
  "LastChangedDate": 1523477145.729,  
  "LastAccessedDate": 1524572133.25,  
  "Tags": [  
    {  
      "Key": "SecondTag",  
      "Value": "AnotherValue"  
    },  
    {  
      "Key": "FirstTag",  
      "Value": "SomeValue"  
    }  
  ],  
  "VersionIdsToStages": {  
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111": [  
      "AWSPREVIOUS"  
    ],  
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222": [  
      "AWSCURRENT"  
    ],  
    "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333": [  
      "AWSPENDING"  
    ]  
  }  
}
```

```
},
"CreateDate": 1521534252.66,
"PrimaryRegion": "us-west-2",
"ReplicationStatus": [
  {
    "Region": "eu-west-3",
    "KmsKeyId": "alias/aws/secretsmanager",
    "Status": "InSync",
    "StatusMessage": "Replication succeeded"
  }
]
}
```

For more information, see [Secret](#) in the *Secrets Manager User Guide*.

- For API details, see [DescribeSecret](#) in *AWS CLI Command Reference*.

get-random-password

The following code example shows how to use `get-random-password`.

AWS CLI

To generate a random password

The following `get-random-password` example generates a random password 20 characters long that includes at least one uppercase letter, lowercase letter, number, and punctuation.

```
aws secretsmanager get-random-password \
  --require-each-included-type \
  --password-length 20
```

Output:

```
{
  "RandomPassword": "EXAMPLE-PASSWORD"
}
```

For more information, see [Create and manage secrets](#) in the *Secrets Manager User Guide*.

- For API details, see [GetRandomPassword](#) in *AWS CLI Command Reference*.

get-resource-policy

The following code example shows how to use `get-resource-policy`.

AWS CLI

To retrieve the resource-based policy attached to a secret

The following `get-resource-policy` example retrieves the resource-based policy attached to a secret.

```
aws secretsmanager get-resource-policy \  
  --secret-id MyTestSecret
```

Output:

```
{  
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-  
a1b2c3",  
  "Name": "MyTestSecret",  
  "ResourcePolicy": "{\n  \"Version\": \"2012-10-17\",  
  \"Statement\": [\n    {\n      \"Effect\":  
\\\"Allow\\\",  
      \"Principal\": {\n        \"AWS\": \"arn:aws:iam::123456789012:root\"\n      },  
      \"Action\":  
      \"secretsmanager:GetSecretValue\",  
      \"Resource\": \"*\"\n    }\n  ]\n}"
```

For more information, see [Authentication and access control](#) in the *Secrets Manager User Guide*.

- For API details, see [GetResourcePolicy](#) in *AWS CLI Command Reference*.

get-secret-value

The following code example shows how to use `get-secret-value`.

AWS CLI

Example 1: To retrieve the encrypted secret value of a secret

The following `get-secret-value` example gets the current secret value.

```
aws secretsmanager get-secret-value \  
  --secret-id MyTestSecret
```

```
--secret-id MyTestSecret
```

Output:

```
{
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
  "Name": "MyTestSecret",
  "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "SecretString": "{\"user\":\"diegor\",\"password\":\"EXAMPLE-PASSWORD\"}",
  "VersionStages": [
    "AWSCURRENT"
  ],
  "CreateDate": 1523477145.713
}
```

For more information, see [Retrieve a secret](#) in the *Secrets Manager User Guide*.

Example 2: To retrieve the previous secret value

The following `get-secret-value` example gets the previous secret value.:

```
aws secretsmanager get-secret-value \
  --secret-id MyTestSecret
  --version-stage AWSPREVIOUS
```

Output:

```
{
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
  "Name": "MyTestSecret",
  "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "SecretString": "{\"user\":\"diegor\",\"password\":\"PREVIOUS-EXAMPLE-PASSWORD
\"}",
  "VersionStages": [
    "AWSPREVIOUS"
  ],
  "CreateDate": 1523477145.713
}
```

For more information, see [Retrieve a secret](#) in the *Secrets Manager User Guide*.

- For API details, see [GetSecretValue](#) in *AWS CLI Command Reference*.

list-secret-version-ids

The following code example shows how to use `list-secret-version-ids`.

AWS CLI

To list all of the secret versions associated with a secret

The following `list-secret-version-ids` example gets a list of all of the versions of a secret.

```
aws secretsmanager list-secret-version-ids \  
  --secret-id MyTestSecret
```

Output:

```
{  
  "Versions": [  
    {  
      "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
      "VersionStages": [  
        "AWSPREVIOUS"  
      ],  
      "LastAccessedDate": 1523477145.713,  
      "CreateDate": 1523477145.713  
    },  
    {  
      "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",  
      "VersionStages": [  
        "AWSCURRENT"  
      ],  
      "LastAccessedDate": 1523477145.713,  
      "CreateDate": 1523486221.391  
    },  
    {  
      "CreateDate": 1.51197446236E9,  
      "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333;"  
    }  
  ],  
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-a1b2c3",
```

```
"Name": "MyTestSecret"  
}
```

For more information, see [Version](#) in the *Secrets Manager User Guide*.

- For API details, see [ListSecretVersionIds](#) in *AWS CLI Command Reference*.

list-secrets

The following code example shows how to use `list-secrets`.

AWS CLI

Example 1: To list the secrets in your account

The following `list-secrets` example gets a list of the secrets in your account.

```
aws secretsmanager list-secrets
```

Output:

```
{  
  "SecretList": [  
    {  
      "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-a1b2c3",  
      "Name": "MyTestSecret",  
      "LastChangedDate": 1523477145.729,  
      "SecretVersionsToStages": {  
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111": [  
          "AWSCURRENT"  
        ]  
      }  
    },  
    {  
      "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:AnotherSecret-d4e5f6",  
      "Name": "AnotherSecret",  
      "LastChangedDate": 1523482025.685,  
      "SecretVersionsToStages": {  
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222": [  
          "AWSCURRENT"  
        ]  
      }  
    }  
  ]  
}
```



```
    ]
  }
}
]
```

For more information, see [Find a secret](#) in the *Secrets Manager User Guide*.

Example 2: To filter the list of secrets in your account

The following `list-secrets` example gets a list of the secrets in your account that have `Test` in the name. Filtering by name is case sensitive.

```
aws secretsmanager list-secrets \
  --filter Key="name",Values="Test"
```

Output:

```
{
  "SecretList": [
    {
      "ARN": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:MyTestSecret-a1b2c3",
      "Name": "MyTestSecret",
      "LastChangedDate": 1523477145.729,
      "SecretVersionsToStages": {
        "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111": [
          "AWSCURRENT"
        ]
      }
    }
  ]
}
```

For more information, see [Find a secret](#) in the *Secrets Manager User Guide*.

Example 3: To list the secrets in your account managed by another service

The following `list-secrets` example returns the secrets in your account that are managed by Amazon RDS.

```
aws secretsmanager list-secrets \
```

```
--filter Key="owning-service",Values="rds"
```

Output:

```
{
  "SecretList": [
    {
      "Name": "rds!cluster-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "Tags": [
        {
          "Value": "arn:aws:rds:us-
west-2:123456789012:cluster:database-1",
          "Key": "aws:rds:primaryDBClusterArn"
        },
        {
          "Value": "rds",
          "Key": "aws:secretsmanager:owningService"
        }
      ],
      "RotationRules": {
        "AutomaticallyAfterDays": 1
      },
      "LastChangedDate": 1673477781.275,
      "LastRotatedDate": 1673477781.26,
      "SecretVersionsToStages": {
        "a1b2c3d4-5678-90ab-cdef-EXAMPLEEaaaaa": [
          "AWSPREVIOUS"
        ],
        "a1b2c3d4-5678-90ab-cdef-EXAMPLEebbbbb": [
          "AWSCURRENT",
          "AWSPENDING"
        ]
      },
      "OwningService": "rds",
      "RotationEnabled": true,
      "CreatedDate": 1673467300.7,
      "LastAccessedDate": 1673395200.0,
      "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:rds!
cluster-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111-a1b2c3",
      "Description": "Secret associated with primary RDS DB cluster:
arn:aws:rds:us-west-2:123456789012:cluster:database-1"
    }
  ]
}
```

```
}
```

For more information, see [Secrets managed by other services](#) in the *Secrets Manager User Guide*.

- For API details, see [ListSecrets](#) in *AWS CLI Command Reference*.

put-resource-policy

The following code example shows how to use `put-resource-policy`.

AWS CLI

To add a resource-based policy to a secret

The following `put-resource-policy` example adds a permissions policy to a secret, checking first that the policy does not provide broad access to the secret. The policy is read from a file. For more information, see [Loading AWS CLI parameters from a file](#) in the *AWS CLI User Guide*.

```
aws secretsmanager put-resource-policy \  
  --secret-id MyTestSecret \  
  --resource-policy file://mypolicy.json \  
  --block-public-policy
```

Contents of `mypolicy.json`:

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Principal": {  
        "AWS": "arn:aws:iam::123456789012:role/MyRole"  
      },  
      "Action": "secretsmanager:GetSecretValue",  
      "Resource": "*"   
    }  
  ]  
}
```

Output:

```
{
```

```
"ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
  "Name": "MyTestSecret"
}
```

For more information, see [Attach a permissions policy to a secret](#) in the *Secrets Manager User Guide*.

- For API details, see [PutResourcePolicy](#) in *AWS CLI Command Reference*.

put-secret-value

The following code example shows how to use `put-secret-value`.

AWS CLI

Example 1: To store a new secret value in a secret

The following `put-secret-value` example creates a new version of a secret with two key-value pairs.

```
aws secretsmanager put-secret-value \
  --secret-id MyTestSecret \
  --secret-string "{\"user\": \"diegor\", \"password\": \"EXAMPLE-PASSWORD\"}"
```

Output:

```
{
  "ARN": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:MyTestSecret-1a2b3c",
  "Name": "MyTestSecret",
  "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "VersionStages": [
    "AWSCURRENT"
  ]
}
```

For more information, see [Modify a secret](#) in the *Secrets Manager User Guide*.

Example 2: To store a new secret value from credentials in a JSON file

The following `put-secret-value` example creates a new version of a secret from credentials in a file. For more information, see [Loading AWS CLI parameters from a file](#) in the *AWS CLI User Guide*.

```
aws secretsmanager put-secret-value \  
  --secret-id MyTestSecret \  
  --secret-string file://mycreds.json
```

Contents of `mycreds.json`:

```
{  
  "engine": "mysql",  
  "username": "saanvis",  
  "password": "EXAMPLE-PASSWORD",  
  "host": "my-database-endpoint.us-west-2.rds.amazonaws.com",  
  "dbname": "myDatabase",  
  "port": "3306"  
}
```

Output:

```
{  
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-a1b2c3",  
  "Name": "MyTestSecret",  
  "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
  "VersionStages": [  
    "AWSCURRENT"  
  ]  
}
```

For more information, see [Modify a secret](#) in the *Secrets Manager User Guide*.

- For API details, see [PutSecretValue](#) in *AWS CLI Command Reference*.

remove-regions-from-replication

The following code example shows how to use `remove-regions-from-replication`.

AWS CLI

To delete a replica secret

The following `remove-regions-from-replication` example deletes a replica secret in `eu-west-3`. To delete a primary secret that is replicated to other regions, first delete the replicas and then call `delete-secret`.

```
aws secretsmanager remove-regions-from-replication \  
  --secret-id MyTestSecret \  
  --remove-replica-regions eu-west-3
```

Output:

```
{  
  "ARN": "arn:aws:secretsmanager:us-  
west-2:123456789012:secret:MyTestSecret-1a2b3c",  
  "ReplicationStatus": []  
}
```

For more information, see [Delete a replica secret](#) in the *Secrets Manager User Guide*.

- For API details, see [RemoveRegionsFromReplication](#) in *AWS CLI Command Reference*.

replicate-secret-to-regions

The following code example shows how to use `replicate-secret-to-regions`.

AWS CLI

To replicate a secret to another region

The following `replicate-secret-to-regions` example replicates a secret to `eu-west-3`. The replica is encrypted with the AWS managed key `aws/secretsmanager`.

```
aws secretsmanager replicate-secret-to-regions \  
  --secret-id MyTestSecret \  
  --add-replica-regions Region=eu-west-3
```

Output:

```
{  
  "ARN": "arn:aws:secretsmanager:us-  
west-2:123456789012:secret:MyTestSecret-1a2b3c",  
  "ReplicationStatus": [  
    {  
      "Region": "eu-west-3",  
      "Status": "REPLICATED"  
    }  
  ]  
}
```

```
{
  "Region": "eu-west-3",
  "KmsKeyId": "alias/aws/secretsmanager",
  "Status": "InProgress"
}
]
```

For more information, see [Replicate a secret to another Region](#) in the *Secrets Manager User Guide*.

- For API details, see [ReplicateSecretToRegions](#) in *AWS CLI Command Reference*.

restore-secret

The following code example shows how to use `restore-secret`.

AWS CLI

To restore a previously deleted secret

The following `restore-secret` example restores a secret that was previously scheduled for deletion.

```
aws secretsmanager restore-secret \
  --secret-id MyTestSecret
```

Output:

```
{
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
  "Name": "MyTestSecret"
}
```

For more information, see [Delete a secret](#) in the *Secrets Manager User Guide*.

- For API details, see [RestoreSecret](#) in *AWS CLI Command Reference*.

rotate-secret

The following code example shows how to use `rotate-secret`.

AWS CLI

Example 1: To configure and start automatic rotation for a secret

The following `rotate-secret` example configures and starts automatic rotation for a secret. Secrets Manager rotates the secret once immediately, and then every eight hours in a two hour window. The output shows the `VersionId` of the new secret version created by rotation.

```
aws secretsmanager rotate-secret \  
  --secret-id MyTestDatabaseSecret \  
  --rotation-lambda-arn arn:aws:lambda:us-  
west-2:1234566789012:function:SecretsManagerTestRotationLambda \  
  --rotation-rules "{\"ScheduleExpression\": \"cron(0 8/8 * * ? *)\", \"Duration  
\": \"2h\"}"
```

Output:

```
{  
  "ARN": "aws:arn:secretsmanager:us-  
west-2:123456789012:secret:MyTestDatabaseSecret-a1b2c3",  
  "Name": "MyTestDatabaseSecret",  
  "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"  
}
```

For more information, see [Rotate secrets](#) in the *Secrets Manager User Guide*.

Example 2: To configure and start automatic rotation on a rotation interval

The following `rotate-secret` example configures and starts automatic rotation for a secret. Secrets Manager rotates the secret once immediately, and then every 10 days. The output shows the `VersionId` of the new secret version created by rotation.

```
aws secretsmanager rotate-secret \  
  --secret-id MyTestDatabaseSecret \  
  --rotation-lambda-arn arn:aws:lambda:us-  
west-2:1234566789012:function:SecretsManagerTestRotationLambda \  
  --rotation-rules "{\"ScheduleExpression\": \"rate(10 days)\"}"
```

Output:

```
{
```



```
"ARN": "aws:arn:secretsmanager:us-  
west-2:123456789012:secret:MyTestDatabaseSecret-a1b2c3",  
  "Name": "MyTestDatabaseSecret",  
  "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"  
}
```

For more information, see [Rotate secrets](#) in the *Secrets Manager User Guide*.

Example 3: To rotate a secret immediately

The following `rotate-secret` example starts an immediate rotation. The output shows the `VersionId` of the new secret version created by rotation. The secret must already have rotation configured.

```
aws secretsmanager rotate-secret \  
  --secret-id MyTestDatabaseSecret
```

Output:

```
{  
  "ARN": "aws:arn:secretsmanager:us-  
west-2:123456789012:secret:MyTestDatabaseSecret-a1b2c3",  
  "Name": "MyTestDatabaseSecret",  
  "VersionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"  
}
```

For more information, see [Rotate secrets](#) in the *Secrets Manager User Guide*.

- For API details, see [RotateSecret](#) in *AWS CLI Command Reference*.

stop-replication-to-replica

The following code example shows how to use `stop-replication-to-replica`.

AWS CLI

To promote a replica secret to a primary

The following `stop-replication-to-replica` example removes the link between a replica secret to the primary. The replica secret is promoted to a primary secret in the replica region. You must call `stop-replication-to-replica` from within the replica region.

```
aws secretsmanager stop-replication-to-replica \  
  --secret-id MyTestSecret
```

Output:

```
{  
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-  
a1b2c3"  
}
```

For more information, see [Promote a replica secret](#) in the *Secrets Manager User Guide*.

- For API details, see [StopReplicationToReplica](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use tag-resource.

AWS CLI

Example 1: To add a tag to a secret

The following example shows how to attach a tag with shorthand syntax.

```
aws secretsmanager tag-resource \  
  --secret-id MyTestSecret \  
  --tags Key=FirstTag,Value=FirstValue
```

This command produces no output.

For more information, see [Tag your secrets](#) in the *Secrets Manager User Guide*.

Example 2: To add multiple tags to a secret

The following tag-resource example attaches two key-value tags to a secret.

```
aws secretsmanager tag-resource \  
  --secret-id MyTestSecret \  
  --tags '[{"Key": "FirstTag", "Value": "FirstValue"}, {"Key": "SecondTag",  
"Value": "SecondValue"}]'
```

This command produces no output.

For more information, see [Tag secrets](#) in the *Secrets Manager User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from a secret

The following `untag-resource` example removes two tags from a secret. For each tag, both key and value are removed.

```
aws secretsmanager untag-resource \  
  --secret-id MyTestSecret \  
  --tag-keys '[ "FirstTag", "SecondTag"]'
```

This command produces no output.

For more information, see [Tag secrets](#) in the *Secrets Manager User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-secret-version-stage

The following code example shows how to use `update-secret-version-stage`.

AWS CLI

Example 1: To revert a secret to the previous version

The following `update-secret-version-stage` example moves the `AWSCURRENT` staging label to the previous version of a secret, which reverts the secret to the previous version. To find the ID for the previous version, use `list-secret-version-ids`. For this example, the version with the `AWSCURRENT` label is `a1b2c3d4-5678-90ab-cdef-EXAMPLE11111` and the version with the `AWSPREVIOUS` label is `a1b2c3d4-5678-90ab-cdef-EXAMPLE22222`. In this example, you move the `AWSCURRENT` label from version 11111 to 22222. Because the `AWSCURRENT`

label is removed from a version, `update-secret-version-stage` automatically moves the `AWSPREVIOUS` label to that version (11111). The effect is that the `AWSCURRENT` and `AWSPREVIOUS` versions are swapped.

```
aws secretsmanager update-secret-version-stage \  
  --secret-id MyTestSecret \  
  --version-stage AWSCURRENT \  
  --move-to-version-id a1b2c3d4-5678-90ab-cdef-EXAMPLE22222 \  
  --remove-from-version-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{  
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-  
a1b2c3",  
  "Name": "MyTestSecret"  
}
```

For more information, see [Version](#) in the *Secrets Manager User Guide*.

Example 2: To add a staging label attached to a version of a secret

The following `update-secret-version-stage` example adds a staging label to a version of a secret. You can review the results by running `list-secret-version-ids` and viewing the `VersionStages` response field for the affected version.

```
aws secretsmanager update-secret-version-stage \  
  --secret-id MyTestSecret \  
  --version-stage STAGINGLABEL1 \  
  --move-to-version-id EXAMPLE1-90ab-cdef-fedc-ba987EXAMPLE
```

Output:

```
{  
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-  
a1b2c3",  
  "Name": "MyTestSecret"  
}
```

For more information, see [Version](#) in the *Secrets Manager User Guide*.

Example 3: To delete a staging label attached to a version of a secret

The following `update-secret-version-stage` example deletes a staging label that is attached to a version of a secret. You can review the results by running `list-secret-version-ids` and viewing the `VersionStages` response field for the affected version.

```
aws secretsmanager update-secret-version-stage \  
  --secret-id MyTestSecret \  
  --version-stage STAGINGLABEL1 \  
  --remove-from-version-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{  
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-  
a1b2c3",  
  "Name": "MyTestSecret"  
}
```

For more information, see [Version](#) in the *Secrets Manager User Guide*.

- For API details, see [UpdateSecretVersionStage](#) in *AWS CLI Command Reference*.

update-secret

The following code example shows how to use `update-secret`.

AWS CLI

Example 1: To update the description of a secret

The following `update-secret` example updates the description of a secret.

```
aws secretsmanager update-secret \  
  --secret-id MyTestSecret \  
  --description "This is a new description for the secret."
```

Output:

```
{
```

```
"ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
  "Name": "MyTestSecret"
}
```

For more information, see [Modify a secret](#) in the *Secrets Manager User Guide*.

Example 2: To update the encryption key associated with a secret

The following `update-secret` example updates the KMS key used to encrypt the secret value. The KMS key must be in the same region as the secret.

```
aws secretsmanager update-secret \
  --secret-id MyTestSecret \
  --kms-key-id arn:aws:kms:us-west-2:123456789012:key/EXAMPLE1-90ab-cdef-fedc-
ba987EXAMPLE
```

Output:

```
{
  "ARN": "arn:aws:secretsmanager:us-west-2:123456789012:secret:MyTestSecret-
a1b2c3",
  "Name": "MyTestSecret"
}
```

For more information, see [Modify a secret](#) in the *Secrets Manager User Guide*.

- For API details, see [UpdateSecret](#) in *AWS CLI Command Reference*.

validate-resource-policy

The following code example shows how to use `validate-resource-policy`.

AWS CLI

To validate a resource policy

The following `validate-resource-policy` example checks that a resource policy doesn't grant broad access to a secret. The policy is read from a file on disk. For more information, see [Loading AWS CLI parameters from a file](#) in the *AWS CLI User Guide*.

```
aws secretsmanager validate-resource-policy \
```

```
--resource-policy file://mypolicy.json
```

Contents of `mypolicy.json`:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:role/MyRole"
      },
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "*"
    }
  ]
}
```

Output:

```
{
  "PolicyValidationPassed": true,
  "ValidationErrors": []
}
```

For more information, see [Permissions reference for Secrets Manager](#) in the *Secrets Manager User Guide*.

- For API details, see [ValidateResourcePolicy](#) in *AWS CLI Command Reference*.

Security Hub examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Security Hub.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

accept-administrator-invitation

The following code example shows how to use `accept-administrator-invitation`.

AWS CLI

To accept an invitation from an administrator account

The following `accept-administrator-invitation` example accepts the specified invitation from the specified administrator account.

```
aws securityhub accept-invitation \  
  --administrator-id 123456789012 \  
  --invitation-id 7ab938c5d52d7904ad09f9e7c20cc4eb
```

This command produces no output.

For more information, see [Managing administrator and member accounts](#) in the *AWS Security Hub User Guide*.

- For API details, see [AcceptAdministratorInvitation](#) in *AWS CLI Command Reference*.

accept-invitation

The following code example shows how to use `accept-invitation`.

AWS CLI

To accept an invitation from an administrator account

The following `accept-invitation` example accepts the specified invitation from the specified administrator account.


```
aws securityhub accept-invitation \  
  --master-id 123456789012 \  
  --invitation-id 7ab938c5d52d7904ad09f9e7c20cc4eb
```

This command produces no output.

For more information, see [Managing administrator and member accounts](#) in the *AWS Security Hub User Guide*.

- For API details, see [AcceptInvitation](#) in *AWS CLI Command Reference*.

batch-delete-automation-rules

The following code example shows how to use `batch-delete-automation-rules`.

AWS CLI

To delete automation rules

The following `batch-delete-automation-rules` example deletes the specified automation rule. You can delete one or more rules with a single command. Only the Security Hub administrator account can run this command.

```
aws securityhub batch-delete-automation-rules \  
  --automation-rules-arns '["arn:aws:securityhub:us-  
east-1:123456789012:automation-rule/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"]'
```

Output:

```
{  
  "ProcessedAutomationRules": [  
    "arn:aws:securityhub:us-east-1:123456789012:automation-rule/  
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"  
  ],  
  "UnprocessedAutomationRules": []  
}
```

For more information, see [Deleting automation rules](#) in the *AWS Security Hub User Guide*.

- For API details, see [BatchDeleteAutomationRules](#) in *AWS CLI Command Reference*.

batch-disable-standards

The following code example shows how to use batch-disable-standards.

AWS CLI

To disable a standard

The following batch-disable-standards example disables the standard associated with the specified subscription ARN.

```
aws securityhub batch-disable-standards \  
  --standards-subscription-arns "arn:aws:securityhub:us-  
west-1:123456789012:subscription/pci-dss/v/3.2.1"
```

Output:

```
{  
  "StandardsSubscriptions": [  
    {  
      "StandardsArn": "arn:aws:securityhub:eu-central-1::standards/pci-dss/  
v/3.2.1",  
      "StandardsInput": { },  
      "StandardsStatus": "DELETING",  
      "StandardsSubscriptionArn": "arn:aws:securityhub:us-  
west-1:123456789012:subscription/pci-dss/v/3.2.1"  
    }  
  ]  
}
```

For more information, see [Disabling or enabling a security standard](#) in the *AWS Security Hub User Guide*.

- For API details, see [BatchDisableStandards](#) in *AWS CLI Command Reference*.

batch-enable-standards

The following code example shows how to use batch-enable-standards.

AWS CLI

To enable a standard

The following `batch-enable-standards` example enables the PCI DSS standard for the requesting account.

```
aws securityhub batch-enable-standards \  
  --standards-subscription-requests '{"StandardsArn":"arn:aws:securityhub:us-  
west-1::standards/pci-dss/v/3.2.1"}'
```

Output:

```
{  
  "StandardsSubscriptions": [  
    {  
      "StandardsArn": "arn:aws:securityhub:us-west-1::standards/pci-dss/  
v/3.2.1",  
      "StandardsInput": { },  
      "StandardsStatus": "PENDING",  
      "StandardsSubscriptionArn": "arn:aws:securityhub:us-  
west-1:123456789012:subscription/pci-dss/v/3.2.1"  
    }  
  ]  
}
```

For more information, see [Disabling or enabling a security standard](#) in the *AWS Security Hub User Guide*.

- For API details, see [BatchEnableStandards](#) in *AWS CLI Command Reference*.

batch-get-automation-rules

The following code example shows how to use `batch-get-automation-rules`.

AWS CLI

To get details for automation rules

The following `batch-get-automation-rules` example gets details for the specified automation rule. You can get details for one or more automation rules with a single command.

```
aws securityhub batch-get-automation-rules \  
  --automation-rules-arns '['arn:aws:securityhub:us-  
east-1:123456789012:automation-rule/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"]'
```

Output:

```
{
  "Rules": [
    {
      "RuleArn": "arn:aws:securityhub:us-east-1:123456789012:automation-rule/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "RuleStatus": "ENABLED",
      "RuleOrder": 1,
      "RuleName": "Suppress informational findings",
      "Description": "Suppress GuardDuty findings with Informational
severity",
      "IsTerminal": false,
      "Criteria": {
        "ProductName": [
          {
            "Value": "GuardDuty",
            "Comparison": "EQUALS"
          }
        ],
        "SeverityLabel": [
          {
            "Value": "INFORMATIONAL",
            "Comparison": "EQUALS"
          }
        ],
        "WorkflowStatus": [
          {
            "Value": "NEW",
            "Comparison": "EQUALS"
          }
        ],
        "RecordState": [
          {
            "Value": "ACTIVE",
            "Comparison": "EQUALS"
          }
        ]
      },
      "Actions": [
        {
          "Type": "FINDING_FIELDS_UPDATE",
          "FindingFieldsUpdate": {
            "Note": {
```

```

        "Text": "Automatically suppress GuardDuty findings with
Informational severity",
        "UpdatedBy": "sechub-automation"
    },
    "Workflow": {
        "Status": "SUPPRESSED"
    }
}
],
"CreatedAt": "2023-05-31T17:56:14.837000+00:00",
"UpdatedAt": "2023-05-31T17:59:38.466000+00:00",
"CreatedBy": "arn:aws:iam::123456789012:role/Admin"
}
],
"UnprocessedAutomationRules": []
}

```

For more information, see [Viewing automation rules](#) in the *AWS Security Hub User Guide*.

- For API details, see [BatchGetAutomationRules](#) in *AWS CLI Command Reference*.

batch-get-configuration-policy-associations

The following code example shows how to use `batch-get-configuration-policy-associations`.

AWS CLI

To get configuration association details for a batch of targets

The following `batch-get-configuration-policy-associations` example retrieves association details for the specified targets. You can provide account IDs, organizational unit IDs, or the root ID for the target.

```
aws securityhub batch-get-configuration-policy-associations \
  --target '{"OrganizationalUnitId": "ou-6hi7-8j91k12m"}
```

Output:

```
{
```

```

"ConfigurationPolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
"TargetId": "ou-6hi7-8j91kl2m",
"TargetType": "ORGANIZATIONAL_UNIT",
"AssociationType": "APPLIED",
"UpdatedAt": "2023-09-26T21:13:01.816000+00:00",
"AssociationStatus": "SUCCESS",
"AssociationStatusMessage": "Association applied successfully on this target."
}

```

For more information, see [Viewing Security Hub configuration policies](#) in the *AWS Security Hub User Guide*.

- For API details, see [BatchGetConfigurationPolicyAssociations](#) in *AWS CLI Command Reference*.

batch-get-security-controls

The following code example shows how to use `batch-get-security-controls`.

AWS CLI

To get security control details

The following `batch-get-security-controls` example gets details for the security controls ACM.1 and IAM.1 in the current AWS account and AWS Region.

```

aws securityhub batch-get-security-controls \
  --security-control-ids '["ACM.1", "IAM.1"]'

```

Output:

```

{
  "SecurityControls": [
    {
      "SecurityControlId": "ACM.1",
      "SecurityControlArn": "arn:aws:securityhub:us-east-2:123456789012:security-control/ACM.1",
      "Title": "Imported and ACM-issued certificates should be renewed after a specified time period",
      "Description": "This control checks whether an AWS Certificate Manager (ACM) certificate is renewed within the specified time period. It checks both imported certificates and certificates provided by ACM. The control fails if the

```

```

certificate isn't renewed within the specified time period. Unless you provide a
custom parameter value for the renewal period, Security Hub uses a default value of
30 days.",
    "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/
ACM.1/remediation",
    "SeverityRating": "MEDIUM",
    "SecurityControlStatus": "ENABLED"
    "UpdateStatus": "READY",
    "Parameters": {
        "daysToExpiration": {
            "ValueType": CUSTOM,
            "Value": {
                "Integer": 15
            }
        }
    },
    "LastUpdateReason": "Updated control parameter"
},
{
    "SecurityControlId": "IAM.1",
    "SecurityControlArn": "arn:aws:securityhub:us-
east-2:123456789012:security-control/IAM.1",
    "Title": "IAM policies should not allow full \"*\" administrative
privileges",
    "Description": "This AWS control checks whether the default version of
AWS Identity and Access Management (IAM) policies (also known as customer managed
policies) do not have administrator access with a statement that has \"Effect\":
\"Allow\" with \"Action\": \"*\" over \"Resource\": \"*\". It only checks for
the Customer Managed Policies that you created, but not inline and AWS Managed
Policies.",
    "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/
IAM.1/remediation",
    "SeverityRating": "HIGH",
    "SecurityControlStatus": "ENABLED"
    "UpdateStatus": "READY",
    "Parameters": {}
}
]
}

```

For more information, see [Viewing details for a control](#) in the *AWS Security Hub User Guide*.

- For API details, see [BatchGetSecurityControls](#) in *AWS CLI Command Reference*.

batch-get-standards-control-associations

The following code example shows how to use `batch-get-standards-control-associations`.

AWS CLI

To get the enablement status of a control

The following `batch-get-standards-control-associations` example identifies whether the specified controls are enabled in the specified standards.

```
aws securityhub batch-get-standards-control-associations \
  --standards-control-association-ids '[{"SecurityControlId":
  "Config.1","StandardsArn": "arn:aws:securityhub:us-east-1:123456789012:ruleset/cis-
  aws-foundations-benchmark/v/1.2.0"}, {"SecurityControlId": "IAM.6","StandardsArn":
  "arn:aws:securityhub:us-east-1:123456789012:standards/aws-foundational-security-
  best-practices/v/1.0.0"}]'
```

Output:

```
{
  "StandardsControlAssociationDetails": [
    {
      "StandardsArn": "arn:aws:securityhub:::ruleset/cis-aws-foundations-
      benchmark/v/1.2.0",
      "SecurityControlId": "Config.1",
      "SecurityControlArn": "arn:aws:securityhub:us-
      east-1:068873283051:security-control/Config.1",
      "AssociationStatus": "ENABLED",
      "RelatedRequirements": [
        "CIS AWS Foundations 2.5"
      ],
      "UpdatedAt": "2022-10-27T16:07:12.960000+00:00",
      "StandardsControlTitle": "Ensure AWS Config is enabled",
      "StandardsControlDescription": "AWS Config is a web service that
      performs configuration management of supported AWS resources within your account
      and delivers log files to you. The recorded information includes the configuration
      item (AWS resource), relationships between configuration items (AWS resources), and
      any configuration changes between resources. It is recommended to enable AWS Config
      in all regions.",
      "StandardsControlArns": [
```



```

        "arn:aws:securityhub:us-east-1:068873283051:control/cis-aws-
foundations-benchmark/v/1.2.0/2.5"
    ]
  },
  {
    "StandardsArn": "arn:aws:securityhub:us-east-1::standards/aws-
foundational-security-best-practices/v/1.0.0",
    "SecurityControlId": "IAM.6",
    "SecurityControlArn": "arn:aws:securityhub:us-
east-1:068873283051:security-control/IAM.6",
    "AssociationStatus": "DISABLED",
    "RelatedRequirements": [],
    "UpdatedAt": "2022-11-22T21:30:35.080000+00:00",
    "UpdatedReason": "test",
    "StandardsControlTitle": "Hardware MFA should be enabled for the root
user",
    "StandardsControlDescription": "This AWS control checks whether your AWS
account is enabled to use a hardware multi-factor authentication (MFA) device to
sign in with root user credentials.",
    "StandardsControlArns": [
      "arn:aws:securityhub:us-east-1:068873283051:control/aws-
foundational-security-best-practices/v/1.0.0/IAM.6"
    ]
  }
]
}

```

For more information, see [Enabling and disabling controls in specific standards](#) in the *AWS Security Hub User Guide*.

- For API details, see [BatchGetStandardsControlAssociations](#) in *AWS CLI Command Reference*.

batch-import-findings

The following code example shows how to use `batch-import-findings`.

AWS CLI

To update a finding

The following `batch-import-findings` example updates a finding.

```
aws securityhub batch-import-findings \
```

```

--findings '
  [{
    "AwsAccountId": "123456789012",
    "CreatedAt": "2020-05-27T17:05:54.832Z",
    "Description": "Vulnerability in a CloudTrail trail",
    "FindingProviderFields": {
      "Severity": {
        "Label": "LOW",
        "Original": "10"
      },
      "Types": [
        "Software and Configuration Checks/Vulnerabilities/CVE"
      ]
    },
    "GeneratorId": "TestGeneratorId",
    "Id": "Id1",
    "ProductArn": "arn:aws:securityhub:us-
west-1:123456789012:product/123456789012/default",
    "Resources": [
      {
        "Id": "arn:aws:cloudtrail:us-west-1:123456789012:trail/
TrailName",
        "Partition": "aws",
        "Region": "us-west-1",
        "Type": "AwsCloudTrailTrail"
      }
    ],
    "SchemaVersion": "2018-10-08",
    "Title": "CloudTrail trail vulnerability",
    "UpdatedAt": "2020-06-02T16:05:54.832Z"
  ]]'

```

Output:

```

{
  "FailedCount": 0,
  "SuccessCount": 1,
  "FailedFindings": []
}

```

For more information, see [Using BatchImportFindings to create and update findings](#) in the *AWS Security Hub User Guide*.

- For API details, see [BatchImportFindings](#) in *AWS CLI Command Reference*.

batch-update-automation-rules

The following code example shows how to use `batch-update-automation-rules`.

AWS CLI

To update automation rules

The following `batch-update-automation-rules` example updates the specified automation rule. You can update one or more rules with a single command. Only the Security Hub administrator account can run this command.

```
aws securityhub batch-update-automation-rules \
  --update-automation-rules-request-items '[ \
    { \
      "Actions": [{ \
        "Type": "FINDING_FIELDS_UPDATE", \
        "FindingFieldsUpdate": { \
          "Note": { \
            "Text": "Known issue that is a risk", \
            "UpdatedBy": "sechub-automation" \
          }, \
          "Workflow": { \
            "Status": "NEW" \
          } \
        } \
      }], \
      "Criteria": { \
        "SeverityLabel": [{ \
          "Value": "LOW", \
          "Comparison": "EQUALS" \
        }] \
      }, \
      "RuleArn": "arn:aws:securityhub:us-east-1:123456789012:automation-rule/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111", \
      "RuleOrder": 1, \
      "RuleStatus": "DISABLED" \
    } \
  ]'
```

Output:

```
{
  "ProcessedAutomationRules": [
    "arn:aws:securityhub:us-east-1:123456789012:automation-rule/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
  ],
  "UnprocessedAutomationRules": []
}
```

For more information, see [Editing automation rules](#) in the *AWS Security Hub User Guide*.

- For API details, see [BatchUpdateAutomationRules](#) in *AWS CLI Command Reference*.

batch-update-findings

The following code example shows how to use `batch-update-findings`.

AWS CLI**Example 1: To update a finding**

The following `batch-update-findings` example updates two findings to add a note, change the severity label, and resolve it.

```
aws securityhub batch-update-findings \
  --finding-identifiers '[{"Id": "arn:aws:securityhub:us-
west-1:123456789012:subscription/pci-dss/v/3.2.1/PCI.Lambda.2/finding/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111", "ProductArn": "arn:aws:securityhub:us-
west-1::product/aws/securityhub"}, {"Id": "arn:aws:securityhub:us-
west-1:123456789012:subscription/pci-dss/v/3.2.1/PCI.Lambda.2/finding/
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222", "ProductArn": "arn:aws:securityhub:us-
west-1::product/aws/securityhub"}]' \
  --note '{"Text": "Known issue that is not a risk.", "UpdatedBy": "user1"}' \
  --severity '{"Label": "LOW"}' \
  --workflow '{"Status": "RESOLVED"}'
```

Output:

```
{
  "ProcessedFindings": [
    {
```

```

        "Id": "arn:aws:securityhub:us-west-1:123456789012:subscription/pci-dss/v/3.2.1/PCI.Lambda.2/finding/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "ProductArn": "arn:aws:securityhub:us-west-1::product/aws/securityhub"
    },
    {
        "Id": "arn:aws:securityhub:us-west-1:123456789012:subscription/pci-dss/v/3.2.1/PCI.Lambda.2/finding/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "ProductArn": "arn:aws:securityhub:us-west-1::product/aws/securityhub"
    }
],
"UnprocessedFindings": []
}

```

For more information, see [Using BatchUpdateFindings to update a finding](#) in the *AWS Security Hub User Guide*.

Example 2: To update a finding using shorthand syntax

The following `batch-update-findings` example updates two findings to add a note, change the severity label, and resolve it using shorthand syntax.

```

aws securityhub batch-update-findings \
  --finding-identifiers Id="arn:aws:securityhub:us-west-1:123456789012:subscription/pci-dss/v/3.2.1/PCI.Lambda.2/finding/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",ProductArn="arn:aws:securityhub:us-west-1::product/aws/securityhub" Id="arn:aws:securityhub:us-west-1:123456789012:subscription/pci-dss/v/3.2.1/PCI.Lambda.2/finding/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",ProductArn="arn:aws:securityhub:us-west-1::product/aws/securityhub" \
  --note Text="Known issue that is not a risk.",UpdatedBy="user1" \
  --severity Label="LOW" \
  --workflow Status="RESOLVED"

```

Output:

```

{
  "ProcessedFindings": [
    {
      "Id": "arn:aws:securityhub:us-west-1:123456789012:subscription/pci-dss/v/3.2.1/PCI.Lambda.2/finding/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "ProductArn": "arn:aws:securityhub:us-west-1::product/aws/securityhub"
    },

```

```

    {
      "Id": "arn:aws:securityhub:us-west-1:123456789012:subscription/pci-dss/
v/3.2.1/PCI.Lambda.2/finding/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
      "ProductArn": "arn:aws:securityhub:us-west-1::product/aws/securityhub"
    }
  ],
  "UnprocessedFindings": []
}

```

For more information, see [Using BatchUpdateFindings to update a finding](#) in the *AWS Security Hub User Guide*.

- For API details, see [BatchUpdateFindings](#) in *AWS CLI Command Reference*.

batch-update-standards-control-associations

The following code example shows how to use `batch-update-standards-control-associations`.

AWS CLI

To update the enablement status of a control in enabled standards

The following `batch-update-standards-control-associations` example disables `CloudTrail.1` in the specified standards.

```

aws securityhub batch-update-standards-control-associations \
  --standards-control-association-updates '[{"SecurityControlId": "CloudTrail.1",
"StandardsArn": "arn:aws:securityhub::ruleset/cis-aws-foundations-benchmark/
v/1.2.0", "AssociationStatus": "DISABLED", "UpdatedReason": "Not applicable
to environment"}, {"SecurityControlId": "CloudTrail.1", "StandardsArn":
"arn:aws:securityhub::standards/cis-aws-foundations-benchmark/v/1.4.0",
"AssociationStatus": "DISABLED", "UpdatedReason": "Not applicable to
environment"}]'

```

This command produces no output when successful.

For more information, see [Enabling and disabling controls in specific standards](#) and [Enabling and disabling controls in all standards](#) in the *AWS Security Hub User Guide*.

- For API details, see [BatchUpdateStandardsControlAssociations](#) in *AWS CLI Command Reference*.

create-action-target

The following code example shows how to use `create-action-target`.

AWS CLI

To create a custom action

The following `create-action-target` example creates a custom action. It provides the name, description, and identifier for the action.

```
aws securityhub create-action-target \  
  --name "Send to remediation" \  
  --description "Action to send the finding for remediation tracking" \  
  --id "Remediation"
```

Output:

```
{  
  "ActionTargetArn": "arn:aws:securityhub:us-west-1:123456789012:action/custom/  
Remediation"  
}
```

For more information, see [Creating a custom action and associating it with a CloudWatch Events rule](#) in the *AWS Security Hub User Guide*.

- For API details, see [CreateActionTarget](#) in *AWS CLI Command Reference*.

create-automation-rule

The following code example shows how to use `create-automation-rule`.

AWS CLI

To create an automation rule

The following `create-automation-rule` example creates an automation rule in the current AWS account and AWS Region. Security Hub filters your findings based on the specified criteria and applies the actions to matching findings. Only the Security Hub administrator account can run this command.

```
aws securityhub create-automation-rule \  

```

```

--actions '[{ \
  "Type": "FINDING_FIELDS_UPDATE", \
  "FindingFieldsUpdate": { \
    "Severity": { \
      "Label": "HIGH" \
    }, \
    "Note": { \
      "Text": "Known issue that is a risk. Updated by automation rules", \
      "UpdatedBy": "sechub-automation" \
    } \
  } \
}]' \
--criteria '{ \
  "SeverityLabel": [{ \
    "Value": "INFORMATIONAL", \
    "Comparison": "EQUALS" \
  }] \
}' \
--description "A sample rule" \
--no-is-terminal \
--rule-name "sample rule" \
--rule-order 1 \
--rule-status "ENABLED"

```

Output:

```

{
  "RuleArn": "arn:aws:securityhub:us-east-1:123456789012:automation-rule/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}

```

For more information, see [Creating automation rules](#) in the *AWS Security Hub User Guide*.

- For API details, see [CreateAutomationRule](#) in *AWS CLI Command Reference*.

create-configuration-policy

The following code example shows how to use create-configuration-policy.

AWS CLI

To create a configuration policy

The following `create-configuration-policy` example creates a configuration policy with the specified settings.

```
aws securityhub create-configuration-policy \
  --name "SampleConfigurationPolicy" \
  --description "SampleDescription" \
  --configuration-policy '{"SecurityHub": {"ServiceEnabled":
true, "EnabledStandardIdentifiers": ["arn:aws:securityhub:eu-
central-1::standards/aws-foundational-security-best-practices/
v/1.0.0", "arn:aws:securityhub:::ruleset/cis-aws-foundations-benchmark/
v/1.2.0"], "SecurityControlsConfiguration": {"DisabledSecurityControlIdentifiers":
["CloudTrail.2"], "SecurityControlCustomParameters": [{"SecurityControlId":
"ACM.1", "Parameters": {"daysToExpiration": {"ValueType": "CUSTOM", "Value":
{"Integer": 15}}}]}}}' \
  --tags '{"Environment": "Prod"}'
```

Output:

```
{
  "Arn": "arn:aws:securityhub:eu-central-1:123456789012:configuration-policy/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "Name": "SampleConfigurationPolicy",
  "Description": "SampleDescription",
  "UpdatedAt": "2023-11-28T20:28:04.494000+00:00",
  "CreatedAt": "2023-11-28T20:28:04.494000+00:00",
  "ConfigurationPolicy": {
    "SecurityHub": {
      "ServiceEnabled": true,
      "EnabledStandardIdentifiers": [
        "arn:aws:securityhub:eu-central-1::standards/aws-foundational-
security-best-practices/v/1.0.0",
        "arn:aws:securityhub:::ruleset/cis-aws-foundations-benchmark/
v/1.2.0"
      ],
      "SecurityControlsConfiguration": {
        "DisabledSecurityControlIdentifiers": [
          "CloudTrail.2"
        ],
        "SecurityControlCustomParameters": [
          {
            "SecurityControlId": "ACM.1",
            "Parameters": {
```

```
    "daysToExpiration": {
      "ValueType": "CUSTOM",
      "Value": {
        "Integer": 15
      }
    }
  }
}
]
```

For more information, see [Creating and associating Security Hub configuration policies](#) in the *AWS Security Hub User Guide*.

- For API details, see [CreateConfigurationPolicy](#) in *AWS CLI Command Reference*.

create-finding-aggregator

The following code example shows how to use `create-finding-aggregator`.

AWS CLI

To enable finding aggregation

The following `create-finding-aggregator` example configures finding aggregation. It is run from US East (Virginia), which designates US East (Virginia) as the aggregation Region. It indicates to only link specified Regions, and to not automatically link new Regions. It selects US West (N. California) and US West (Oregon) as the linked Regions.

```
aws securityhub create-finding-aggregator \
  --region us-east-1 \
  --region-linking-mode SPECIFIED_REGIONS \
  --regions us-west-1,us-west-2
```

Output:

```
{
  "FindingAggregatorArn": "arn:aws:securityhub:us-east-1:222222222222:finding-
  aggregator/123e4567-e89b-12d3-a456-426652340000",
```

```
"FindingAggregationRegion": "us-east-1",
"RegionLinkingMode": "SPECIFIED_REGIONS",
"Regions": "us-west-1,us-west-2"
}
```

For more information, see [Enabling finding aggregation](#) in the *AWS Security Hub User Guide*.

- For API details, see [CreateFindingAggregator](#) in *AWS CLI Command Reference*.

create-insight

The following code example shows how to use `create-insight`.

AWS CLI

To create a custom insight

The following `create-insight` example creates a custom insight named `Critical role findings` that returns critical findings that are related to AWS roles.

```
aws securityhub create-insight \
  --filters '{"ResourceType": [{ "Comparison": "EQUALS", "Value": "AwsIamRole"}],
  "SeverityLabel": [{"Comparison": "EQUALS", "Value": "CRITICAL"}]}' \
  --group-by-attribute "ResourceId" \
  --name "Critical role findings"
```

Output:

```
{
  "InsightArn": "arn:aws:securityhub:us-west-1:123456789012:insight/123456789012/
  custom/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

For more information, see [Managing custom insights](#) in the *AWS Security Hub User Guide*.

- For API details, see [CreateInsight](#) in *AWS CLI Command Reference*.

create-members

The following code example shows how to use `create-members`.

AWS CLI

To add accounts as member accounts

The following `create-members` example adds two accounts as member accounts to the requesting administrator account.

```
aws securityhub create-members \  
  --account-details '[{"AccountId": "123456789111"}, {"AccountId":  
  "123456789222"}]'
```

Output:

```
{  
  "UnprocessedAccounts": []  
}
```

For more information, see [Managing administrator and member accounts](#) in the *AWS Security Hub User Guide*.

- For API details, see [CreateMembers](#) in *AWS CLI Command Reference*.

decline-invitations

The following code example shows how to use `decline-invitations`.

AWS CLI

To decline an invitation to be a member account

The following `decline-invitations` example declines an invitation to be a member account of the specified administrator account. The member account is the requesting account.

```
aws securityhub decline-invitations \  
  --account-ids "123456789012"
```

Output:

```
{  
  "UnprocessedAccounts": []  
}
```

For more information, see [Managing administrator and member accounts](#) in the *AWS Security Hub User Guide*.

- For API details, see [DeclineInvitations](#) in *AWS CLI Command Reference*.

delete-action-target

The following code example shows how to use `delete-action-target`.

AWS CLI

To delete a custom action

The following `delete-action-target` example deletes the custom action identified by the specified ARN.

```
aws securityhub delete-action-target \  
    --action-target-arn "arn:aws:securityhub:us-west-1:123456789012:action/custom/  
Remediation"
```

Output:

```
{  
  "ActionTargetArn": "arn:aws:securityhub:us-west-1:123456789012:action/custom/  
Remediation"  
}
```

For more information, see [Creating a custom action and associating it with a CloudWatch Events rule](#) in the *AWS Security Hub User Guide*.

- For API details, see [DeleteActionTarget](#) in *AWS CLI Command Reference*.

delete-configuration-policy

The following code example shows how to use `delete-configuration-policy`.

AWS CLI

To delete a configuration policy

The following `delete-configuration-policy` example deletes the specified configuration policy.

```
aws securityhub delete-configuration-policy \  
  --identifier "arn:aws:securityhub:eu-central-1:123456789012:configuration-  
policy/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

This command produces no output.

For more information, see [Deleting and disassociating Security Hub configuration policies](#) in the *AWS Security Hub User Guide*.

- For API details, see [DeleteConfigurationPolicy](#) in *AWS CLI Command Reference*.

delete-finding-aggregator

The following code example shows how to use delete-finding-aggregator.

AWS CLI

To stop finding aggregation

The following delete-finding-aggregator example stops finding aggregation. It is run from US East (Virginia), which is the aggregation Region.

```
aws securityhub delete-finding-aggregator \  
  --region us-east-1 \  
  --finding-aggregator-arn arn:aws:securityhub:us-east-1:222222222222:finding-  
aggregator/123e4567-e89b-12d3-a456-426652340000
```

This command produces no output.

For more information, see [Stopping finding aggregation](#) in the *AWS Security Hub User Guide*.

- For API details, see [DeleteFindingAggregator](#) in *AWS CLI Command Reference*.

delete-insight

The following code example shows how to use delete-insight.

AWS CLI

To delete a custom insight

The following `delete-insight` example deletes the custom insight with the specified ARN.

```
aws securityhub delete-insight \  
  --insight-arn "arn:aws:securityhub:us-west-1:123456789012:insight/123456789012/  
custom/a1b2c3d4-5678-90ab-cdef-EXAMPLE111111"
```

Output:

```
{  
  "InsightArn": "arn:aws:securityhub:eu-  
central-1:123456789012:insight/123456789012/custom/a1b2c3d4-5678-90ab-cdef-  
EXAMPLE111111"  
}
```

For more information, see [Managing custom insights](#) in the *AWS Security Hub User Guide*.

- For API details, see [DeleteInsight](#) in *AWS CLI Command Reference*.

delete-invitations

The following code example shows how to use `delete-invitations`.

AWS CLI

To delete an invitation to be a member account

The following `delete-invitations` example deletes an invitation to be a member account for the specified administrator account. The member account is the requesting account.

```
aws securityhub delete-invitations \  
  --account-ids "123456789012"
```

Output:

```
{  
  "UnprocessedAccounts": []  
}
```

For more information, see [Managing administrator and member accounts](#) in the *AWS Security Hub User Guide*.

- For API details, see [DeleteInvitations](#) in *AWS CLI Command Reference*.

delete-members

The following code example shows how to use delete-members.

AWS CLI

To delete member accounts

The following delete-members example deletes the specified member accounts from the requesting administrator account.

```
aws securityhub delete-members \  
  --account-ids "123456789111" "123456789222"
```

Output:

```
{  
  "UnprocessedAccounts": []  
}
```

For more information, see [Managing administrator and member accounts](#) in the *AWS Security Hub User Guide*.

- For API details, see [DeleteMembers](#) in *AWS CLI Command Reference*.

describe-action-targets

The following code example shows how to use describe-action-targets.

AWS CLI

To retrieve details about custom actions

The following describe-action-targets example retrieves information about the custom action identified by the specified ARN.

```
aws securityhub describe-action-targets \  
  --action-target-arns "arn:aws:securityhub:us-west-1:123456789012:action/custom/  
  Remediation"
```


Output:

```
{
  "ActionTargets": [
    {
      "ActionTargetArn": "arn:aws:securityhub:us-west-1:123456789012:action/
custom/Remediation",
      "Description": "Action to send the finding for remediation tracking",
      "Name": "Send to remediation"
    }
  ]
}
```

For more information, see [Creating a custom action and associating it with a CloudWatch Events rule](#) in the *AWS Security Hub User Guide*.

- For API details, see [DescribeActionTargets](#) in *AWS CLI Command Reference*.

describe-hub

The following code example shows how to use `describe-hub`.

AWS CLI**To get information about a hub resource**

The following `describe-hub` example returns the subscription date for the specified hub resource. The hub resource is identified by its ARN.

```
aws securityhub describe-hub \
  --hub-arn "arn:aws:securityhub:us-west-1:123456789012:hub/default"
```

Output:

```
{
  "HubArn": "arn:aws:securityhub:us-west-1:123456789012:hub/default",
  "SubscribedAt": "2019-11-19T23:15:10.046Z"
}
```

For more information, see [AWS::SecurityHub::Hub](#) in the *AWS CloudFormation User Guide*.

- For API details, see [DescribeHub](#) in *AWS CLI Command Reference*.

describe-organization-configuration

The following code example shows how to use `describe-organization-configuration`.

AWS CLI

To view how Security Hub is configured for an organization

The following `describe-organization-configuration` example returns information about the way an organization is configured in Security Hub. In this example, the organization uses central configuration. Only the Security Hub administrator account can run this command.

```
aws securityhub describe-organization-configuration
```

Output:

```
{
  "AutoEnable": false,
  "MemberAccountLimitReached": false,
  "AutoEnableStandards": "NONE",
  "OrganizationConfiguration": {
    "ConfigurationType": "LOCAL",
    "Status": "ENABLED",
    "StatusMessage": "Central configuration has been enabled successfully"
  }
}
```

For more information, see [Managing accounts with AWS Organizations](#) in the *AWS Security Hub User Guide*.

- For API details, see [DescribeOrganizationConfiguration](#) in *AWS CLI Command Reference*.

describe-products

The following code example shows how to use `describe-products`.

AWS CLI

To return information about available product integrations

The following `describe-products` example returns the available product integrations one at a time.

```
aws securityhub describe-products \
  --max-results 1
```

Output:

```
{
  "NextToken": "U2FsdGVkX18vvP10qb7RD1rWRWVFBJI46M0IAb+nZmRJmR15NoRi2gm13sdQEn30/pq/78dGs+bKpgA+7HMPH00qX33/zoRI+uIG/F9yLNhc0r0WzFUdy36JcXLQji3Rpnn/cD1SVkGA98qI3zPOSDg==",
  "Products": [
    {
      "ProductArn": "arn:aws:securityhub:us-west-1:123456789333:product/crowdstrike/crowdstrike-falcon",
      "ProductName": "CrowdStrike Falcon",
      "CompanyName": "CrowdStrike",
      "Description": "CrowdStrike Falcon's single lightweight sensor unifies next-gen antivirus, endpoint detection and response, and 24/7 managed hunting, via the cloud.",
      "Categories": [
        "Endpoint Detection and Response (EDR)",
        "AV Scanning and Sandboxing",
        "Threat Intelligence Feeds and Reports",
        "Endpoint Forensics",
        "Network Forensics"
      ],
      "IntegrationTypes": [
        "SEND_FINDINGS_TO_SECURITY_HUB"
      ],
      "MarketplaceUrl": "https://aws.amazon.com/marketplace/seller-profile?id=a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "ActivationUrl": "https://falcon.crowdstrike.com/support/documentation",
      "ProductSubscriptionResourcePolicy": "{\"Version\": \"2012-10-17\", \"Statement\": [{\"Effect\": \"Allow\", \"Principal\": {\"AWS\": \"123456789333\"}, \"Action\": [\"securityhub:BatchImportFindings\"], \"Resource\": \"arn:aws:securityhub:us-west-1:123456789012:product-subscription/crowdstrike/crowdstrike-falcon\", \"Condition\": {\"StringEquals\": {\"securityhub:TargetAccount\": \"123456789012\"}}}], {\"Effect\": \"Allow\", \"Principal\": {\"AWS\": \"123456789012\"}, \"Action\": [\"securityhub:BatchImportFindings\"], \"Resource\": \"arn:aws:securityhub:us-west-1:123456789333:product/crowdstrike/crowdstrike-falcon\", \"Condition\": {\"StringEquals\": {\"securityhub:TargetAccount\": \"123456789012\"}}}]}"
    }
  ]
}
```

```
}
```

For more information, see [Managing product integrations](#) in the *AWS Security Hub User Guide*.

- For API details, see [DescribeProducts](#) in *AWS CLI Command Reference*.

describe-standards-controls

The following code example shows how to use `describe-standards-controls`.

AWS CLI

To request the list of controls in an enabled standard

The following `describe-standards-controls` example requests the list of controls in the requestor account's subscription to the PCI DSS standard. The request returns two controls at a time.

```
aws securityhub describe-standards-controls \
  --standards-subscription-arn "arn:aws:securityhub:us-
west-1:123456789012:subscription/pci-dss/v/3.2.1" \
  --max-results 2
```

Output:

```
{
  "Controls": [
    {
      "StandardsControlArn": "arn:aws:securityhub:us-
west-1:123456789012:control/pci-dss/v/3.2.1/PCI.AutoScaling.1",
      "ControlStatus": "ENABLED",
      "ControlStatusUpdatedAt": "2020-05-15T18:49:04.473000+00:00",
      "ControlId": "PCI.AutoScaling.1",
      "Title": "Auto scaling groups associated with a load balancer should use
health checks",
      "Description": "This AWS control checks whether your Auto Scaling groups
that are associated with a load balancer are using Elastic Load Balancing health
checks.",
      "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/
PCI.AutoScaling.1/remediation",
      "SeverityRating": "LOW",
      "RelatedRequirements": [
        "PCI DSS 2.2"
```

```

    ]
  },
  {
    "StandardsControlArn": "arn:aws:securityhub:us-
west-1:123456789012:control/pci-dss/v/3.2.1/PCI.CW.1",
    "ControlStatus": "ENABLED",
    "ControlStatusUpdatedAt": "2020-05-15T18:49:04.498000+00:00",
    "ControlId": "PCI.CW.1",
    "Title": "A log metric filter and alarm should exist for usage of the
\"root\" user",
    "Description": "This control checks for the CloudWatch metric
filters using the following pattern { $.userIdentity.type = \"Root\" &&
$.userIdentity.invokedBy NOT EXISTS && $.eventType != \"AwsServiceEvent\" }
It checks that the log group name is configured for use with active multi-
region CloudTrail, that there is at least one Event Selector for a Trail with
IncludeManagementEvents set to true and ReadWriteType set to All, and that there is
at least one active subscriber to an SNS topic associated with the alarm.",
    "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/
PCI.CW.1/remediation",
    "SeverityRating": "MEDIUM",
    "RelatedRequirements": [
      "PCI DSS 7.2.1"
    ]
  }
],
  "NextToken": "U2FsdGvKX1+eNkPoZHVl11ip5HUYQPWSWZGmftcmJiHL8JoKEsCDuaKayiPDyLK
+LiTkShveo0dvfxXCk0BaGhohIXhsIedN+LSjQV/
17kfCfJcq4PziNC1N9xe9aq2pjlLVZnznTfSImrodT5bRNHe4fELCQq/z+5ka
+5Lzmc11axcwTd5lKgQyQqmUVoeriHZhyIiBgWkF7oNYdBVG80EortVWvSkoUTt
+B2ThcnC7l43kI0UNxlkZ6sc64AsW"
}

```

For more information, see [Viewing details for controls](#) in the *AWS Security Hub User Guide*.

- For API details, see [DescribeStandardsControls](#) in *AWS CLI Command Reference*.

describe-standards

The following code example shows how to use describe-standards.

AWS CLI

To return a list of available standards

The following describe-standards example returns the list of available standards.

```
aws securityhub describe-standards
```

Output:

```
{
  "Standards": [
    {
      "StandardsArn": "arn:aws:securityhub:us-west-1::standards/aws-
foundational-security-best-practices/v/1.0.0",
      "Name": "AWS Foundational Security Best Practices v1.0.0",
      "Description": "The AWS Foundational Security Best Practices standard
is a set of automated security checks that detect when AWS accounts and deployed
resources do not align to security best practices. The standard is defined by AWS
security experts. This curated set of controls helps improve your security posture
in AWS, and cover AWS's most popular and foundational services.",
      "EnabledByDefault": true
    },
    {
      "StandardsArn": "arn:aws:securityhub:::ruleset/cis-aws-foundations-
benchmark/v/1.2.0",
      "Name": "CIS AWS Foundations Benchmark v1.2.0",
      "Description": "The Center for Internet Security (CIS) AWS Foundations
Benchmark v1.2.0 is a set of security configuration best practices for AWS. This
Security Hub standard automatically checks for your compliance readiness against a
subset of CIS requirements.",
      "EnabledByDefault": true
    },
    {
      "StandardsArn": "arn:aws:securityhub:us-west-1::standards/pci-dss/
v/3.2.1",
      "Name": "PCI DSS v3.2.1",
      "Description": "The Payment Card Industry Data Security Standard (PCI
DSS) v3.2.1 is an information security standard for entities that store, process,
and/or transmit cardholder data. This Security Hub standard automatically checks
for your compliance readiness against a subset of PCI DSS requirements.",
      "EnabledByDefault": false
    }
  ]
}
```

For more information, see [Security standards in AWS Security Hub](#) in the *AWS Security Hub User Guide*.

- For API details, see [DescribeStandards](#) in *AWS CLI Command Reference*.

disable-import-findings-for-product

The following code example shows how to use `disable-import-findings-for-product`.

AWS CLI

To stop receiving findings from a product integration

The following `disable-import-findings-for-product` example disables the flow of findings for the specified subscription to a product integration.

```
aws securityhub disable-import-findings-for-product \  
  --product-subscription-arn "arn:aws:securityhub:us-west-1:123456789012:product-  
subscription/crowdstrike/crowdstrike-falcon"
```

This command produces no output.

For more information, see [Managing product integrations](#) in the *AWS Security Hub User Guide*.

- For API details, see [DisableImportFindingsForProduct](#) in *AWS CLI Command Reference*.

disable-organization-admin-account

The following code example shows how to use `disable-organization-admin-account`.

AWS CLI

To remove a Security Hub administrator account

The following `disable-organization-admin-account` example revokes the specified account's assignment as a Security Hub administrator account for AWS Organizations.

```
aws securityhub disable-organization-admin-account \  
  --admin-account-id 777788889999
```

This command produces no output.

For more information, see [Designating a Security Hub administrator account](#) in the *AWS Security Hub User Guide*.

- For API details, see [DisableOrganizationAdminAccount](#) in *AWS CLI Command Reference*.

disable-security-hub

The following code example shows how to use `disable-security-hub`.

AWS CLI

To disable AWS Security Hub

The following `disable-security-hub` example disables AWS Security Hub for the requesting account.

```
aws securityhub disable-security-hub
```

This command produces no output.

For more information, see [Disabling AWS Security Hub](#) in the *AWS Security Hub User Guide*.

- For API details, see [DisableSecurityHub](#) in *AWS CLI Command Reference*.

disassociate-from-administrator-account

The following code example shows how to use `disassociate-from-administrator-account`.

AWS CLI

To disassociate from an administrator account

The following `disassociate-from-administrator-account` example disassociates the requesting account from its current administrator account.

```
aws securityhub disassociate-from-administrator-account
```

This command produces no output.

For more information, see [Managing administrator and member accounts](#) in the *AWS Security Hub User Guide*.

- For API details, see [DisassociateFromAdministratorAccount](#) in *AWS CLI Command Reference*.

disassociate-from-master-account

The following code example shows how to use `disassociate-from-master-account`.

AWS CLI

To disassociate from an administrator account

The following `disassociate-from-master-account` example disassociates the requesting account from its current administrator account.

```
aws securityhub disassociate-from-master-account
```

This command produces no output.

For more information, see [Managing administrator and member accounts](#) in the *AWS Security Hub User Guide*.

- For API details, see [DisassociateFromMasterAccount](#) in *AWS CLI Command Reference*.

disassociate-members

The following code example shows how to use `disassociate-members`.

AWS CLI

To disassociate member accounts

The following `disassociate-members` example disassociates the specified member accounts from the requesting administrator account.

```
aws securityhub disassociate-members \  
  --account-ids "123456789111" "123456789222"
```

This command produces no output.

For more information, see [Managing administrator and member accounts](#) in the *AWS Security Hub User Guide*.

- For API details, see [DisassociateMembers](#) in *AWS CLI Command Reference*.

enable-import-findings-for-product

The following code example shows how to use `enable-import-findings-for-product`.

AWS CLI

To start receiving findings from a product integration

The following `enable-import-findings-for-product` example enables the flow of findings from the specified product integration.

```
aws securityhub enable-import-findings-for-product \  
  --product-arn "arn:aws:securityhub:us-east-1:123456789333:product/crowdstrike/  
crowdstrike-falcon"
```

Output:

```
{  
  "ProductSubscriptionArn": "arn:aws:securityhub:us-east-1:123456789012:product-  
subscription/crowdstrike/crowdstrike-falcon"  
}
```

For more information, see [Managing product integrations](#) in the *AWS Security Hub User Guide*.

- For API details, see [EnableImportFindingsForProduct](#) in *AWS CLI Command Reference*.

enable-organization-admin-account

The following code example shows how to use `enable-organization-admin-account`.

AWS CLI

To designate an organization account as a Security Hub administrator account

The following `enable-organization-admin-account` example designates the specified account as a Security Hub administrator account.

```
aws securityhub enable-organization-admin-account \  
  --admin-account-id 777788889999
```

This command produces no output.

For more information, see [Designating a Security Hub administrator account](#) in the *AWS Security Hub User Guide*.

- For API details, see [EnableOrganizationAdminAccount](#) in *AWS CLI Command Reference*.

enable-security-hub

The following code example shows how to use `enable-security-hub`.

AWS CLI

To enable AWS Security Hub

The following `enable-security-hub` example enables AWS Security Hub for the requesting account. It configures Security Hub to enable the default standards. For the hub resource, it assigns the value `Security` to the tag `Department`.

```
aws securityhub enable-security-hub \  
  --enable-default-standards \  
  --tags '{"Department": "Security"}'
```

This command produces no output.

For more information, see [Enabling Security Hub](#) in the *AWS Security Hub User Guide*.

- For API details, see [EnableSecurityHub](#) in *AWS CLI Command Reference*.

get-administrator-account

The following code example shows how to use `get-administrator-account`.

AWS CLI

To retrieve information about an administrator account

The following `get-administrator-account` example retrieves information about the administrator account for the requesting account.

```
aws securityhub get-administrator-account
```

Output:

```
{
  "Master": {
    "AccountId": "123456789012",
    "InvitationId": "7ab938c5d52d7904ad09f9e7c20cc4eb",
    "InvitedAt": 2020-06-01T20:21:18.042000+00:00,
    "MemberStatus": "ASSOCIATED"
  }
}
```

For more information, see [Managing administrator and member accounts](#) in the *AWS Security Hub User Guide*.

- For API details, see [GetAdministratorAccount](#) in *AWS CLI Command Reference*.

get-configuration-policy-association

The following code example shows how to use `get-configuration-policy-association`.

AWS CLI

To get configuration association details for a target

The following `get-configuration-policy-association` example retrieves association details for the specified target. You can provide an account ID, organizational unit ID, or the root ID for the target.

```
aws securityhub get-configuration-policy-association \
  --target '{"OrganizationalUnitId": "ou-6hi7-8j91kl2m"}'
```

Output:

```
{
  "ConfigurationPolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
  "TargetId": "ou-6hi7-8j91kl2m",
  "TargetType": "ORGANIZATIONAL_UNIT",
  "AssociationType": "APPLIED",
  "UpdatedAt": "2023-09-26T21:13:01.816000+00:00",
  "AssociationStatus": "SUCCESS",
  "AssociationStatusMessage": "Association applied successfully on this target."
}
```

For more information, see [Viewing Security Hub configuration policies](#) in the *AWS Security Hub User Guide*.

- For API details, see [GetConfigurationPolicyAssociation](#) in *AWS CLI Command Reference*.

get-configuration-policy

The following code example shows how to use `get-configuration-policy`.

AWS CLI

To view configuration policy details

The following `get-configuration-policy` example retrieves details about the specified configuration policy.

```
aws securityhub get-configuration-policy \  
  --identifier "arn:aws:securityhub:eu-central-1:123456789012:configuration-policy/  
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

Output:

```
{  
  "Arn": "arn:aws:securityhub:eu-central-1:123456789012:configuration-policy/  
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
  "Id": "ce5ed1e7-9639-4e2f-9313-fa87fcef944b",  
  "Name": "SampleConfigurationPolicy",  
  "Description": "SampleDescription",  
  "UpdatedAt": "2023-11-28T20:28:04.494000+00:00",  
  "CreatedAt": "2023-11-28T20:28:04.494000+00:00",  
  "ConfigurationPolicy": {  
    "SecurityHub": {  
      "ServiceEnabled": true,  
      "EnabledStandardIdentifiers": [  
        "arn:aws:securityhub:eu-central-1::standards/aws-foundational-  
security-best-practices/v/1.0.0",  
        "arn:aws:securityhub:::ruleset/cis-aws-foundations-benchmark/  
v/1.2.0"  
      ],  
      "SecurityControlsConfiguration": {  
        "DisabledSecurityControlIdentifiers": [  
          "CloudTrail.2"  
        ],  
      }  
    }  
  }  
}
```

```

    "SecurityControlCustomParameters": [
      {
        "SecurityControlId": "ACM.1",
        "Parameters": {
          "daysToExpiration": {
            "ValueType": "CUSTOM",
            "Value": {
              "Integer": 15
            }
          }
        }
      }
    ]
  }
}

```

For more information, see [Viewing Security Hub configuration policies](#) in the *AWS Security Hub User Guide*.

- For API details, see [GetConfigurationPolicy](#) in *AWS CLI Command Reference*.

get-enabled-standards

The following code example shows how to use `get-enabled-standards`.

AWS CLI

To retrieve information about an enabled standard

The following `get-enabled-standards` example retrieves information about the PCI DSS standard.

```

aws securityhub get-enabled-standards \
  --standards-subscription-arn "arn:aws:securityhub:us-
west-1:123456789012:subscription/pci-dss/v/3.2.1"

```

Output:

```

{
  "StandardsSubscriptions": [
    {

```

```

        "StandardsArn": "arn:aws:securityhub:us-west-1::standards/pci-dss/
v/3.2.1",
        "StandardsInput": { },
        "StandardsStatus": "READY",
        "StandardsSubscriptionArn": "arn:aws:securityhub:us-
west-1:123456789012:subscription/pci-dss/v/3.2.1"
    }
]
}

```

For more information, see [Security standards in AWS Security Hub](#) in the *AWS Security Hub User Guide*.

- For API details, see [GetEnabledStandards](#) in *AWS CLI Command Reference*.

get-finding-aggregator

The following code example shows how to use `get-finding-aggregator`.

AWS CLI

To retrieve the current finding aggregation configuration

The following `get-finding-aggregator` example retrieves the current finding aggregation configuration.

```

aws securityhub get-finding-aggregator \
  --finding-aggregator-arn arn:aws:securityhub:us-east-1:222222222222:finding-
aggregator/123e4567-e89b-12d3-a456-426652340000

```

Output:

```

{
  "FindingAggregatorArn": "arn:aws:securityhub:us-east-1:222222222222:finding-
aggregator/123e4567-e89b-12d3-a456-426652340000",
  "FindingAggregationRegion": "us-east-1",
  "RegionLinkingMode": "SPECIFIED_REGIONS",
  "Regions": "us-west-1,us-west-2"
}

```

For more information, see [Viewing the current finding aggregation configuration](#) in the *AWS Security Hub User Guide*.

- For API details, see [GetFindingAggregator](#) in *AWS CLI Command Reference*.

get-finding-history

The following code example shows how to use `get-finding-history`.

AWS CLI

To get finding history

The following `get-finding-history` example gets up to the last 90 days of history for the specified finding. In this example, the results are limited to two records of finding history.

```
aws securityhub get-finding-history \
  --finding-identifier Id="arn:aws:securityhub:us-
east-1:123456789012:security-control/S3.17/finding/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",ProductArn="arn:aws:securityhub:us-east-1::product/aws/securityhub"
```

Output:

```
{
  "Records": [
    {
      "FindingIdentifier": {
        "Id": "arn:aws:securityhub:us-east-1:123456789012:security-control/
S3.17/finding/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "ProductArn": "arn:aws:securityhub:us-east-1::product/aws/
securityhub"
      },
      "UpdateTime": "2023-06-02T03:15:25.685000+00:00",
      "FindingCreated": false,
      "UpdateSource": {
        "Type": "BATCH_IMPORT_FINDINGS",
        "Identity": "arn:aws:securityhub:us-east-1::product/aws/securityhub"
      },
      "Updates": [
        {
          "UpdatedField": "Compliance.RelatedRequirements",
          "OldValue": "[\"NIST.800-53.r5 SC-12(2)\",\"NIST.800-53.r5
SC-12(3)\",\"NIST.800-53.r5 SC-12(6)\",\"NIST.800-53.r5 CM-3(6)\",\"NIST.800-53.r5
SC-13\", \"NIST.800-53.r5 SC-28\", \"NIST.800-53.r5 SC-28(1)\", \"NIST.800-53.r5
SC-7(10)\"]",
```



```

        "NewValue": "[\\"NIST.800-53.r5 SC-12(2)\",\\"NIST.800-53.r5
CM-3(6)\",\\"NIST.800-53.r5 SC-13\",\\"NIST.800-53.r5 SC-28\",\\"NIST.800-53.r5
SC-28(1)\",\\"NIST.800-53.r5 SC-7(10)\",\\"NIST.800-53.r5 CA-9(1)\",\\"NIST.800-53.r5
SI-7(6)\",\\"NIST.800-53.r5 AU-9\"]"
    },
    {
        "UpdatedField": "LastObservedAt",
        "OldValue": "2023-06-01T09:15:38.587Z",
        "NewValue": "2023-06-02T03:15:22.946Z"
    },
    {
        "UpdatedField": "UpdatedAt",
        "OldValue": "2023-06-01T09:15:31.049Z",
        "NewValue": "2023-06-02T03:15:14.861Z"
    },
    {
        "UpdatedField": "ProcessedAt",
        "OldValue": "2023-06-01T09:15:41.058Z",
        "NewValue": "2023-06-02T03:15:25.685Z"
    }
]
},
{
    "FindingIdentifier": {
        "Id": "arn:aws:securityhub:us-east-1:123456789012:security-control/
S3.17/finding/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "ProductArn": "arn:aws:securityhub:us-east-1::product/aws/
securityhub"
    },
    "UpdateTime": "2023-05-23T02:06:51.518000+00:00",
    "FindingCreated": "true",
    "UpdateSource": {
        "Type": "BATCH_IMPORT_FINDINGS",
        "Identity": "arn:aws:securityhub:us-east-1::product/aws/securityhub"
    },
    "Updates": []
}
]
}

```

For more information, see [Finding history](#) in the *AWS Security Hub User Guide*.

- For API details, see [GetFindingHistory](#) in *AWS CLI Command Reference*.

get-findings

The following code example shows how to use get-findings.

AWS CLI

Example 1: To return findings generated for a specific standard

The following get-findings example returns findings for the PCI DSS standard.

```
aws securityhub get-findings \  
  --filters '{"GeneratorId":[{"Value": "pci-dss","Comparison":"PREFIX"}]}' \  
  --max-items 1
```

Output:

```
{  
  "Findings": [  
    {  
      "SchemaVersion": "2018-10-08",  
      "Id": "arn:aws:securityhub:eu-central-1:123456789012:subscription/pci-  
dss/v/3.2.1/PCI.Lambda.2/finding/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
      "ProductArn": "arn:aws:securityhub:us-west-1::product/aws/securityhub",  
      "GeneratorId": "pci-dss/v/3.2.1/PCI.Lambda.2",  
      "AwsAccountId": "123456789012",  
      "Types": [  
        "Software and Configuration Checks/Industry and Regulatory  
Standards/PCI-DSS"  
      ],  
      "FindingProviderFields": {  
        "Severity": {  
          "Original": 0,  
          "Label": "INFORMATIONAL"  
        },  
        "Types": [  
          "Software and Configuration Checks/Industry and Regulatory  
Standards/PCI-DSS"  
        ]  
      },  
      "FirstObservedAt": "2020-06-02T14:02:49.159Z",  
      "LastObservedAt": "2020-06-02T14:02:52.397Z",  
      "CreatedAt": "2020-06-02T14:02:49.159Z",  
      "UpdatedAt": "2020-06-02T14:02:52.397Z",  
      "Severity": {
```

```
        "Original": 0,
        "Label": "INFORMATIONAL",
        "Normalized": 0
    },
    "Title": "PCI.Lambda.2 Lambda functions should be in a VPC",
    "Description": "This AWS control checks whether a Lambda function is in
a VPC.",
    "Remediation": {
        "Recommendation": {
            "Text": "For directions on how to fix this issue, please consult
the AWS Security Hub PCI DSS documentation.",
            "Url": "https://docs.aws.amazon.com/console/securityhub/
PCI.Lambda.2/remediation"
        }
    },
    "ProductFields": {
        "StandardsArn": "arn:aws:securityhub::standards/pci-dss/v/3.2.1",
        "StandardsSubscriptionArn": "arn:aws:securityhub:us-
west-1:123456789012:subscription/pci-dss/v/3.2.1",
        "ControlId": "PCI.Lambda.2",
        "RecommendationUrl": "https://docs.aws.amazon.com/console/
securityhub/PCI.Lambda.2/remediation",
        "RelatedAWSResources:0/name": "securityhub-lambda-inside-
vpc-0e904a3b",
        "RelatedAWSResources:0/type": "AWS::Config::ConfigRule",
        "StandardsControlArn": "arn:aws:securityhub:us-
west-1:123456789012:control/pci-dss/v/3.2.1/PCI.Lambda.2",
        "aws/securityhub/SeverityLabel": "INFORMATIONAL",
        "aws/securityhub/ProductName": "Security Hub",
        "aws/securityhub/CompanyName": "AWS",
        "aws/securityhub/FindingId": "arn:aws:securityhub:eu-
central-1::product/aws/securityhub/arn:aws:securityhub:eu-
central-1:123456789012:subscription/pci-dss/v/3.2.1/PCI.Lambda.2/finding/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    },
    "Resources": [
        {
            "Type": "AwsAccount",
            "Id": "AWS:::Account:123456789012",
            "Partition": "aws",
            "Region": "us-west-1"
        }
    ],
    "Compliance": {
```

```

        "Status": "PASSED",
        "RelatedRequirements": [
            "PCI DSS 1.2.1",
            "PCI DSS 1.3.1",
            "PCI DSS 1.3.2",
            "PCI DSS 1.3.4"
        ]
    },
    "WorkflowState": "NEW",
    "Workflow": {
        "Status": "NEW"
    },
    "RecordState": "ARCHIVED"
}
],
"NextToken": "eyJ0ZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAxfQ=="
}

```

Example 2: To return critical-severity findings that have a workflow status of NOTIFIED

The following `get-findings` example returns findings that have a severity label value of `CRITICAL` and a workflow status of `NOTIFIED`. The results are sorted in descending order by the value of `Confidence`.

```

aws securityhub get-findings \
  --filters '{"SeverityLabel":[{"Value":
"CRITICAL","Comparison":"EQUALS"}],"WorkflowStatus":
[{"Value":"NOTIFIED","Comparison":"EQUALS"}]}' \
  --sort-criteria '{ "Field": "Confidence", "SortOrder": "desc"}' \
  --max-items 1

```

Output:

```

{
  "Findings": [
    {
      "SchemaVersion": "2018-10-08",
      "Id": "arn:aws:securityhub:us-west-1: 123456789012:subscription/cis-aws-
foundations-benchmark/v/1.2.0/1.13/finding/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "ProductArn": "arn:aws:securityhub:us-west-2::product/aws/securityhub",
      "GeneratorId": "arn:aws:securityhub:::ruleset/cis-aws-foundations-
benchmark/v/1.2.0/rule/1.13",
      "AwsAccountId": "123456789012",

```

```
    "Types": [
      "Software and Configuration Checks/Industry and Regulatory
Standards/CIS AWS Foundations Benchmark"
    ],
    "FindingProviderFields" {
      "Severity": {
        "Original": 90,
        "Label": "CRITICAL"
      },
      "Types": [
        "Software and Configuration Checks/Industry and Regulatory
Standards/CIS AWS Foundations Benchmark"
      ]
    },
    "FirstObservedAt": "2020-05-21T20:16:34.752Z",
    "LastObservedAt": "2020-06-09T08:16:37.171Z",
    "CreatedAt": "2020-05-21T20:16:34.752Z",
    "UpdatedAt": "2020-06-09T08:16:36.430Z",
    "Severity": {
      "Original": 90,
      "Label": "CRITICAL",
      "Normalized": 90
    },
    "Title": "1.13 Ensure MFA is enabled for the \"root\" account",
    "Description": "The root account is the most privileged user in an AWS
account. MFA adds an extra layer of protection on top of a user name and password.
With MFA enabled, when a user signs in to an AWS website, they will be prompted for
their user name and password as well as for an authentication code from their AWS
MFA device.",
    "Remediation": {
      "Recommendation": {
        "Text": "For directions on how to fix this issue, please consult
the AWS Security Hub CIS documentation.",
        "Url": "https://docs.aws.amazon.com/console/securityhub/
standards-cis-1.13/remediation"
      }
    },
    "ProductFields": {
      "StandardsGuideArn": "arn:aws:securityhub:::ruleset/cis-aws-
foundations-benchmark/v/1.2.0",
      "StandardsGuideSubscriptionArn": "arn:aws:securityhub:us-
west-1:123456789012:subscription/cis-aws-foundations-benchmark/v/1.2.0",
      "RuleId": "1.13",
```

```

        "RecommendationUrl": "https://docs.aws.amazon.com/console/
securityhub/standards-cis-1.13/remediation",
        "RelatedAWSResources:0/name": "securityhub-root-account-mfa-
enabled-5pftha",
        "RelatedAWSResources:0/type": "AWS::Config::ConfigRule",
        "StandardsControlArn": "arn:aws:securityhub:us-
west-1:123456789012:control/cis-aws-foundations-benchmark/v/1.2.0/1.13",
        "aws/securityhub/SeverityLabel": "CRITICAL",
        "aws/securityhub/ProductName": "Security Hub",
        "aws/securityhub/CompanyName": "AWS",
        "aws/securityhub/FindingId": "arn:aws:securityhub:us-
west-1::product/aws/securityhub/arn:aws:securityhub:us-
west-1:123456789012:subscription/cis-aws-foundations-benchmark/v/1.2.0/1.13/finding/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    },
    "Resources": [
        {
            "Type": "AwsAccount",
            "Id": "AWS:::Account:123456789012",
            "Partition": "aws",
            "Region": "us-west-1"
        }
    ],
    "Compliance": {
        "Status": "FAILED"
    },
    "WorkflowState": "NEW",
    "Workflow": {
        "Status": "NOTIFIED"
    },
    "RecordState": "ACTIVE"
}
]
}

```

For more information, see [Filtering and grouping findings](#) in the *AWS Security Hub User Guide*.

- For API details, see [GetFindings](#) in *AWS CLI Command Reference*.

get-insight-results

The following code example shows how to use `get-insight-results`.

AWS CLI

To retrieve the results for an insight

The following `get-insight-results` example returns the list of insight results for the insight with the specified ARN.

```
aws securityhub get-insight-results \  
  --insight-arn "arn:aws:securityhub:us-west-1:123456789012:insight/123456789012/  
custom/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

Output:

```
{  
  "InsightResults": {  
    "GroupByAttribute": "ResourceId",  
    "InsightArn": "arn:aws:securityhub:us-  
west-1:123456789012:insight/123456789012/custom/a1b2c3d4-5678-90ab-cdef-  
EXAMPLE11111",  
    "ResultValues": [  
      {  
        "Count": 10,  
        "GroupByAttributeValue": "AWS:::Account:123456789111"  
      },  
      {  
        "Count": 3,  
        "GroupByAttributeValue": "AWS:::Account:123456789222"  
      }  
    ]  
  }  
}
```

For more information, see [Viewing and taking action on insight results and findings](#) in the *AWS Security Hub User Guide*.

- For API details, see [GetInsightResults](#) in *AWS CLI Command Reference*.

get-insights

The following code example shows how to use `get-insights`.

AWS CLI

To retrieve details about an insight

The following `get-insights` example retrieves the configuration details for the insight with the specified ARN.

```
aws securityhub get-insights \  
  --insight-arns "arn:aws:securityhub:us-west-1:123456789012:insight/123456789012/  
custom/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

Output:

```
{  
  "Insights": [  
    {  
      "Filters": {  
        "ResourceType": [  
          {  
            "Comparison": "EQUALS",  
            "Value": "AwsIamRole"  
          }  
        ],  
        "SeverityLabel": [  
          {  
            "Comparison": "EQUALS",  
            "Value": "CRITICAL"  
          }  
        ],  
      },  
      "GroupByAttribute": "ResourceId",  
      "InsightArn": "arn:aws:securityhub:us-  
west-1:123456789012:insight/123456789012/custom/a1b2c3d4-5678-90ab-cdef-  
EXAMPLE11111",  
      "Name": "Critical role findings"  
    }  
  ]  
}
```

For more information, see [Insights in AWS Security Hub](#) in the *AWS Security Hub User Guide*.

- For API details, see [GetInsights](#) in *AWS CLI Command Reference*.

get-invitations-count

The following code example shows how to use `get-invitations-count`.

AWS CLI

To retrieve the number of invitations that were not accepted

The following `get-invitations-count` example retrieves the number of invitations that the requesting account declined or did not respond to.

```
aws securityhub get-invitations-count
```

Output:

```
{
  "InvitationsCount": 3
}
```

For more information, see [Managing administrator and member accounts](#) in the *AWS Security Hub User Guide*.

- For API details, see [GetInvitationsCount](#) in *AWS CLI Command Reference*.

get-master-account

The following code example shows how to use `get-master-account`.

AWS CLI

To retrieve information about an administrator account

The following `get-master-account` example retrieves information about the administrator account for the requesting account.

```
aws securityhub get-master-account
```

Output:

```
{
```

```
"Master": {
  "AccountId": "123456789012",
  "InvitationId": "7ab938c5d52d7904ad09f9e7c20cc4eb",
  "InvitedAt": 2020-06-01T20:21:18.042000+00:00,
  "MemberStatus": "ASSOCIATED"
}
```

For more information, see [Managing administrator and member accounts](#) in the *AWS Security Hub User Guide*.

- For API details, see [GetMasterAccount](#) in *AWS CLI Command Reference*.

get-members

The following code example shows how to use `get-members`.

AWS CLI

To retrieve information about selected member accounts

The following `get-members` example retrieves information about the specified member accounts.

```
aws securityhub get-members \
  --account-ids "444455556666" "777788889999"
```

Output:

```
{
  "Members": [
    {
      "AccountId": "123456789111",
      "AdministratorId": "123456789012",
      "InvitedAt": 2020-06-01T20:15:15.289000+00:00,
      "MasterId": "123456789012",
      "MemberStatus": "ASSOCIATED",
      "UpdatedAt": 2020-06-01T20:15:15.289000+00:00
    },
    {
      "AccountId": "123456789222",
```

```

    "AdministratorId": "123456789012",
    "InvitedAt": 2020-06-01T20:15:15.289000+00:00,
    "MasterId": "123456789012",
    "MemberStatus": "ASSOCIATED",
    "UpdatedAt": 2020-06-01T20:15:15.289000+00:00
  }
],
"UnprocessedAccounts": [ ]
}

```

For more information, see [Managing administrator and member accounts](#) in the *AWS Security Hub User Guide*.

- For API details, see [GetMembers](#) in *AWS CLI Command Reference*.

get-security-control-definition

The following code example shows how to use `get-security-control-definition`.

AWS CLI

To get security control definition details

The following `get-security-control-definition` example retrieves definition details for a Security Hub security control. Details include the control title, description, Region availability, parameters, and other information.

```
aws securityhub get-security-control-definition \
  --security-control-id ACM.1
```

Output:

```

{
  "SecurityControlDefinition": {
    "SecurityControlId": "ACM.1",
    "Title": "Imported and ACM-issued certificates should be renewed after a
specified time period",
    "Description": "This control checks whether an AWS Certificate Manager
(ACM) certificate is renewed within the specified time period. It checks both
imported certificates and certificates provided by ACM. The control fails if the
certificate isn't renewed within the specified time period. Unless you provide a

```

```
custom parameter value for the renewal period, Security Hub uses a default value of
30 days.",
  "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/ACM.1/
remediation",
  "SeverityRating": "MEDIUM",
  "CurrentRegionAvailability": "AVAILABLE",
  "ParameterDefinitions": {
    "daysToExpiration": {
      "Description": "Number of days within which the ACM certificate must
be renewed",
      "ConfigurationOptions": {
        "Integer": {
          "DefaultValue": 30,
          "Min": 14,
          "Max": 365
        }
      }
    }
  }
}
```

For more information, see [Custom control parameters](#) in the *AWS Security Hub User Guide*.

- For API details, see [GetSecurityControlDefinition](#) in *AWS CLI Command Reference*.

invite-members

The following code example shows how to use `invite-members`.

AWS CLI

To send invitations to member accounts

The following `invite-members` example sends invitations to the specified member accounts.

```
aws securityhub invite-members \
  --account-ids "123456789111" "123456789222"
```

Output:

```
{
  "UnprocessedAccounts": []
}
```

```
}
```

For more information, see [Managing administrator and member accounts](#) in the *AWS Security Hub User Guide*.

- For API details, see [InviteMembers](#) in *AWS CLI Command Reference*.

list-automation-rules

The following code example shows how to use `list-automation-rules`.

AWS CLI

To view a list of automation rules

The following `list-automation-rules` example lists the automation rules for an AWS account. Only the Security Hub administrator account can run this command.

```
aws securityhub list-automation-rules \  
  --max-results 3 \  
  --next-token NULL
```

Output:

```
{  
  "AutomationRulesMetadata": [  
    {  
      "RuleArn": "arn:aws:securityhub:us-east-1:123456789012:automation-rule/  
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
      "RuleStatus": "ENABLED",  
      "RuleOrder": 1,  
      "RuleName": "Suppress informational findings",  
      "Description": "Suppress GuardDuty findings with Informational  
severity",  
      "IsTerminal": false,  
      "CreatedAt": "2023-05-31T17:56:14.837000+00:00",  
      "UpdatedAt": "2023-05-31T17:59:38.466000+00:00",  
      "CreatedBy": "arn:aws:iam::123456789012:role/Admin"  
    },  
    {  
      "RuleArn": "arn:aws:securityhub:us-east-1:123456789012:automation-rule/  
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
```

```

    "RuleStatus": "ENABLED",
    "RuleOrder": 1,
    "RuleName": "sample rule",
    "Description": "A sample rule",
    "IsTerminal": false,
    "CreatedAt": "2023-07-15T23:37:20.223000+00:00",
    "UpdatedAt": "2023-07-15T23:37:20.223000+00:00",
    "CreatedBy": "arn:aws:iam::123456789012:role/Admin"
  },
  {
    "RuleArn": "arn:aws:securityhub:us-east-1:123456789012:automation-rule/
a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "RuleStatus": "ENABLED",
    "RuleOrder": 1,
    "RuleName": "sample rule",
    "Description": "A sample rule",
    "IsTerminal": false,
    "CreatedAt": "2023-07-15T23:45:25.126000+00:00",
    "UpdatedAt": "2023-07-15T23:45:25.126000+00:00",
    "CreatedBy": "arn:aws:iam::123456789012:role/Admin"
  }
]
}

```

For more information, see [Viewing automation rules](#) in the *AWS Security Hub User Guide*.

- For API details, see [ListAutomationRules](#) in *AWS CLI Command Reference*.

list-configuration-policies

The following code example shows how to use `list-configuration-policies`.

AWS CLI

To list configuration policy summaries

The following `list-configuration-policies` example lists a summary of configuration policies for the organization.

```
aws securityhub list-configuration-policies \
  --max-items 3
```

Output:

```
{
  "ConfigurationPolicySummaries": [
    {
      "Arn": "arn:aws:securityhub:eu-central-1:123456789012:configuration-policy/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "Name": "SampleConfigurationPolicy1",
      "Description": "SampleDescription1",
      "UpdatedAt": "2023-09-26T21:08:36.214000+00:00",
      "ServiceEnabled": true
    },
    {
      "Arn": "arn:aws:securityhub:eu-central-1:123456789012:configuration-policy/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
      "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
      "Name": "SampleConfigurationPolicy2",
      "Description": "SampleDescription2",
      "UpdatedAt": "2023-11-28T19:26:25.207000+00:00",
      "ServiceEnabled": true
    },
    {
      "Arn": "arn:aws:securityhub:eu-central-1:123456789012:configuration-policy/a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
      "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
      "Name": "SampleConfigurationPolicy3",
      "Description": "SampleDescription3",
      "UpdatedAt": "2023-11-28T20:28:04.494000+00:00",
      "ServiceEnabled": true
    }
  ]
}
```

For more information, see [Viewing Security Hub configuration policies](#) in the *AWS Security Hub User Guide*.

- For API details, see [ListConfigurationPolicies](#) in *AWS CLI Command Reference*.

list-configuration-policy-associations

The following code example shows how to use `list-configuration-policy-associations`.

AWS CLI

To list configuration associations

The following `list-configuration-policy-associations` example lists a summary of configuration associations for the organization. The response include associations with configuration policies and self-managed behavior.

```
aws securityhub list-configuration-policy-associations \  
  --association-type "APPLIED" \  
  --max-items 4
```

Output:

```
{  
  "ConfigurationPolicyAssociationSummaries": [  
    {  
      "ConfigurationPolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
      "TargetId": "r-1ab2",  
      "TargetType": "ROOT",  
      "AssociationType": "APPLIED",  
      "UpdatedAt": "2023-11-28T19:26:49.417000+00:00",  
      "AssociationStatus": "FAILED",  
      "AssociationStatusMessage": "Policy association failed because 2  
organizational units or accounts under this root failed."  
    },  
    {  
      "ConfigurationPolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",  
      "TargetId": "ou-1ab2-c3de4f5g",  
      "TargetType": "ORGANIZATIONAL_UNIT",  
      "AssociationType": "APPLIED",  
      "UpdatedAt": "2023-09-26T21:14:05.283000+00:00",  
      "AssociationStatus": "FAILED",  
      "AssociationStatusMessage": "One or more children under this target  
failed association."  
    },  
    {  
      "ConfigurationPolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",  
      "TargetId": "ou-6hi7-8j91kl2m",  
      "TargetType": "ORGANIZATIONAL_UNIT",  
      "AssociationType": "APPLIED",  
      "UpdatedAt": "2023-09-26T21:13:01.816000+00:00",  
      "AssociationStatus": "SUCCESS",  
      "AssociationStatusMessage": "Association applied successfully on this  
target."  
    },  
    {
```



```
"ConfigurationPolicyId": "SELF_MANAGED_SECURITY_HUB",
"TargetId": "111122223333",
"TargetType": "ACCOUNT",
"AssociationType": "APPLIED",
"UpdatedAt": "2023-11-28T22:01:26.409000+00:00",
"AssociationStatus": "SUCCESS"
}
}
```

For more information, see [Viewing Security Hub configuration policies](#) in the *AWS Security Hub User Guide*.

- For API details, see [ListConfigurationPolicyAssociations](#) in *AWS CLI Command Reference*.

list-enabled-products-for-import

The following code example shows how to use `list-enabled-products-for-import`.

AWS CLI

To return the list of enabled product integrations

The following `list-enabled-products-for-import` example returns the list of subscription ARNS for the currently enabled product integrations.

```
aws securityhub list-enabled-products-for-import
```

Output:

```
{
  "ProductSubscriptions": [ "arn:aws:securityhub:us-west-1:123456789012:product-
subscription/crowdstrike/crowdstrike-falcon", "arn:aws:securityhub:us-
west-1:123456789012:product-subscription/aws/securityhub" ]
}
```

For more information, see [Managing product integrations](#) in the *AWS Security Hub User Guide*.

- For API details, see [ListEnabledProductsForImport](#) in *AWS CLI Command Reference*.

list-finding-aggregators

The following code example shows how to use `list-finding-aggregators`.

AWS CLI

To list the available widgets

The following `list-finding-aggregators` example returns the ARN of the finding aggregation configuration.

```
aws securityhub list-finding-aggregators
```

Output:

```
{
  "FindingAggregatorArn": "arn:aws:securityhub:us-east-1:222222222222:finding-
aggregator/123e4567-e89b-12d3-a456-426652340000"
}
```

For more information, see [Viewing the current finding aggregation configuration](#) in the *AWS Security Hub User Guide*.

- For API details, see [ListFindingAggregators](#) in *AWS CLI Command Reference*.

list-invitations

The following code example shows how to use `list-invitations`.

AWS CLI

To display a list of invitations

The following `list-invitations` example retrieves the list of invitations sent to the requesting account.

```
aws securityhub list-invitations
```

Output:

```
{
  "Invitations": [
    {
      "AccountId": "123456789012",
```

```
        "InvitationId": "7ab938c5d52d7904ad09f9e7c20cc4eb",
        "InvitedAt": 2020-06-01T20:21:18.042000+00:00,
        "MemberStatus": "ASSOCIATED"
    },
],
}
```

For more information, see [Managing administrator and member accounts](#) in the *AWS Security Hub User Guide*.

- For API details, see [ListInvitations](#) in *AWS CLI Command Reference*.

list-members

The following code example shows how to use `list-members`.

AWS CLI

To retrieve a list of member accounts

The following `list-members` example returns the list of member accounts for the requesting administrator account.

```
aws securityhub list-members
```

Output:

```
{
  "Members": [
    {
      "AccountId": "123456789111",
      "AdministratorId": "123456789012",
      "InvitedAt": 2020-06-01T20:15:15.289000+00:00,
      "MasterId": "123456789012",
      "MemberStatus": "ASSOCIATED",
      "UpdatedAt": 2020-06-01T20:15:15.289000+00:00
    },
    {
      "AccountId": "123456789222",
      "AdministratorId": "123456789012",
      "InvitedAt": 2020-06-01T20:15:15.289000+00:00,
      "MasterId": "123456789012",

```

```
        "MemberStatus": "ASSOCIATED",
        "UpdatedAt": 2020-06-01T20:15:15.289000+00:00
      },
    ],
  }
}
```

For more information, see [Managing administrator and member accounts](#) in the *AWS Security Hub User Guide*.

- For API details, see [ListMembers](#) in *AWS CLI Command Reference*.

list-organization-admin-accounts

The following code example shows how to use `list-organization-admin-accounts`.

AWS CLI

To list the designated Security Hub administrator accounts

The following `list-organization-admin-accounts` example lists the Security Hub administrator accounts for an organization.

```
aws securityhub list-organization-admin-accounts
```

Output:

```
{
  AdminAccounts": [
    { "AccountId": "777788889999" },
    { "Status": "ENABLED" }
  ]
}
```

For more information, see [Designating a Security Hub administrator account](#) in the *AWS Security Hub User Guide*.

- For API details, see [ListOrganizationAdminAccounts](#) in *AWS CLI Command Reference*.

list-security-control-definitions

The following code example shows how to use `list-security-control-definitions`.

AWS CLI

Example 1: To list all available security controls

The following `list-security-control-definitions` example lists the available security controls across all Security Hub standards. This example limits the results to three controls.

```
aws securityhub list-security-control-definitions \  
  --max-items 3
```

Output:

```
{  
  "SecurityControlDefinitions": [  
    {  
      "SecurityControlId": "ACM.1",  
      "Title": "Imported and ACM-issued certificates should be renewed after a  
specified time period",  
      "Description": "This control checks whether an AWS Certificate Manager  
(ACM) certificate is renewed within the specified time period. It checks both  
imported certificates and certificates provided by ACM. The control fails if the  
certificate isn't renewed within the specified time period. Unless you provide a  
custom parameter value for the renewal period, Security Hub uses a default value of  
30 days.",  
      "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/  
ACM.1/remediation",  
      "SeverityRating": "MEDIUM",  
      "CurrentRegionAvailability": "AVAILABLE",  
      "CustomizableProperties": [  
        "Parameters"  
      ]  
    },  
    {  
      "SecurityControlId": "ACM.2",  
      "Title": "RSA certificates managed by ACM should use a key length of at  
least 2,048 bits",  
      "Description": "This control checks whether RSA certificates managed by  
AWS Certificate Manager use a key length of at least 2,048 bits. The control fails  
if the key length is smaller than 2,048 bits.",  
      "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/  
ACM.2/remediation",  
      "SeverityRating": "HIGH",  
      "CurrentRegionAvailability": "AVAILABLE",
```

```

        "CustomizableProperties": [],
    },
    {
        "SecurityControlId": "APIGateway.1",
        "Title": "API Gateway REST and WebSocket API execution logging should be
enabled",
        "Description": "This control checks whether all stages of an Amazon
API Gateway REST or WebSocket API have logging enabled. The control fails if
the 'loggingLevel' isn't 'ERROR' or 'INFO' for all stages of the API. Unless you
provide custom parameter values to indicate that a specific log type should be
enabled, Security Hub produces a passed finding if the logging level is either
'ERROR' or 'INFO'.",
        "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/
APIGateway.1/remediation",
        "SeverityRating": "MEDIUM",
        "CurrentRegionAvailability": "AVAILABLE",
        "CustomizableProperties": [
            "Parameters"
        ]
    }
],
"NextToken": "U2FsdGVkX1/UprCPzxVbkDeHikDXbDxfgJZ1w2RG1XWsFPTMTIQPVE0m/
FduIGxS70bRtAbaUt/8/RCQcg2PU0YXI20hH/Grho0Tgv+Tsm0qvQVFhkJepWmqh
+NYawjocVBeos6xzn/8qnbF9IuwGg=="
}

```

For more information, see [Viewing details for a standard](#) in the *AWS Security Hub User Guide*.

Example 2: To list available security controls for a specific standard

The following `list-security-control-definitions` example lists the available security controls for the CIS AWS Foundations Benchmark v1.4.0. This example limits the results to three controls.

```

aws securityhub list-security-control-definitions \
  --standards-arn "arn:aws:securityhub:us-east-1::standards/cis-aws-foundations-
benchmark/v/1.4.0" \
  --max-items 3

```

Output:

```
{
```

```

"SecurityControlDefinitions": [
  {
    "SecurityControlId": "CloudTrail.1",
    "Title": "CloudTrail should be enabled and configured with at least one
multi-Region trail that includes read and write management events",
    "Description": "This AWS control checks that there is at least one
multi-region AWS CloudTrail trail includes read and write management events.",
    "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/
CloudTrail.1/remediation",
    "SeverityRating": "HIGH",
    "CurrentRegionAvailability": "AVAILABLE",
    "CustomizableProperties": []
  },
  {
    "SecurityControlId": "CloudTrail.2",
    "Title": "CloudTrail should have encryption at-rest enabled",
    "Description": "This AWS control checks whether AWS CloudTrail is
configured to use the server side encryption (SSE) AWS Key Management Service (AWS
KMS) customer master key (CMK) encryption. The check will pass if the KmsKeyId is
defined.",
    "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/
CloudTrail.2/remediation",
    "SeverityRating": "MEDIUM",
    "CurrentRegionAvailability": "AVAILABLE",
    "CustomizableProperties": []
  },
  {
    "SecurityControlId": "CloudTrail.4",
    "Title": "CloudTrail log file validation should be enabled",
    "Description": "This AWS control checks whether CloudTrail log file
validation is enabled.",
    "RemediationUrl": "https://docs.aws.amazon.com/console/securityhub/
CloudTrail.4/remediation",
    "SeverityRating": "MEDIUM",
    "CurrentRegionAvailability": "AVAILABLE",
    "CustomizableProperties": []
  }
],
"NextToken": "eyJ0ZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAzfQ=="
}

```

For more information, see [Viewing details for a standard](#) in the *AWS Security Hub User Guide*.

- For API details, see [ListSecurityControlDefinitions](#) in *AWS CLI Command Reference*.

list-standards-control-associations

The following code example shows how to use `list-standards-control-associations`.

AWS CLI

To get the enablement status of a control in each enabled standard

The following `list-standards-control-associations` example lists the enablement status of `CloudTrail.1` in each enabled standard.

```
aws securityhub list-standards-control-associations \  
  --security-control-id CloudTrail.1
```

Output:

```
{  
  "StandardsControlAssociationSummaries": [  
    {  
      "StandardsArn": "arn:aws:securityhub:us-east-2::standards/nist-800-53/  
v/5.0.0",  
      "SecurityControlId": "CloudTrail.1",  
      "SecurityControlArn": "arn:aws:securityhub:us-  
east-2:123456789012:security-control/CloudTrail.1",  
      "AssociationStatus": "ENABLED",  
      "RelatedRequirements": [  
        "NIST.800-53.r5 AC-2(4)",  
        "NIST.800-53.r5 AC-4(26)",  
        "NIST.800-53.r5 AC-6(9)",  
        "NIST.800-53.r5 AU-10",  
        "NIST.800-53.r5 AU-12",  
        "NIST.800-53.r5 AU-2",  
        "NIST.800-53.r5 AU-3",  
        "NIST.800-53.r5 AU-6(3)",  
        "NIST.800-53.r5 AU-6(4)",  
        "NIST.800-53.r5 AU-14(1)",  
        "NIST.800-53.r5 CA-7",  
        "NIST.800-53.r5 SC-7(9)",  
        "NIST.800-53.r5 SI-3(8)",  
        "NIST.800-53.r5 SI-4(20)",  
        "NIST.800-53.r5 SI-7(8)",  
        "NIST.800-53.r5 SA-8(22)"  
      ],  
    },  
  ],  
}
```



```
    "UpdatedAt": "2023-05-15T17:52:21.304000+00:00",
    "StandardsControlTitle": "CloudTrail should be enabled and configured
with at least one multi-Region trail that includes read and write management
events",
    "StandardsControlDescription": "This AWS control checks that there is
at least one multi-region AWS CloudTrail trail includes read and write management
events."
  },
  {
    "StandardsArn": "arn:aws:securityhub::ruleset/cis-aws-foundations-
benchmark/v/1.2.0",
    "SecurityControlId": "CloudTrail.1",
    "SecurityControlArn": "arn:aws:securityhub:us-
east-2:123456789012:security-control/CloudTrail.1",
    "AssociationStatus": "ENABLED",
    "RelatedRequirements": [
      "CIS AWS Foundations 2.1"
    ],
    "UpdatedAt": "2020-02-10T21:22:53.998000+00:00",
    "StandardsControlTitle": "Ensure CloudTrail is enabled in all regions",
    "StandardsControlDescription": "AWS CloudTrail is a web service that
records AWS API calls for your account and delivers log files to you. The recorded
information includes the identity of the API caller, the time of the API call,
the source IP address of the API caller, the request parameters, and the response
elements returned by the AWS service."
  },
  {
    "StandardsArn": "arn:aws:securityhub:us-east-2::standards/aws-
foundational-security-best-practices/v/1.0.0",
    "SecurityControlId": "CloudTrail.1",
    "SecurityControlArn": "arn:aws:securityhub:us-
east-2:123456789012:security-control/CloudTrail.1",
    "AssociationStatus": "DISABLED",
    "RelatedRequirements": [],
    "UpdatedAt": "2023-05-15T19:31:52.671000+00:00",
    "UpdatedReason": "Alternative compensating controls are in place",
    "StandardsControlTitle": "CloudTrail should be enabled and configured
with at least one multi-Region trail that includes read and write management
events",
    "StandardsControlDescription": "This AWS control checks that there is
at least one multi-region AWS CloudTrail trail includes read and write management
events."
  },
  {
    {
```

```

    "StandardsArn": "arn:aws:securityhub:us-east-2::standards/cis-aws-
foundations-benchmark/v/1.4.0",
    "SecurityControlId": "CloudTrail.1",
    "SecurityControlArn": "arn:aws:securityhub:us-
east-2:123456789012:security-control/CloudTrail.1",
    "AssociationStatus": "ENABLED",
    "RelatedRequirements": [
        "CIS AWS Foundations Benchmark v1.4.0/3.1"
    ],
    "UpdatedAt": "2022-11-10T15:40:36.021000+00:00",
    "StandardsControlTitle": "Ensure CloudTrail is enabled in all regions",
    "StandardsControlDescription": "AWS CloudTrail is a web service that
records AWS API calls for your account and delivers log files to you. The recorded
information includes the identity of the API caller, the time of the API call,
the source IP address of the API caller, the request parameters, and the response
elements returned by the AWS service. CloudTrail provides a history of AWS API
calls for an account, including API calls made via the Management Console, SDKs,
command line tools, and higher-level AWS services (such as CloudFormation)."
    }
  ]
}

```

For more information, see [Enabling and disabling controls in specific standards](#) in the *AWS Security Hub User Guide*.

- For API details, see [ListStandardsControlAssociations](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To retrieve the tags assigned to a resource

The following `list-tags-for-resource` example returns the tags assigned to the specified hub resource.

```

aws securityhub list-tags-for-resource \
  --resource-arn "arn:aws:securityhub:us-west-1:123456789012:hub/default"

```

Output:

```
{
  "Tags": {
    "Department" : "Operations",
    "Area" : "USMidwest"
  }
}
```

For more information, see [AWS::SecurityHub::Hub](#) in the *AWS CloudFormation User Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

start-configuration-policy-association

The following code example shows how to use start-configuration-policy-association.

AWS CLI

Example 1: To associate a configuration policy

The following start-configuration-policy-association example associates the specified configuration policy with the specified organizational unit. A configuration may be associated with a target account, organizational unit, or the root.

```
aws securityhub start-configuration-policy-association \
  --configuration-policy-identifier "arn:aws:securityhub:eu-
central-1:123456789012:configuration-policy/a1b2c3d4-5678-90ab-cdef-EXAMPLE33333" \
  --target '{"OrganizationalUnitId": "ou-6hi7-8j91kl2m"}'
```

Output:

```
{
  "ConfigurationPolicyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
  "TargetId": "ou-6hi7-8j91kl2m",
  "TargetType": "ORGANIZATIONAL_UNIT",
  "AssociationType": "APPLIED",
  "UpdatedAt": "2023-11-29T17:40:52.468000+00:00",
  "AssociationStatus": "PENDING"
}
```

For more information, see [Creating and associating Security Hub configuration policies](#) in the *AWS Security Hub User Guide*.

Example 2: To associate a self-managed configuration

The following `start-configuration-policy-association` example associates a self-managed configuration with the specified account.

```
aws securityhub start-configuration-policy-association \  
  --configuration-policy-identifier "SELF_MANAGED_SECURITY_HUB" \  
  --target '{"OrganizationalUnitId": "123456789012"}'
```

Output:

```
{  
  "ConfigurationPolicyId": "SELF_MANAGED_SECURITY_HUB",  
  "TargetId": "123456789012",  
  "TargetType": "ACCOUNT",  
  "AssociationType": "APPLIED",  
  "UpdatedAt": "2023-11-29T17:40:52.468000+00:00",  
  "AssociationStatus": "PENDING"  
}
```

For more information, see [Creating and associating Security Hub configuration policies](#) in the *AWS Security Hub User Guide*.

- For API details, see [StartConfigurationPolicyAssociation](#) in *AWS CLI Command Reference*.

start-configuration-policy-disassociation

The following code example shows how to use `start-configuration-policy-disassociation`.

AWS CLI

Example 1: To disassociate a configuration policy

The following `start-configuration-policy-disassociation` example disassociates a configuration policy from the specified organizational unit. A configuration may be disassociated from a target account, organizational unit, or the root.

```
aws securityhub start-configuration-policy-disassociation \  
  --configuration-policy-identifier "arn:aws:securityhub:eu-  
central-1:123456789012:configuration-policy/a1b2c3d4-5678-90ab-cdef-EXAMPLE33333" \  
  --target '{"OrganizationalUnitId": "123456789012"}'
```

```
--target '{"OrganizationalUnitId": "ou-6hi7-8j91k12m"}'
```

This command produces no output.

For more information, see [Disassociating a configuration from accounts and OUs](#) in the *AWS Security Hub User Guide*.

Example 2: To disassociate a self-managed configuration

The following `start-configuration-policy-disassociation` example disassociates a self-managed configuration from the specified account.

```
aws securityhub start-configuration-policy-disassociation \  
  --configuration-policy-identifier "SELF_MANAGED_SECURITY_HUB" \  
  --target '{"AccountId": "123456789012"}'
```

This command produces no output.

For more information, see [Disassociating a configuration from accounts and OUs](#) in the *AWS Security Hub User Guide*.

- For API details, see [StartConfigurationPolicyDisassociation](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To assign a tag to a resource

The following `tag-resource` example assigns values for the Department and Area tags to the specified hub resource.

```
aws securityhub tag-resource \  
  --resource-arn "arn:aws:securityhub:us-west-1:123456789012:hub/default" \  
  --tags '{"Department":"Operations", "Area":"USMidwest"}'
```

This command produces no output.

For more information, see [AWS::SecurityHub::Hub](#) in the *AWS CloudFormation User Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag value from a resource

The following `untag-resource` example removes the Department tag from the specified hub resource.

```
aws securityhub untag-resource \  
  --resource-arn "arn:aws:securityhub:us-west-1:123456789012:hub/default" \  
  --tag-keys "Department"
```

This command produces no output.

For more information, see [AWS::SecurityHub::Hub](#) in the *AWS CloudFormation User Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-action-target

The following code example shows how to use `update-action-target`.

AWS CLI

To update a custom action

The following `update-action-target` example updates the name of the custom action identified by the specified ARN.

```
aws securityhub update-action-target \  
  --action-target-arn "arn:aws:securityhub:us-west-1:123456789012:action/custom/  
Remediation" \  
  --name "Send to remediation"
```

This command produces no output.

For more information, see [Creating a custom action and associating it with a CloudWatch Events rule](#) in the *AWS Security Hub User Guide*.

- For API details, see [UpdateActionTarget](#) in *AWS CLI Command Reference*.

update-configuration-policy

The following code example shows how to use update-configuration-policy.

AWS CLI

To update a configuration policy

The following update-configuration-policy example updates an existing configuration policy to use the specified settings.

```
aws securityhub update-configuration-policy \
  --identifier "arn:aws:securityhub:eu-central-1:508236694226:configuration-
policy/09f37766-57d8-4ede-9d33-5d8b0fecf70e" \
  --name "SampleConfigurationPolicyUpdated" \
  --description "SampleDescriptionUpdated" \
  --configuration-policy '{"SecurityHub": {"ServiceEnabled":
true, "EnabledStandardIdentifiers": ["arn:aws:securityhub:eu-
central-1::standards/aws-foundational-security-best-practices/
v/1.0.0","arn:aws:securityhub:::ruleset/cis-aws-foundations-benchmark/
v/1.2.0"],"SecurityControlsConfiguration":{"DisabledSecurityControlIdentifiers":
["CloudWatch.1"], "SecurityControlCustomParameters": [{"SecurityControlId":
"ACM.1", "Parameters": {"daysToExpiration": {"ValueType": "CUSTOM", "Value":
{"Integer": 21}}}]}}}' \
  --updated-reason "Disabling CloudWatch.1 and changing parameter value"
```

Output:

```
{
  "Arn": "arn:aws:securityhub:eu-central-1:123456789012:configuration-policy/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "Name": "SampleConfigurationPolicyUpdated",
  "Description": "SampleDescriptionUpdated",
  "UpdatedAt": "2023-11-28T20:28:04.494000+00:00",
  "CreatedAt": "2023-11-28T20:28:04.494000+00:00",
  "ConfigurationPolicy": {
    "SecurityHub": {
      "ServiceEnabled": true,
      "EnabledStandardIdentifiers": [
```

```

        "arn:aws:securityhub:eu-central-1::standards/aws-foundational-
security-best-practices/v/1.0.0",
        "arn:aws:securityhub:::ruleset/cis-aws-foundations-benchmark/
v/1.2.0"
    ],
    "SecurityControlsConfiguration": {
        "DisabledSecurityControlIdentifiers": [
            "CloudWatch.1"
        ],
        "SecurityControlCustomParameters": [
            {
                "SecurityControlId": "ACM.1",
                "Parameters": {
                    "daysToExpiration": {
                        "ValueType": "CUSTOM",
                        "Value": {
                            "Integer": 21
                        }
                    }
                }
            }
        ]
    }
}

```

For more information, see [Updating Security Hub configuration policies](#) in the *AWS Security Hub User Guide*.

- For API details, see [UpdateConfigurationPolicy](#) in *AWS CLI Command Reference*.

update-finding-aggregator

The following code example shows how to use update-finding-aggregator.

AWS CLI

To update the current finding aggregation configuration

The following update-finding-aggregator example changes the finding aggregation configuration to link from selected Regions. It is run from US East (Virginia), which is the

aggregation Region. It selects US West (N. California) and US West (Oregon) as the linked Regions.

```
aws securityhub update-finding-aggregator \  
  --region us-east-1 \  
  --finding-aggregator-arn arn:aws:securityhub:us-east-1:222222222222:finding-  
aggregator/123e4567-e89b-12d3-a456-426652340000 \  
  --region-linking-mode SPECIFIED_REGIONS \  
  --regions us-west-1,us-west-2
```

This command produces no output.

For more information, see [Updating the finding aggregation configuration](#) in the *AWS Security Hub User Guide*.

- For API details, see [UpdateFindingAggregator](#) in *AWS CLI Command Reference*.

update-insight

The following code example shows how to use update-insight.

AWS CLI

Example 1: To change the filter for a custom insight

The following update-insight example changes the filters for a custom insight. The updated insight looks for findings with a high severity that are related to AWS roles.

```
aws securityhub update-insight \  
  --insight-arn "arn:aws:securityhub:us-west-1:123456789012:insight/123456789012/  
custom/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111" \  
  --filters '{"ResourceType": [{"Comparison": "EQUALS", "Value": "AwsIamRole"}],  
"SeverityLabel": [{"Comparison": "EQUALS", "Value": "HIGH"}]}' \  
  --name "High severity role findings"
```

Example 2: To change the grouping attribute for a custom insight

The following update-insight example changes the grouping attribute for the custom insight with the specified ARN. The new grouping attribute is the resource ID.

```
aws securityhub update-insight \  
  --insight-arn "arn:aws:securityhub:us-west-1:123456789012:insight/123456789012/  
custom/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111" \  
  --grouping-attribute "ResourceID" \  
  --name "High severity role findings"
```

```
--insight-arn "arn:aws:securityhub:us-west-1:123456789012:insight/123456789012/custom/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111" \  
--group-by-attribute "ResourceId" \  
--name "Critical role findings"
```

Output:

```
{  
  "Insights": [  
    {  
      "InsightArn": "arn:aws:securityhub:us-west-1:123456789012:insight/123456789012/custom/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
      "Name": "Critical role findings",  
      "Filters": {  
        "SeverityLabel": [  
          {  
            "Value": "CRITICAL",  
            "Comparison": "EQUALS"  
          }  
        ],  
        "ResourceType": [  
          {  
            "Value": "AwsIamRole",  
            "Comparison": "EQUALS"  
          }  
        ]  
      },  
      "GroupByAttribute": "ResourceId"  
    }  
  ]  
}
```

For more information, see [Managing custom insights](#) in the *AWS Security Hub User Guide*.

- For API details, see [UpdateInsight](#) in *AWS CLI Command Reference*.

update-organization-configuration

The following code example shows how to use `update-organization-configuration`.

AWS CLI

To update how Security Hub is configured for an organization

The following `update-organization-configuration` example specifies that Security Hub should use central configuration to configure an organization. After running this command, the delegated Security Hub administrator can create and manage configuration policies to configure the organization. The delegated administrator can also use this command to switch from central to local configuration. If local configuration is the configuration type, the delegated administrator can choose whether to automatically enable Security Hub and default security standards in new organization accounts.

```
aws securityhub update-organization-configuration \  
  --no-auto-enable \  
  --organization-configuration '{"ConfigurationType": "CENTRAL"}'
```

This command produces no output.

For more information, see [Managing accounts with AWS Organizations](#) in the *AWS Security Hub User Guide*.

- For API details, see [UpdateOrganizationConfiguration](#) in *AWS CLI Command Reference*.

update-security-control

The following code example shows how to use `update-security-control`.

AWS CLI

To update security control properties

The following `update-security-control` example specifies custom values for a Security Hub security control parameter.

```
aws securityhub update-security-control \  
  --security-control-id ACM.1 \  
  --parameters '{"daysToExpiration": {"ValueType": "CUSTOM", "Value": {"Integer":  
15}}}' \  
  --last-update-reason "Internal compliance requirement"
```

This command produces no output.

For more information, see [Custom control parameters](#) in the *AWS Security Hub User Guide*.

- For API details, see [UpdateSecurityControl](#) in *AWS CLI Command Reference*.

update-security-hub-configuration

The following code example shows how to use `update-security-hub-configuration`.

AWS CLI

To update Security Hub configuration

The following `update-security-hub-configuration` example configures Security Hub to automatically enable new controls for enabled standards.

```
aws securityhub update-security-hub-configuration \  
  --auto-enable-controls
```

This command produces no output.

For more information, see [Enabling new controls automatically](#) in the *AWS Security Hub User Guide*.

- For API details, see [UpdateSecurityHubConfiguration](#) in *AWS CLI Command Reference*.

update-standards-control

The following code example shows how to use `update-standards-control`.

AWS CLI

Example 1: To disable a control

The following `update-standards-control` example disables the `PCI.AutoScaling.1` control.

```
aws securityhub update-standards-control \  
  --standards-control-arn "arn:aws:securityhub:us-west-1:123456789012:control/pci-  
dss/v/3.2.1/PCI.AutoScaling.1" \  
  --control-status "DISABLED" \  
  --disabled-reason "Not applicable for my service"
```

This command produces no output.

Example 2: To enable a control

The following `update-standards-control` example enables the `PCI.AutoScaling.1` control.

```
aws securityhub update-standards-control \  
  --standards-control-arn "arn:aws:securityhub:us-west-1:123456789012:control/pci-  
dss/v/3.2.1/PCI.AutoScaling.1" \  
  --control-status "ENABLED"
```

This command produces no output.

For more information, see [Disabling and enabling individual controls](#) in the *AWS Security Hub User Guide*.

- For API details, see [UpdateStandardsControl](#) in *AWS CLI Command Reference*.

AWS Serverless Application Repository examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Serverless Application Repository.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

put-application-policy

The following code example shows how to use `put-application-policy`.

AWS CLI

Example 1: To share an application publicly

The following `put-application-policy` shares an application publicly, so anyone can find and deploy your application in the AWS Serverless Application Repository.

```
aws serverlessrepo put-application-policy \  
  --application-id arn:aws:serverlessrepo:us-east-1:123456789012:applications/my-  
test-application \  
  --statements Principals='*',Actions=Deploy
```

Output:

```
{  
  "Statements": [  
    {  
      "Actions": [  
        "Deploy"  
      ],  
      "Principals": [  
        ""  
      ],  
      "StatementId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"  
    }  
  ]  
}
```

Example 2: To share an application privately

The following `put-application-policy` shares an application privately, so only specific AWS accounts can find and deploy your application in the AWS Serverless Application Repository.

```
aws serverlessrepo put-application-policy \  
  --application-id arn:aws:serverlessrepo:us-east-1:123456789012:applications/my-  
test-application \  
  --statements Principals=111111111111,222222222222,Actions=Deploy
```

Output:

```
{  
  "Statements": [  

```

```
    {
      "Actions": [
        "Deploy"
      ],
      "Principals": [
        "111111111111",
        "222222222222"
      ],
      "StatementId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"
    }
  ]
}
```

For more information, see [Sharing an Application Through the Console](#) in the *AWS Serverless Application Repository Developer Guide*

- For API details, see [PutApplicationPolicy](#) in *AWS CLI Command Reference*.

Service Catalog examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Service Catalog.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

accept-portfolio-share

The following code example shows how to use `accept-portfolio-share`.

AWS CLI

To accept a portfolio share

The following `accept-portfolio-share` example accepts an offer, made by another user, to share the specified portfolio.

```
aws servicecatalog accept-portfolio-share \  
  --portfolio-id port-2s6wuabcdefghijk
```

This command produces no output.

- For API details, see [AcceptPortfolioShare](#) in *AWS CLI Command Reference*.

associate-principal-with-portfolio

The following code example shows how to use `associate-principal-with-portfolio`.

AWS CLI

To associate a principal with a portfolio

The following `associate-principal-with-portfolio` example associates a user with the specified portfolio.

```
aws servicecatalog associate-principal-with-portfolio \  
  --portfolio-id port-2s6abcdefwdh4 \  
  --principal-arn arn:aws:iam::123456789012:user/usertest \  
  --principal-type IAM
```

This command produces no output.

- For API details, see [AssociatePrincipalWithPortfolio](#) in *AWS CLI Command Reference*.

associate-product-with-portfolio

The following code example shows how to use `associate-product-with-portfolio`.

AWS CLI

To associate a product with a portfolio

The following `associate-product-with-portfolio` example associates the given product with the specified portfolio.

```
aws servicecatalog associate-product-with-portfolio
  --product-id prod-3p5abcdef3oyk
  --portfolio-id port-2s6abcdef5wdh4
```

This command produces no output.

- For API details, see [AssociateProductWithPortfolio](#) in *AWS CLI Command Reference*.

associate-tag-option-with-resource

The following code example shows how to use `associate-tag-option-with-resource`.

AWS CLI

To associate a TagOption with a resource

The following `associate-tag-option-with-resource` example associates the specified TagOption with the specified resource.

```
aws servicecatalog associate-tag-option-with-resource \
  --resource-id port-2s6abcdq5wdh4 \
  --tag-option-id tag-p3abc2pkpz5qc
```

This command produces no output.

- For API details, see [AssociateTagOptionWithResource](#) in *AWS CLI Command Reference*.

copy-product

The following code example shows how to use `copy-product`.

AWS CLI

To copy a product

The following `copy-product` example makes a copy of the specified product, using a JSON file to pass parameters.

```
aws servicecatalog copy-product --cli-input-json file:///copy-product-input.json
```

Contents of `copy-product-input.json`:

```
{
  "SourceProductArn": "arn:aws:catalog:us-west-2:123456789012:product/prod-
tcabcd3syn2xy",
  "TargetProductName": "copy-of-myproduct",
  "CopyOptions": [
    "CopyTags"
  ]
}
```

Output:

```
{
  "CopyProductToken": "copyproduct-abc5defgjkdji"
}
```

- For API details, see [CopyProduct](#) in *AWS CLI Command Reference*.

create-portfolio-share

The following code example shows how to use `create-portfolio-share`.

AWS CLI

To share a portfolio with an account

The following `create-portfolio-share` example shares the specified portfolio with the specified account.

```
aws servicecatalog create-portfolio-share \
  --portfolio-id port-2s6abcdef5wdh4 \
  --account-id 794123456789
```

This command produces no output.

- For API details, see [CreatePortfolioShare](#) in *AWS CLI Command Reference*.

create-portfolio

The following code example shows how to use `create-portfolio`.

AWS CLI

To create a portfolio

The following `create-portfolio` example creates a portfolio.

```
aws servicecatalog create-portfolio \
  --provider-name my-provider \
  --display-name my-portfolio
```

Output:

```
{
  "PortfolioDetail": {
    "ProviderName": "my-provider",
    "DisplayName": "my-portfolio",
    "CreatedTime": 1571337221.555,
    "ARN": "arn:aws:catalog:us-east-2:123456789012:portfolio/
port-2s6xmplq5wdh4",
    "Id": "port-2s6xmplq5wdh4"
  }
}
```

- For API details, see [CreatePortfolio](#) in *AWS CLI Command Reference*.

create-product

The following code example shows how to use `create-product`.

AWS CLI

To create a product

The following `create-product` example creates a product, using a JSON file to pass parameters.

```
aws servicecatalog create-product \
  --cli-input-json file://create-product-input.json
```

Contents of `create-product-input.json`:

```
{
  "AcceptLanguage": "en",
  "Name": "test-product",
  "Owner": "test-owner",
  "Description": "test-description",
  "Distributor": "test-distributor",
  "SupportDescription": "test-support",
  "SupportEmail": "test@amazon.com",
  "SupportUrl": "https://aws.amazon.com",
  "ProductType": "CLOUD_FORMATION_TEMPLATE",
  "Tags": [
    {
      "Key": "region",
      "Value": "us-east-1"
    }
  ],
  "ProvisioningArtifactParameters": {
    "Name": "test-version-name",
    "Description": "test-version-description",
    "Info": {
      "LoadTemplateFromURL": "https://s3-us-west-1.amazonaws.com/
cloudformation-templates-us-west-1/my-cfn-template.template"
    },
    "Type": "CLOUD_FORMATION_TEMPLATE"
  }
}
```

Output:

```
{
  "Tags": [
    {
      "Key": "region",
      "Value": "us-east-1"
    }
  ],
  "ProductViewDetail": {
    "CreatedTime": 1576025036.0,
    "ProductARN": "arn:aws:catalog:us-west-2:1234568542028:product/
prod-3p5abcdef3oyk",
    "Status": "CREATED",
    "ProductViewSummary": {
      "Type": "CLOUD_FORMATION_TEMPLATE",
```

```
    "Distributor": "test-distributor",
    "SupportUrl": "https://aws.amazon.com",
    "SupportEmail": "test@amazon.com",
    "Id": "prodview-abcd42wvx45um",
    "SupportDescription": "test-support",
    "ShortDescription": "test-description",
    "Owner": "test-owner",
    "Name": "test-product2",
    "HasDefaultPath": false,
    "ProductId": "prod-3p5abcdef3oyk"
  }
},
"ProvisioningArtifactDetail": {
  "CreatedTime": 1576025036.0,
  "Active": true,
  "Id": "pa-pq3p5lil12a34",
  "Description": "test-version-description",
  "Name": "test-version-name",
  "Type": "CLOUD_FORMATION_TEMPLATE"
}
}
```

- For API details, see [CreateProduct](#) in *AWS CLI Command Reference*.

create-provisioning-artifact

The following code example shows how to use create-provisioning-artifact.

AWS CLI

To create a provisioning artifact

The following create-provisioning-artifact example creates a provisioning artifact, using a JSON file to pass parameters.

```
aws servicecatalog create-provisioning-artifact \
  --cli-input-json file://create-provisioning-artifact-input.json
```

Contents of create-provisioning-artifact-input.json:

```
{
  "ProductId": "prod-nfi2abcdefghi",
```

```
"Parameters": {
  "Name": "test-provisioning-artifact",
  "Description": "test description",
  "Info": {
    "LoadTemplateFromURL": "https://s3-us-west-1.amazonaws.com/
cloudformation-templates-us-west-1/my-cfn-template.template"
  },
  "Type": "CLOUD_FORMATION_TEMPLATE"
}
}
```

Output:

```
{
  "Info": {
    "TemplateUrl": "https://s3-us-west-1.amazonaws.com/cloudformation-templates-
us-west-1/my-cfn-template.template"
  },
  "Status": "CREATING",
  "ProvisioningArtifactDetail": {
    "Id": "pa-bb4abcdefwnaio",
    "Name": "test-provisioning-artifact",
    "Description": "test description",
    "Active": true,
    "Type": "CLOUD_FORMATION_TEMPLATE",
    "CreatedTime": 1576022545.0
  }
}
```

- For API details, see [CreateProvisioningArtifact](#) in *AWS CLI Command Reference*.

create-tag-option

The following code example shows how to use create-tag-option.

AWS CLI

To create a TagOption

The following create-tag-option example creates a TagOption.

```
aws servicecatalog create-tag-option
```

```
--key 1234
--value name
```

Output:

```
{
  "TagOptionDetail": {
    "Id": "tag-iabcdn4fzjjms",
    "Value": "name",
    "Active": true,
    "Key": "1234"
  }
}
```

- For API details, see [CreateTagOption](#) in *AWS CLI Command Reference*.

delete-portfolio-share

The following code example shows how to use `delete-portfolio-share`.

AWS CLI

To stop sharing a portfolio with an account

The following `delete-portfolio-share` example stops sharing the portfolio with the specified account.

```
aws servicecatalog delete-portfolio-share \
  --portfolio-id port-2s6abcdq5wdh4 \
  --account-id 123456789012
```

This command produces no output.

- For API details, see [DeletePortfolioShare](#) in *AWS CLI Command Reference*.

delete-portfolio

The following code example shows how to use `delete-portfolio`.

AWS CLI

To delete a portfolio

The following `delete-portfolio` example deletes the specified portfolio.

```
aws servicecatalog delete-portfolio \  
  --id port-abcdlx4gox4do
```

This command produces no output.

- For API details, see [DeletePortfolio](#) in *AWS CLI Command Reference*.

delete-product

The following code example shows how to use `delete-product`.

AWS CLI

To delete a product

The following `delete-product` example deletes the specified product.

```
aws servicecatalog delete-product \  
  --id prod-abcdcek6yhbxi
```

This command produces no output.

- For API details, see [DeleteProduct](#) in *AWS CLI Command Reference*.

delete-provisioning-artifact

The following code example shows how to use `delete-provisioning-artifact`.

AWS CLI

To delete a provisioning artifact

The following `delete-provisioning-artifact` example deletes the specified provisioning artifact.

```
aws servicecatalog delete-provisioning-artifact \  
  --product-id prod-abc2uebuplcpw \  
  --provisioning-artifact-id pa-pqabccddii7ouc
```

This command produces no output.

- For API details, see [DeleteProvisioningArtifact](#) in *AWS CLI Command Reference*.

delete-tag-option

The following code example shows how to use `delete-tag-option`.

AWS CLI

To delete a TagOption

The following `delete-tag-option` example deletes the specified TagOption.

```
aws servicecatalog delete-tag-option \  
  --id tag-iabcdn4fzjjms
```

This command produces no output.

- For API details, see [DeleteTagOption](#) in *AWS CLI Command Reference*.

describe-copy-product-status

The following code example shows how to use `describe-copy-product-status`.

AWS CLI

To describe the status of the copy product operation

The following `describe-copy-product-status` example displays the current status of the specified asynchronous copy product operation.

```
aws servicecatalog describe-copy-product-status \  
  --copy-product-token copyproduct-znn5tf5abcd3w
```

Output:

```
{  
  "CopyProductStatus": "SUCCEEDED",  
  "TargetProductId": "prod-os6hog7abcdt2"  
}
```

- For API details, see [DescribeCopyProductStatus](#) in *AWS CLI Command Reference*.

describe-portfolio

The following code example shows how to use describe-portfolio.

AWS CLI

To describe a portfolio

The following describe-portfolio example displays details for the specified portfolio.

```
aws servicecatalog describe-portfolio \  
  --id port-2s6abcdq5wdh4
```

Output:

```
{  
  "TagOptions": [],  
  "PortfolioDetail": {  
    "ARN": "arn:aws:catalog:us-west-2:687558541234:portfolio/  
port-2s6abcdq5wdh4",  
    "Id": "port-2s6wuzyzq5wdh4",  
    "CreatedTime": 1571337221.555,  
    "DisplayName": "my-portfolio",  
    "ProviderName": "my-provider"  
  },  
  "Tags": []  
}
```

- For API details, see [DescribePortfolio](#) in *AWS CLI Command Reference*.

describe-product-as-admin

The following code example shows how to use describe-product-as-admin.

AWS CLI

To describe a product as an administrator

The following describe-product-as-admin example displays details for the specified product using administrator privileges.

```
aws servicecatalog describe-product-as-admin \  
  --id port-2s6abcdq5wdh4
```

```
--id prod-abcdcek6yhbxix
```

Output:

```
{
  "TagOptions": [],
  "ProductViewDetail": {
    "ProductARN": "arn:aws:catalog:us-west-2:687558542028:product/prod-
abcdcek6yhbxix",
    "ProductViewSummary": {
      "SupportEmail": "test@amazon.com",
      "Type": "CLOUD_FORMATION_TEMPLATE",
      "Distributor": "test-distributor",
      "ShortDescription": "test-description",
      "Owner": "test-owner",
      "Id": "prodview-wi3l2j4abc6vc",
      "SupportDescription": "test-support",
      "ProductId": "prod-abcdcek6yhbxix",
      "HasDefaultPath": false,
      "Name": "test-product3",
      "SupportUrl": "https://aws.amazon.com"
    },
    "CreatedTime": 1577136715.0,
    "Status": "CREATED"
  },
  "ProvisioningArtifactSummaries": [
    {
      "CreatedTime": 1577136715.0,
      "Description": "test-version-description",
      "ProvisioningArtifactMetadata": {
        "SourceProvisioningArtifactId": "pa-abcdxkkiv5fcm"
      },
      "Name": "test-version-name-3",
      "Id": "pa-abcdxkkiv5fcm"
    }
  ],
  "Tags": [
    {
      "Value": "iad",
      "Key": "region"
    }
  ]
}
```

- For API details, see [DescribeProductAsAdmin](#) in *AWS CLI Command Reference*.

describe-provisioned-product

The following code example shows how to use `describe-provisioned-product`.

AWS CLI

To describe a provisioned product

The following `describe-provisioned-product` example displays details for the specified provisioned product.

```
aws servicecatalog describe-provisioned-product \  
  --id pp-dpom27bm4abcd
```

Output:

```
{  
  "ProvisionedProductDetail": {  
    "Status": "ERROR",  
    "CreatedTime": 1577222793.358,  
    "Arn": "arn:aws:servicecatalog:us-west-2:123456789012:stack/mytestppname3/  
pp-dpom27bm4abcd",  
    "Id": "pp-dpom27bm4abcd",  
    "StatusMessage": "AmazonCloudFormationException Parameters: [KeyName]  
must have values (Service: AmazonCloudFormation; Status Code: 400; Error Code:  
ValidationError; Request ID: 5528602a-a9ef-427c-825c-f82c31b814f5)",  
    "IdempotencyToken": "527c5358-2a1a-4b9e-b1b9-7293b0ddff42",  
    "LastRecordId": "rec-tfuawdjovzxge",  
    "Type": "CFN_STACK",  
    "Name": "mytestppname3"  
  },  
  "CloudWatchDashboards": []  
}
```

- For API details, see [DescribeProvisionedProduct](#) in *AWS CLI Command Reference*.

describe-provisioning-artifact

The following code example shows how to use `describe-provisioning-artifact`.

AWS CLI

To describe a provisioning artifact

The following `describe-provisioning-artifact` example displays details for the specified provisioning artifact.

```
aws servicecatalog describe-provisioning-artifact \  
  --provisioning-artifact-id pa-pcz347abcdcfm \  
  --product-id prod-abcdfz3syn2rg
```

Output:

```
{  
  "Info": {  
    "TemplateUrl": "https://awsdocs.s3.amazonaws.com/servicecatalog/  
myexampledevelopment-environment.template"  
  },  
  "ProvisioningArtifactDetail": {  
    "Id": "pa-pcz347abcdcfm",  
    "Active": true,  
    "Type": "CLOUD_FORMATION_TEMPLATE",  
    "Description": "updated description",  
    "CreatedTime": 1562097906.0,  
    "Name": "updated name"  
  },  
  "Status": "AVAILABLE"  
}
```

- For API details, see [DescribeProvisioningArtifact](#) in *AWS CLI Command Reference*.

describe-tag-option

The following code example shows how to use `describe-tag-option`.

AWS CLI

To describe a TagOption

The following `describe-tag-option` example displays details for the specified TagOption.

```
aws servicecatalog describe-tag-option \  
  --tag-option-id tag-option-id
```

```
--id tag-p3tej2abcd5qc
```

Output:

```
{
  "TagOptionDetail": {
    "Active": true,
    "Id": "tag-p3tej2abcd5qc",
    "Value": "value-3",
    "Key": "1234"
  }
}
```

- For API details, see [DescribeTagOption](#) in *AWS CLI Command Reference*.

disassociate-principal-from-portfolio

The following code example shows how to use `disassociate-principal-from-portfolio`.

AWS CLI**To disassociate a principal from a portfolio**

The following `disassociate-principal-from-portfolio` example disassociates the specified principal from the portfolio.

```
aws servicecatalog disassociate-principal-from-portfolio \
  --portfolio-id port-2s6abcdq5wdh4 \
  --principal-arn arn:aws:iam::123456789012:group/myendusers
```

This command produces no output.

- For API details, see [DisassociatePrincipalFromPortfolio](#) in *AWS CLI Command Reference*.

disassociate-product-from-portfolio

The following code example shows how to use `disassociate-product-from-portfolio`.

AWS CLI**To disassociate a product from a portfolio**

The following `disassociate-product-from-portfolio` example disassociates the specified product from the portfolio.

```
aws servicecatalog disassociate-product-from-portfolio \  
  --product-id prod-3p5abcdmu3oyk \  
  --portfolio-id port-2s6abcdq5wdh4
```

This command produces no output.

- For API details, see [DisassociateProductFromPortfolio](#) in *AWS CLI Command Reference*.

disassociate-tag-option-from-resource

The following code example shows how to use `disassociate-tag-option-from-resource`.

AWS CLI

To disassociate a TagOption from a resource

The following `disassociate-tag-option-from-resource` example disassociates the specified `TagOption` from the resource.

```
aws servicecatalog disassociate-tag-option-from-resource \  
  --resource-id port-2s6abcdq5wdh4 \  
  --tag-option-id tag-p3abc2pkpz5qc
```

This command produces no output.

- For API details, see [DisassociateTagOptionFromResource](#) in *AWS CLI Command Reference*.

list-accepted-portfolio-shares

The following code example shows how to use `list-accepted-portfolio-shares`.

AWS CLI

To list accepted portfolio shares

The following `list-accepted-portfolio-shares` example lists all portfolios for which sharing was accepted by this account, including only the default Service Catalog portfolios.

```
aws servicecatalog list-accepted-portfolio-shares \
  --portfolio-share-type "AWS_SERVICECATALOG"
```

Output:

```
{
  "PortfolioDetails": [
    {
      "ARN": "arn:aws:catalog:us-west-2:123456789012:portfolio/port-
d2abcd5dpkuma",
      "Description": "AWS Service Catalog Reference blueprints for often-used
AWS services such as EC2, S3, RDS, VPC and EMR.",
      "CreatedTime": 1574456190.687,
      "ProviderName": "AWS Service Catalog",
      "DisplayName": "Reference Architectures",
      "Id": "port-d2abcd5dpkuma"
    },
    {
      "ARN": "arn:aws:catalog:us-west-2:123456789012:portfolio/port-
abcdefaua7zpu",
      "Description": "AWS well-architected blueprints for high reliability
applications.",
      "CreatedTime": 1574461496.092,
      "ProviderName": "AWS Service Catalog",
      "DisplayName": "High Reliability Architectures",
      "Id": "port-abcdefaua7zpu"
    }
  ]
}
```

- For API details, see [ListAcceptedPortfolioShares](#) in *AWS CLI Command Reference*.

list-portfolio-access

The following code example shows how to use `list-portfolio-access`.

AWS CLI

To list accounts with access to a portfolio

The following `list-portfolio-access` example lists the AWS accounts that have access to the specified portfolio.


```
aws servicecatalog list-portfolio-access \  
  --portfolio-id port-2s6abcdq5wdh4
```

Output:

```
{  
  "AccountIds": [  
    "123456789012"  
  ]  
}
```

- For API details, see [ListPortfolioAccess](#) in *AWS CLI Command Reference*.

list-portfolios-for-product

The following code example shows how to use `list-portfolios-for-product`.

AWS CLI

To list portfolios associated with a product

The following `list-portfolios-for-product` example lists the portfolios associated with the specified product.

```
aws servicecatalog list-portfolios-for-product \  
  --product-id prod-abcdefz3syn2rg
```

Output:

```
{  
  "PortfolioDetails": [  
    {  
      "CreatedTime": 1571337221.555,  
      "Id": "port-2s6abcdq5wdh4",  
      "ARN": "arn:aws:catalog:us-west-2:123456789012:portfolio/  
port-2s6abcdq5wdh4",  
      "DisplayName": "my-portfolio",  
      "ProviderName": "my-provider"  
    },  
    {
```

```
        "CreatedTime": 1559665256.348,  
        "Id": "port-5abcd3e5st4ei",  
        "ARN": "arn:aws:catalog:us-west-2:123456789012:portfolio/  
port-5abcd3e5st4ei",  
        "DisplayName": "test",  
        "ProviderName": "provider-name"  
    }  
]  
}
```

- For API details, see [ListPortfoliosForProduct](#) in *AWS CLI Command Reference*.

list-portfolios

The following code example shows how to use `list-portfolios`.

AWS CLI

To list portfolios

The following `list-portfolios` example lists the Service Catalog portfolios in the current Region.

```
aws servicecatalog list-portfolios
```

Output:

```
{  
  "PortfolioDetails": [  
    {  
      "CreatedTime": 1559665256.348,  
      "ARN": "arn:aws:catalog:us-east-2:123456789012:portfolio/  
port-5pzcxmlst4ei",  
      "DisplayName": "my-portfolio",  
      "Id": "port-5pzcxmlst4ei",  
      "ProviderName": "my-user"  
    }  
  ]  
}
```

- For API details, see [ListPortfolios](#) in *AWS CLI Command Reference*.

list-principals-for-portfolio

The following code example shows how to use `list-principals-for-portfolio`.

AWS CLI

To list all principals for a portfolio

The following `list-principals-for-portfolio` example lists all principals for the specified portfolio.

```
aws servicecatalog list-principals-for-portfolio \  
  --portfolio-id port-2s6abcdq5wdh4
```

Output:

```
{  
  "Principals": [  
    {  
      "PrincipalARN": "arn:aws:iam::123456789012:user/usertest",  
      "PrincipalType": "IAM"  
    }  
  ]  
}
```

- For API details, see [ListPrincipalsForPortfolio](#) in *AWS CLI Command Reference*.

list-provisioning-artifacts

The following code example shows how to use `list-provisioning-artifacts`.

AWS CLI

To list all provisioning artifacts for a product

The following `list-provisioning-artifacts` example lists all provisioning artifacts for the specified product.

```
aws servicecatalog list-provisioning-artifacts \  
  --product-id prod-nfi2abcdefgcpw
```

Output:

```
{
  "ProvisioningArtifactDetails": [
    {
      "Id": "pa-abcdef54ipm6z",
      "Description": "test-version-description",
      "Type": "CLOUD_FORMATION_TEMPLATE",
      "CreatedTime": 1576021147.0,
      "Active": true,
      "Name": "test-version-name"
    },
    {
      "Id": "pa-bb4zyxwwnaio",
      "Description": "test description",
      "Type": "CLOUD_FORMATION_TEMPLATE",
      "CreatedTime": 1576022545.0,
      "Active": true,
      "Name": "test-provisioning-artifact-2"
    }
  ]
}
```

- For API details, see [ListProvisioningArtifacts](#) in *AWS CLI Command Reference*.

list-resources-for-tag-option

The following code example shows how to use `list-resources-for-tag-option`.

AWS CLI

To list resources associated to a TagOption

The following `list-resources-for-tag-option` example lists the resources associated with the specified `TagOption`.

```
aws servicecatalog list-resources-for-tag-option \
  --tag-option-id tag-p3tej2abcd5qc
```

Output:

```
{
  "ResourceDetails": [
    {
```

```
    "ARN": "arn:aws:catalog:us-west-2:123456789012:product/prod-
abcdfz3syn2rg",
    "Name": "my product",
    "Description": "description",
    "CreatedTime": 1562097906.0,
    "Id": "prod-abcdfz3syn2rg"
  }
]
```

- For API details, see [ListResourcesForTagOption](#) in *AWS CLI Command Reference*.

list-tag-options

The following code example shows how to use `list-tag-options`.

AWS CLI

The following `list-tag-options` example lists all values for `TagOptions`.

```
aws servicecatalog list-tag-options
```

Output:

```
{
  "TagOptionDetails": [
    {
      "Value": "newvalue",
      "Active": true,
      "Id": "tag-iabcdn4fzjjms",
      "Key": "1234"
    },
    {
      "Value": "value1",
      "Active": true,
      "Id": "tag-e3abcdvmwvrzy",
      "Key": "key"
    }
  ]
}
```

- For API details, see [ListTagOptions](#) in *AWS CLI Command Reference*.

provision-product

The following code example shows how to use `provision-product`.

AWS CLI

To provision a product

The following `provision-product` example provisions the specified product using the specified provisioning artifact.

```
aws servicecatalog provision-product \  
  --product-id prod-abcdefz3syn2rg \  
  --provisioning-artifact-id pa-abc347pcscfm \  
  --provisioned-product-name "mytestppname3"
```

Output:

```
{  
  "RecordDetail": {  
    "RecordId": "rec-tfuawdabcdege",  
    "CreatedTime": 1577222793.362,  
    "ProvisionedProductId": "pp-abcd27bm4mldq",  
    "PathId": "lpv2-abcdg3jp6t5k6",  
    "RecordErrors": [],  
    "ProductId": "prod-abcdefz3syn2rg",  
    "UpdatedTime": 1577222793.362,  
    "RecordType": "PROVISION_PRODUCT",  
    "ProvisionedProductName": "mytestppname3",  
    "ProvisioningArtifactId": "pa-pcz347abcdcfm",  
    "RecordTags": [],  
    "Status": "CREATED",  
    "ProvisionedProductType": "CFN_STACK"  
  }  
}
```

- For API details, see [ProvisionProduct](#) in *AWS CLI Command Reference*.

reject-portfolio-share

The following code example shows how to use `reject-portfolio-share`.

AWS CLI

To reject a portfolio share

The following `reject-portfolio-share` example rejects the portfolio share for the given portfolio.

```
aws servicecatalog reject-portfolio-share \  
  --portfolio-id port-2s6wuabcdefghijk
```

This command produces no output.

- For API details, see [RejectPortfolioShare](#) in *AWS CLI Command Reference*.

scan-provisioned-products

The following code example shows how to use `scan-provisioned-products`.

AWS CLI

To list all available provisioned products

The following `scan-provisioned-products` example lists available provisioned products.

```
aws servicecatalog scan-provisioned-products
```

Output:

```
{  
  "ProvisionedProducts": [  
    {  
      "Status": "ERROR",  
      "Arn": "arn:aws:servicecatalog:us-west-2:123456789012:stack/  
mytestppname3/pp-abcd27bm4mldq",  
      "StatusMessage": "AmazonCloudFormationException Parameters: [KeyName]  
must have values (Service: AmazonCloudFormation; Status Code: 400; Error Code:  
ValidationError; Request ID: 5528602a-a9ef-427c-825c-f82c31b814f5)",  
      "Id": "pp-abcd27bm4mldq",  
      "Type": "CFN_STACK",  
      "IdempotencyToken": "527c5358-2a1a-4b9e-b1b9-7293b0ddff42",  
      "CreatedTime": 1577222793.358,  
    }  
  ]  
}
```

```
        "Name": "mytestppname3",
        "LastRecordId": "rec-tfuawdabcdxge"
    }
]
}
```

- For API details, see [ScanProvisionedProducts](#) in *AWS CLI Command Reference*.

search-products-as-admin

The following code example shows how to use `search-products-as-admin`.

AWS CLI

To search products with administrator privileges

The following `search-products-as-admin` example searches for products with admin privileges, using a portfolio ID as a filter.

```
aws servicecatalog search-products-as-admin \
  --portfolio-id port-5abcd3e5st4ei
```

Output:

```
{
  "ProductViewDetails": [
    {
      "ProductViewSummary": {
        "Name": "my product",
        "Owner": "owner name",
        "Type": "CLOUD_FORMATION_TEMPLATE",
        "ProductId": "prod-abcdefz3syn2rg",
        "HasDefaultPath": false,
        "Id": "prodview-abcdmyuzv2dlu",
        "ShortDescription": "description"
      },
      "ProductARN": "arn:aws:catalog:us-west-2:123456789012:product/prod-abcdefz3syn2rg",
      "CreatedTime": 1562097906.0,
      "Status": "CREATED"
    }
  ]
}
```



```
}
```

- For API details, see [SearchProductsAsAdmin](#) in *AWS CLI Command Reference*.

search-provisioned-products

The following code example shows how to use `search-provisioned-products`.

AWS CLI

To search provisioned products

The following `search-provisioned-products` example searches for provisioned products matching the specified product ID, using a JSON file to pass parameters.

```
aws servicecatalog search-provisioned-products \  
  --cli-input-json file://search-provisioned-products-input.json
```

Contents of `search-provisioned-products-input.json`:

```
{  
  "Filters": {  
    "SearchQuery": [  
      "prod-tcjevz3syn2rg"  
    ]  
  }  
}
```

Output:

```
{  
  "ProvisionedProducts": [  
    {  
      "ProvisioningArtifactId": "pa-pcz347abcdcfm",  
      "Name": "mytestppname3",  
      "CreatedTime": 1577222793.358,  
      "Id": "pp-abcd27bm4mldq",  
      "Status": "ERROR",  
      "UserArn": "arn:aws:iam::123456789012:user/cliuser",  
      "StatusMessage": "AmazonCloudFormationException Parameters: [KeyName]  
must have values (Service: AmazonCloudFormation; Status Code: 400; Error Code:  
ValidationError; Request ID: 5528602a-a9ef-427c-825c-f82c31b814f5)",
```

```

    "Arn": "arn:aws:servicecatalog:us-west-2:123456789012:stack/
mytestppname3/pp-abcd27bm4mldq",
    "Tags": [
      {
        "Value": "arn:aws:catalog:us-west-2:123456789012:product/prod-
abcdfz3syn2rg",
        "Key": "aws:servicecatalog:productArn"
      },
      {
        "Value": "arn:aws:iam::123456789012:user/cliuser",
        "Key": "aws:servicecatalog:provisioningPrincipalArn"
      },
      {
        "Value": "value-3",
        "Key": "1234"
      },
      {
        "Value": "pa-pcz347abcdcfm",
        "Key": "aws:servicecatalog:provisioningArtifactIdentifier"
      },
      {
        "Value": "arn:aws:catalog:us-west-2:123456789012:portfolio/
port-2s6abcdq5wdh4",
        "Key": "aws:servicecatalog:portfolioArn"
      },
      {
        "Value": "arn:aws:servicecatalog:us-west-2:123456789012:stack/
mytestppname3/pp-abcd27bm4mldq",
        "Key": "aws:servicecatalog:provisionedProductArn"
      }
    ],
    "IdempotencyToken": "527c5358-2a1a-4b9e-b1b9-7293b0ddff42",
    "UserArnSession": "arn:aws:iam::123456789012:user/cliuser",
    "Type": "CFN_STACK",
    "LastRecordId": "rec-tfuawdabcdxge",
    "ProductId": "prod-abcdfz3syn2rg"
  }
],
  "TotalResultsCount": 1
}

```

- For API details, see [SearchProvisionedProducts](#) in *AWS CLI Command Reference*.

update-portfolio

The following code example shows how to use `update-portfolio`.

AWS CLI

To update a portfolio

The following `update-portfolio` example updates the name of the specified portfolio.

```
aws servicecatalog update-portfolio \  
  --id port-5abcd3e5st4ei \  
  --display-name "New portfolio name"
```

Output:

```
{  
  "PortfolioDetail": {  
    "DisplayName": "New portfolio name",  
    "ProviderName": "provider",  
    "ARN": "arn:aws:catalog:us-west-2:123456789012:portfolio/  
port-5abcd3e5st4ei",  
    "Id": "port-5abcd3e5st4ei",  
    "CreatedTime": 1559665256.348  
  },  
  "Tags": []  
}
```

- For API details, see [UpdatePortfolio](#) in *AWS CLI Command Reference*.

update-product

The following code example shows how to use `update-product`.

AWS CLI

To update a product

The following `update-product` example updates the name and owner of the specified product.

```
aws servicecatalog update-product \  
  --id port-5abcd3e5st4ei \  
  --display-name "New portfolio name"
```

```
--id prod-os6abc7drqlt2 \  
--name "New product name" \  
--owner "Updated product owner"
```

Output:

```
{  
  "Tags": [  
    {  
      "Value": "iad",  
      "Key": "region"  
    }  
  ],  
  "ProductViewDetail": {  
    "ProductViewSummary": {  
      "Owner": "Updated product owner",  
      "ProductId": "prod-os6abc7drqlt2",  
      "Distributor": "test-distributor",  
      "SupportUrl": "https://aws.amazon.com",  
      "Name": "New product name",  
      "ShortDescription": "test-description",  
      "HasDefaultPath": false,  
      "Id": "prodview-6abcdgrfhvidy",  
      "SupportDescription": "test-support",  
      "SupportEmail": "test@amazon.com",  
      "Type": "CLOUD_FORMATION_TEMPLATE"  
    },  
    "Status": "CREATED",  
    "ProductARN": "arn:aws:catalog:us-west-2:123456789012:product/prod-  
os6abc7drqlt2",  
    "CreatedTime": 1577136255.0  
  }  
}
```

- For API details, see [UpdateProduct](#) in *AWS CLI Command Reference*.

update-provisioning-artifact

The following code example shows how to use `update-provisioning-artifact`.

AWS CLI

To update a provisioning artifact

The following `update-provisioning-artifact` example updates the name and description of the specified provisioning artifact, using a JSON file to pass parameters.

```
aws servicecatalog update-provisioning-artifact \  
  --cli-input-json file://update-provisioning-artifact-input.json
```

Contents of `update-provisioning-artifact-input.json`:

```
{  
  "ProductId": "prod-abcdefz3syn2rg",  
  "ProvisioningArtifactId": "pa-pcz347abcdcfm",  
  "Name": "updated name",  
  "Description": "updated description"  
}
```

Output:

```
{  
  "Info": {  
    "TemplateUrl": "https://awsdocs.s3.amazonaws.com/servicecatalog/  
myexampledevelopment-environment.template"  
  },  
  "Status": "AVAILABLE",  
  "ProvisioningArtifactDetail": {  
    "Active": true,  
    "Description": "updated description",  
    "Id": "pa-pcz347abcdcfm",  
    "Name": "updated name",  
    "Type": "CLOUD_FORMATION_TEMPLATE",  
    "CreatedTime": 1562097906.0  
  }  
}
```

- For API details, see [UpdateProvisioningArtifact](#) in *AWS CLI Command Reference*.

update-tag-option

The following code example shows how to use `update-tag-option`.

AWS CLI

To update a TagOption

The following `update-tag-option` example updates the value of a `TagOption`, using the specified JSON file.

```
aws servicecatalog update-tag-option --cli-input-json file://update-tag-option-input.json
```

Contents of `update-tag-option-input.json`:

```
{
  "Id": "tag-iabcdn4fzjjms",
  "Value": "newvalue",
  "Active": true
}
```

Output:

```
{
  "TagOptionDetail": {
    "Value": "newvalue",
    "Key": "1234",
    "Active": true,
    "Id": "tag-iabcdn4fzjjms"
  }
}
```

- For API details, see [UpdateTagOption](#) in *AWS CLI Command Reference*.

Service Quotas examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Service Quotas.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

get-aws-default-service-quota

The following code example shows how to use `get-aws-default-service-quota`.

AWS CLI

To describe a default service quota

The following `get-aws-default-service-quota` example displays details for the specified quota.

```
aws service-quotas get-aws-default-service-quota \  
  --service-code ec2 \  
  --quota-code L-1216C47A
```

Output:

```
{  
  "Quota": {  
    "ServiceCode": "ec2",  
    "ServiceName": "Amazon Elastic Compute Cloud (Amazon EC2)",  
    "QuotaArn": "arn:aws:servicequotas:us-east-2::ec2/L-1216C47A",  
    "QuotaCode": "L-1216C47A",  
    "QuotaName": "Running On-Demand Standard (A, C, D, H, I, M, R, T, Z)  
instances",  
    "Value": 5.0,  
    "Unit": "None",  
    "Adjustable": true,  
    "GlobalQuota": false,  
    "UsageMetric": {  
      "MetricNamespace": "AWS/Usage",  
      "MetricName": "ResourceCount",  
      "MetricDimensions": {  
        "Class": "Standard/OnDemand",  
        "Resource": "vCPU",  
        "Service": "EC2",  
        "Type": "Resource"  
      }  
    }  
  }  
}
```

```

        },
        "MetricStatisticRecommendation": "Maximum"
    }
}
}

```

- For API details, see [GetAwsDefaultServiceQuota](#) in *AWS CLI Command Reference*.

get-requested-service-quota-change

The following code example shows how to use `get-requested-service-quota-change`.

AWS CLI

To describe a service quota increase request

The following `get-requested-service-quota-change` example describes the specified quota increase request.

```
aws service-quotas get-requested-service-quota-change \
  --request-id d187537d15254312a9609aa51bbf7624u7W49tP0
```

Output:

```
{
  "RequestedQuota": {
    "Id": "d187537d15254312a9609aa51bbf7624u7W49tP0",
    "CaseId": "6780195351",
    "ServiceCode": "ec2",
    "ServiceName": "Amazon Elastic Compute Cloud (Amazon EC2)",
    "QuotaCode": "L-20F13EBD",
    "QuotaName": "Running Dedicated c5n Hosts",
    "DesiredValue": 2.0,
    "Status": "CASE_OPENED",
    "Created": 1580446904.067,
    "LastUpdated": 1580446953.265,
    "Requester": "{\"accountId\": \"123456789012\", \"callerArn\": \"arn:aws:iam::123456789012:root\"}",
    "QuotaArn": "arn:aws:servicequotas:us-east-2:123456789012:ec2/L-20F13EBD",
    "GlobalQuota": false,
    "Unit": "None"
  }
}
```



```
}
```

- For API details, see [GetRequestedServiceQuotaChange](#) in *AWS CLI Command Reference*.

get-service-quota

The following code example shows how to use `get-service-quota`.

AWS CLI

To describe a service quota

The following `get-service-quota` example displays details about the specified quota.

```
aws service-quotas get-service-quota \  
  --service-code ec2 \  
  --quota-code L-1216C47A
```

Output:

```
{  
  "Quota": {  
    "ServiceCode": "ec2",  
    "ServiceName": "Amazon Elastic Compute Cloud (Amazon EC2)",  
    "QuotaArn": "arn:aws:servicequotas:us-east-2:123456789012:ec2/L-1216C47A",  
    "QuotaCode": "L-1216C47A",  
    "QuotaName": "Running On-Demand Standard (A, C, D, H, I, M, R, T, Z)  
instances",  
    "Value": 1920.0,  
    "Unit": "None",  
    "Adjustable": true,  
    "GlobalQuota": false,  
    "UsageMetric": {  
      "MetricNamespace": "AWS/Usage",  
      "MetricName": "ResourceCount",  
      "MetricDimensions": {  
        "Class": "Standard/OnDemand",  
        "Resource": "vCPU",  
        "Service": "EC2",  
        "Type": "Resource"  
      },  
      "MetricStatisticRecommendation": "Maximum"  
    }  
  }  
}
```

```
    }  
  }  
}
```

- For API details, see [GetServiceQuota](#) in *AWS CLI Command Reference*.

list-aws-default-service-quotas

The following code example shows how to use `list-aws-default-service-quotas`.

AWS CLI

To list the default quotas for a service

The following `list-aws-default-service-quotas` example lists the default values for the quotas for the specified service.

```
aws service-quotas list-aws-default-service-quotas \  
  --service-code xray
```

Output:

```
{  
  "Quotas": [  
    {  
      "ServiceCode": "xray",  
      "ServiceName": "AWS X-Ray",  
      "QuotaArn": "arn:aws:servicequotas:us-west-2::xray/L-C6B6F05D",  
      "QuotaCode": "L-C6B6F05D",  
      "QuotaName": "Indexed annotations per trace",  
      "Value": 50.0,  
      "Unit": "None",  
      "Adjustable": false,  
      "GlobalQuota": false  
    },  
    {  
      "ServiceCode": "xray",  
      "ServiceName": "AWS X-Ray",  
      "QuotaArn": "arn:aws:servicequotas:us-west-2::xray/L-D781C0FD",  
      "QuotaCode": "L-D781C0FD",  
      "QuotaName": "Segment document size",  
      "Value": 64.0,  
    }  
  ]  
}
```

```

        "Unit": "Kilobytes",
        "Adjustable": false,
        "GlobalQuota": false
    },
    {
        "ServiceCode": "xray",
        "ServiceName": "AWS X-Ray",
        "QuotaArn": "arn:aws:servicequotas:us-west-2::xray/L-998BFF16",
        "QuotaCode": "L-998BFF16",
        "QuotaName": "Trace and service graph retention in days",
        "Value": 30.0,
        "Unit": "None",
        "Adjustable": false,
        "GlobalQuota": false
    }
]
}

```

- For API details, see [ListAwsDefaultServiceQuotas](#) in *AWS CLI Command Reference*.

list-requested-service-quota-change-history-by-quota

The following code example shows how to use `list-requested-service-quota-change-history-by-quota`.

AWS CLI

To list your quota increase requests

The following `list-requested-service-quota-change-history-by-quota` example lists the quota increase requests for the specified quota.

```

aws service-quotas list-requested-service-quota-change-history-by-quota \
  --service-code ec2 \
  --quota-code L-20F13EBD

```

Output:

```

{
  "RequestedQuotas": [
    {
      "Id": "d187537d15254312a9609aa51bbf7624u7W49tP0",

```

```

    "CaseId": "6780195351",
    "ServiceCode": "ec2",
    "ServiceName": "Amazon Elastic Compute Cloud (Amazon EC2)",
    "QuotaCode": "L-20F13EBD",
    "QuotaName": "Running Dedicated c5n Hosts",
    "DesiredValue": 2.0,
    "Status": "CASE_OPENED",
    "Created": 1580446904.067,
    "LastUpdated": 1580446953.265,
    "Requester": "{\"accountId\":\"123456789012\",\"callerArn\":
\\\"arn:aws:iam::123456789012:root\\\"}\",
    "QuotaArn": "arn:aws:servicequotas:us-east-2:123456789012:ec2/
L-20F13EBD",
    "GlobalQuota": false,
    "Unit": "None"
  }
]
}

```

- For API details, see [ListRequestedServiceQuotaChangeHistoryByQuota](#) in *AWS CLI Command Reference*.

list-requested-service-quota-change-history

The following code example shows how to use `list-requested-service-quota-change-history`.

AWS CLI

To list your quota increase requests

The following `list-requested-service-quota-change-history` example lists the quota increase requests for the specified service.

```
aws service-quotas list-requested-service-quota-change-history \
  --service-code ec2
```

Output:

```
{
  "RequestedQuotas": [
```

```

    {
      "Id": "d187537d15254312a9609aa51bbf7624u7W49tP0",
      "CaseId": "6780195351",
      "ServiceCode": "ec2",
      "ServiceName": "Amazon Elastic Compute Cloud (Amazon EC2)",
      "QuotaCode": "L-20F13EBD",
      "QuotaName": "Running Dedicated c5n Hosts",
      "DesiredValue": 2.0,
      "Status": "CASE_OPENED",
      "Created": 1580446904.067,
      "LastUpdated": 1580446953.265,
      "Requester": "{\"accountId\":\"123456789012\",\"callerArn\":
\\\"arn:aws:iam::123456789012:root\\\"}\",
      "QuotaArn": "arn:aws:servicequotas:us-east-2:123456789012:ec2/
L-20F13EBD",
      "GlobalQuota": false,
      "Unit": "None"
    }
  ]
}

```

- For API details, see [ListRequestedServiceQuotaChangeHistory](#) in *AWS CLI Command Reference*.

list-service-quotas

The following code example shows how to use `list-service-quotas`.

AWS CLI

To list the quotas for a service

The following `list-service-quotas` example displays details about the quotas for AWS CloudFormation.

```
aws service-quotas list-service-quotas \
  --service-code cloudformation
```

Output:

```
{
  "Quotas": [
```

```
{
  "ServiceCode": "cloudformation",
  "ServiceName": "AWS CloudFormation",
  "QuotaArn": "arn:aws:servicequotas:us-
east-2:123456789012:cloudformation/L-87D14FB7",
  "QuotaCode": "L-87D14FB7",
  "QuotaName": "Output count in CloudFormation template",
  "Value": 60.0,
  "Unit": "None",
  "Adjustable": false,
  "GlobalQuota": false
},
{
  "ServiceCode": "cloudformation",
  "ServiceName": "AWS CloudFormation",
  "QuotaArn": "arn:aws:servicequotas:us-
east-2:123456789012:cloudformation/L-0485CB21",
  "QuotaCode": "L-0485CB21",
  "QuotaName": "Stack count",
  "Value": 200.0,
  "Unit": "None",
  "Adjustable": true,
  "GlobalQuota": false
}
]
```

- For API details, see [ListServiceQuotas](#) in *AWS CLI Command Reference*.

list-services

The following code example shows how to use `list-services`.

AWS CLI

To list the available services

The following command lists the services that are available in Service Quotas.

```
aws service-quotas list-services
```

Output:

```
{
  "Services": [
    {
      "ServiceCode": "AWSCloudMap",
      "ServiceName": "AWS Cloud Map"
    },
    {
      "ServiceCode": "access-analyzer",
      "ServiceName": "Access Analyzer"
    },
    {
      "ServiceCode": "acm",
      "ServiceName": "AWS Certificate Manager (ACM)"
    },
    ...truncated...
    {
      "ServiceCode": "xray",
      "ServiceName": "AWS X-Ray"
    }
  ]
}
```

You can add the `--query` parameter to filter the display to the information that you are interested in. The following example displays only the service codes.

```
aws service-quotas list-services \
  --query Services[*].ServiceCode
```

Output:

```
[
  "AWSCloudMap",
  "access-analyzer",
  "acm",
  "acm-pca",
  "amplify",
  "apigateway",
  "application-autoscaling",
  ...truncated...
  "xray"
```

```
]
```

- For API details, see [ListServices](#) in *AWS CLI Command Reference*.

request-service-quota-increase

The following code example shows how to use `request-service-quota-increase`.

AWS CLI

To request a service quota increase

The following `request-service-quota-increase` example requests an increase in the specified service quota.

```
aws service-quotas request-service-quota-increase \  
  --service-code ec2 \  
  --quota-code L-20F13EBD \  
  --desired-value 2
```

Output:

```
{  
  "RequestedQuota": {  
    "Id": "d187537d15254312a9609aa51bbf7624u7W49tP0",  
    "ServiceCode": "ec2",  
    "ServiceName": "Amazon Elastic Compute Cloud (Amazon EC2)",  
    "QuotaCode": "L-20F13EBD",  
    "QuotaName": "Running Dedicated c5n Hosts",  
    "DesiredValue": 2.0,  
    "Status": "PENDING",  
    "Created": 1580446904.067,  
    "Requester": "{\"accountId\":\"123456789012\",\"callerArn\":  
  \\\narn:aws:iam::123456789012:root\"}",  
    "QuotaArn": "arn:aws:servicequotas:us-east-2:123456789012:ec2/L-20F13EBD",  
    "GlobalQuota": false,  
    "Unit": "None"  
  }  
}
```

- For API details, see [RequestServiceQuotaIncrease](#) in *AWS CLI Command Reference*.

Amazon SES examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon SES.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

delete-identity

The following code example shows how to use `delete-identity`.

AWS CLI

To delete an identity

The following example uses the `delete-identity` command to delete an identity from the list of identities verified with Amazon SES:

```
aws ses delete-identity --identity user@example.com
```

For more information about verified identities, see *Verifying Email Addresses and Domains in Amazon SES* in the *Amazon Simple Email Service Developer Guide*.

- For API details, see [DeleteIdentity](#) in *AWS CLI Command Reference*.

get-identity-dkim-attributes

The following code example shows how to use `get-identity-dkim-attributes`.

AWS CLI

To get the Amazon SES Easy DKIM attributes for a list of identities

The following example uses the `get-identity-dkim-attributes` command to retrieve the Amazon SES Easy DKIM attributes for a list of identities:

```
aws ses get-identity-dkim-attributes --identities "example.com" "user@example.com"
```

Output:

```
{
  "DkimAttributes": {
    "example.com": {
      "DkimTokens": [
        "EXAMPLEjcs5xoyqytjsotsijas7236gr",
        "EXAMPLEjr76cvoc6mysspnioorxsn6ep",
        "EXAMPLEkbnkqkhlm2lyz77ppkulerm4k"
      ],
      "DkimEnabled": true,
      "DkimVerificationStatus": "Success"
    },
    "user@example.com": {
      "DkimEnabled": false,
      "DkimVerificationStatus": "NotStarted"
    }
  }
}
```

If you call this command with an identity that you have never submitted for verification, that identity won't appear in the output.

For more information about Easy DKIM, see Easy DKIM in Amazon SES in the *Amazon Simple Email Service Developer Guide*.

- For API details, see [GetIdentityDkimAttributes](#) in *AWS CLI Command Reference*.

get-identity-notification-attributes

The following code example shows how to use `get-identity-notification-attributes`.

AWS CLI

To get the Amazon SES notification attributes for a list of identities

The following example uses the `get-identity-notification-attributes` command to retrieve the Amazon SES notification attributes for a list of identities:

```
aws ses get-identity-notification-attributes --identities "user1@example.com"
"user2@example.com"
```

Output:

```
{
  "NotificationAttributes": {
    "user1@example.com": {
      "ForwardingEnabled": false,
      "ComplaintTopic": "arn:aws:sns:us-east-1:EXAMPLE65304:MyTopic",
      "BounceTopic": "arn:aws:sns:us-east-1:EXAMPLE65304:MyTopic",
      "DeliveryTopic": "arn:aws:sns:us-east-1:EXAMPLE65304:MyTopic"
    },
    "user2@example.com": {
      "ForwardingEnabled": true
    }
  }
}
```

This command returns the status of email feedback forwarding and, if applicable, the Amazon Resource Names (ARNs) of the Amazon SNS topics that bounce, complaint, and delivery notifications are sent to.

If you call this command with an identity that you have never submitted for verification, that identity won't appear in the output.

For more information about notifications, see *Using Notifications With Amazon SES* in the *Amazon Simple Email Service Developer Guide*.

- For API details, see [GetIdentityNotificationAttributes](#) in *AWS CLI Command Reference*.

`get-identity-verification-attributes`

The following code example shows how to use `get-identity-verification-attributes`.

AWS CLI

To get the Amazon SES verification status for a list of identities

The following example uses the `get-identity-verification-attributes` command to retrieve the Amazon SES verification status for a list of identities:

```
aws ses get-identity-verification-attributes --identities "user1@example.com"
"user2@example.com"
```

Output:

```
{
  "VerificationAttributes": {
    "user1@example.com": {
      "VerificationStatus": "Success"
    },
    "user2@example.com": {
      "VerificationStatus": "Pending"
    }
  }
}
```

If you call this command with an identity that you have never submitted for verification, that identity won't appear in the output.

For more information about verified identities, see *Verifying Email Addresses and Domains in Amazon SES* in the *Amazon Simple Email Service Developer Guide*.

- For API details, see [GetIdentityVerificationAttributes](#) in *AWS CLI Command Reference*.

get-send-quota

The following code example shows how to use `get-send-quota`.

AWS CLI

To get your Amazon SES sending limits

The following example uses the `get-send-quota` command to return your Amazon SES sending limits:

```
aws ses get-send-quota
```

Output:

```
{
  "Max24HourSend": 200.0,
  "SentLast24Hours": 1.0,
  "MaxSendRate": 1.0
}
```

Max24HourSend is your sending quota, which is the maximum number of emails that you can send in a 24-hour period. The sending quota reflects a rolling time period. Every time you try to send an email, Amazon SES checks how many emails you sent in the previous 24 hours. As long as the total number of emails that you have sent is less than your quota, your send request will be accepted and your email will be sent.

SentLast24Hours is the number of emails that you have sent in the previous 24 hours.

MaxSendRate is the maximum number of emails that you can send per second.

Note that sending limits are based on recipients rather than on messages. For example, an email that has 10 recipients counts as 10 against your sending quota.

For more information, see *Managing Your Amazon SES Sending Limits* in the *Amazon Simple Email Service Developer Guide*.

- For API details, see [GetSendQuota](#) in *AWS CLI Command Reference*.

get-send-statistics

The following code example shows how to use `get-send-statistics`.

AWS CLI**To get your Amazon SES sending statistics**

The following example uses the `get-send-statistics` command to return your Amazon SES sending statistics

```
aws ses get-send-statistics
```

Output:

```
{
  "SendDataPoints": [
    {
      "Complaints": 0,
      "Timestamp": "2013-06-12T19:32:00Z",
      "DeliveryAttempts": 2,
      "Bounces": 0,
      "Rejects": 0
    },
    {
      "Complaints": 0,
      "Timestamp": "2013-06-12T00:47:00Z",
      "DeliveryAttempts": 1,
      "Bounces": 0,
      "Rejects": 0
    }
  ]
}
```

The result is a list of data points, representing the last two weeks of sending activity. Each data point in the list contains statistics for a 15-minute interval.

In this example, there are only two data points because the only emails that the user sent in the last two weeks fell within two 15-minute intervals.

For more information, see *Monitoring Your Amazon SES Usage Statistics* in the *Amazon Simple Email Service Developer Guide*.

- For API details, see [GetSendStatistics](#) in *AWS CLI Command Reference*.

list-identities

The following code example shows how to use `list-identities`.

AWS CLI**To list all identities (email addresses and domains) for a specific AWS account**

The following example uses the `list-identities` command to list all identities that have been submitted for verification with Amazon SES:

```
aws ses list-identities
```

Output:

```
{
  "Identities": [
    "user@example.com",
    "example.com"
  ]
}
```

The list that is returned contains all identities regardless of verification status (verified, pending verification, failure, etc.).

In this example, email addresses *and* domains are returned because we did not specify the `identity-type` parameter.

For more information about verification, see [Verifying Email Addresses and Domains in Amazon SES](#) in the *Amazon Simple Email Service Developer Guide*.

- For API details, see [ListIdentities](#) in *AWS CLI Command Reference*.

send-email

The following code example shows how to use `send-email`.

AWS CLI**To send a formatted email using Amazon SES**

The following example uses the `send-email` command to send a formatted email:

```
aws ses send-email --from sender@example.com --destination file://destination.json
--message file://message.json
```

Output:

```
{
  "MessageId": "EXAMPLEf3a5efcd1-51adec81-d2a4-4e3f-9fe2-5d85c1b23783-000000"
}
```

The destination and the message are JSON data structures saved in .json files in the current directory. These files are as follows:

destination.json:

```
{
  "ToAddresses": ["recipient1@example.com", "recipient2@example.com"],
  "CcAddresses": ["recipient3@example.com"],
  "BccAddresses": []
}
```

message.json:

```
{
  "Subject": {
    "Data": "Test email sent using the AWS CLI",
    "Charset": "UTF-8"
  },
  "Body": {
    "Text": {
      "Data": "This is the message body in text format.",
      "Charset": "UTF-8"
    },
    "Html": {
      "Data": "This message body contains HTML formatting. It can, for example,
contain links like this one: <a class=\"uLink\" href=\"http://docs.aws.amazon.com/
ses/latest/DeveloperGuide\" target=\"_blank\">Amazon SES Developer Guide</a>.",
      "Charset": "UTF-8"
    }
  }
}
```

Replace the sender and recipient email addresses with the ones you want to use. Note that the sender's email address must be verified with Amazon SES. Until you are granted production access to Amazon SES, you must also verify the email address of each recipient unless the recipient is the Amazon SES mailbox simulator. For more information on verification, see *Verifying Email Addresses and Domains in Amazon SES* in the *Amazon Simple Email Service Developer Guide*.

The Message ID in the output indicates that the call to send-email was successful.

If you don't receive the email, check your Junk box.

For more information on sending formatted email, see [Sending Formatted Email Using the Amazon SES API](#) in the *Amazon Simple Email Service Developer Guide*.

- For API details, see [SendEmail](#) in *AWS CLI Command Reference*.

send-raw-email

The following code example shows how to use `send-raw-email`.

AWS CLI

To send a raw email using Amazon SES

The following example uses the `send-raw-email` command to send an email with a TXT attachment:

```
aws ses send-raw-email --raw-message file://message.json
```

Output:

```
{
  "MessageId": "EXAMPLEf3f73d99b-c63fb06f-d263-41f8-a0fb-d0dc67d56c07-000000"
}
```

The raw message is a JSON data structure saved in a file named `message.json` in the current directory. It contains the following:

```
{
  "Data": "From: sender@example.com\nTo: recipient@example.com\nSubject: Test email
sent using the AWS CLI (contains an attachment)\nMIME-Version: 1.0\nContent-type:
Multipart/Mixed; boundary=\"NextPart\"\n\n--NextPart\nContent-Type: text/plain
\n\nThis is the message body.\n\n--NextPart\nContent-Type: text/plain;\nContent-
Disposition: attachment; filename=\"attachment.txt\"\n\nThis is the text in the
attachment.\n\n--NextPart--"
}
```

As you can see, "Data" is one long string that contains the entire raw email content in MIME format, including an attachment called `attachment.txt`.

Replace `sender@example.com` and `recipient@example.com` with the addresses you want to use. Note that the sender's email address must be verified with Amazon SES. Until you are granted

production access to Amazon SES, you must also verify the email address of the recipient unless the recipient is the Amazon SES mailbox simulator. For more information on verification, see *Verifying Email Addresses and Domains in Amazon SES* in the *Amazon Simple Email Service Developer Guide*.

The Message ID in the output indicates that the call to `send-raw-email` was successful.

If you don't receive the email, check your Junk box.

For more information on sending raw email, see *Sending Raw Email Using the Amazon SES API* in the *Amazon Simple Email Service Developer Guide*.

- For API details, see [SendRawEmail](#) in *AWS CLI Command Reference*.

set-identity-dkim-enabled

The following code example shows how to use `set-identity-dkim-enabled`.

AWS CLI

To enable or disable Easy DKIM for an Amazon SES verified identity

The following example uses the `set-identity-dkim-enabled` command to disable DKIM for a verified email address:

```
aws ses set-identity-dkim-enabled --identity user@example.com --no-dkim-enabled
```

For more information about Easy DKIM, see *Easy DKIM in Amazon SES* in the *Amazon Simple Email Service Developer Guide*.

- For API details, see [SetIdentityDkimEnabled](#) in *AWS CLI Command Reference*.

set-identity-feedback-forwarding-enabled

The following code example shows how to use `set-identity-feedback-forwarding-enabled`.

AWS CLI

To enable or disable bounce and complaint email feedback forwarding for an Amazon SES verified identity

The following example uses the `set-identity-feedback-forwarding-enabled` command to enable a verified email address to receive bounce and complaint notifications by email:

```
aws ses set-identity-feedback-forwarding-enabled --identity user@example.com --forwarding-enabled
```

You are required to receive bounce and complaint notifications via either Amazon SNS or email feedback forwarding, so you can only disable email feedback forwarding if you select an Amazon SNS topic for both bounce and complaint notifications.

For more information about notifications, see *Using Notifications With Amazon SES* in the *Amazon Simple Email Service Developer Guide*.

- For API details, see [SetIdentityFeedbackForwardingEnabled](#) in *AWS CLI Command Reference*.

set-identity-notification-topic

The following code example shows how to use `set-identity-notification-topic`.

AWS CLI

To set the Amazon SNS topic to which Amazon SES will publish bounce, complaint, and/or delivery notifications for a verified identity

The following example uses the `set-identity-notification-topic` command to specify the Amazon SNS topic to which a verified email address will receive bounce notifications:

```
aws ses set-identity-notification-topic --identity user@example.com --notification-type Bounce --sns-topic arn:aws:sns:us-east-1:EXAMPLE65304:MyTopic
```

For more information about notifications, see *Using Notifications With Amazon SES* in the *Amazon Simple Email Service Developer Guide*.

- For API details, see [SetIdentityNotificationTopic](#) in *AWS CLI Command Reference*.

verify-domain-dkim

The following code example shows how to use `verify-domain-dkim`.

AWS CLI

To generate a verified domain's DKIM tokens for DKIM signing with Amazon SES

The following example uses the `verify-domain-dkim` command to generate DKIM tokens for a domain that has been verified with Amazon SES:

```
aws ses verify-domain-dkim --domain example.com
```

Output:

```
{
  "DkimTokens": [
    "EXAMPLEq76owjnks3lnluwg65scbemvw",
    "EXAMPLEi3dnsj67hstzaj673klariwx2",
    "EXAMPLEwfbtcukvimehexktdtaz6naj"
  ]
}
```

To set up DKIM, you must use the returned DKIM tokens to update your domain's DNS settings with CNAME records that point to DKIM public keys hosted by Amazon SES. For more information, see [Easy DKIM in Amazon SES](#) in the *Amazon Simple Email Service Developer Guide*.

- For API details, see [VerifyDomainDkim](#) in *AWS CLI Command Reference*.

verify-domain-identity

The following code example shows how to use `verify-domain-identity`.

AWS CLI

To verify a domain with Amazon SES

The following example uses the `verify-domain-identity` command to verify a domain:

```
aws ses verify-domain-identity --domain example.com
```

Output:

```
{
  "VerificationToken": "eoEmxw+YaYhb3h3iVJHuXMJXqeu1q1/wmvjuEXAMPLE"
}
```

To complete domain verification, you must add a TXT record with the returned verification token to your domain's DNS settings. For more information, see *Verifying Domains in Amazon SES in the Amazon Simple Email Service Developer Guide*.

- For API details, see [VerifyDomainIdentity](#) in *AWS CLI Command Reference*.

verify-email-identity

The following code example shows how to use `verify-email-identity`.

AWS CLI

To verify an email address with Amazon SES

The following example uses the `verify-email-identity` command to verify an email address:

```
aws ses verify-email-identity --email-address user@example.com
```

Before you can send an email using Amazon SES, you must verify the address or domain that you are sending the email from to prove that you own it. If you do not have production access yet, you also need to verify any email addresses that you send emails to except for email addresses provided by the Amazon SES mailbox simulator.

After `verify-email-identity` is called, the email address will receive a verification email. The user must click on the link in the email to complete the verification process.

For more information, see *Verifying Email Addresses in Amazon SES in the Amazon Simple Email Service Developer Guide*.

- For API details, see [VerifyEmailIdentity](#) in *AWS CLI Command Reference*.

Shield examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Shield.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-drt-log-bucket

The following code example shows how to use `associate-drt-log-bucket`.

AWS CLI

To authorize the DRT to access an Amazon S3 bucket

The following `associate-drt-log-bucket` example creates an association between the DRT and the specified S3 bucket. This permits the DRT to access the bucket on behalf of the account.:

```
aws shield associate-drt-log-bucket \  
  --log-bucket flow-logs-for-website-lb
```

This command produces no output.

For more information, see [Authorize the DDoS Response Team](#) in the *AWS Shield Advanced Developer Guide*.

- For API details, see [AssociateDrtLogBucket](#) in *AWS CLI Command Reference*.

associate-drt-role

The following code example shows how to use `associate-drt-role`.

AWS CLI

To authorize the DRT to mitigate potential attacks on your behalf

The following `associate-drt-role` example creates an association between the DRT and the specified role. The DRT can use the role to access and manage the account.

```
aws shield associate-drt-role \  
  --role-arn arn:aws:iam::123456789012:role/service-role/DrtRole
```

This command produces no output.

For more information, see [Authorize the DDoS Response Team](#) in the *AWS Shield Advanced Developer Guide*.

- For API details, see [AssociateDrtRole](#) in *AWS CLI Command Reference*.

create-protection

The following code example shows how to use `create-protection`.

AWS CLI

To enable AWS Shield Advanced protection for a single AWS resource

The following `create-protection` example enables Shield Advanced protection for the specified AWS CloudFront distribution.

```
aws shield create-protection \  
  --name "Protection for CloudFront distribution" \  
  --resource-arn arn:aws:cloudfront::123456789012:distribution/E198WC25FX0WY8
```

Output:

```
{  
  "ProtectionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"  
}
```

For more information, see [Specify Your Resources to Protect](#) in the *AWS Shield Advanced Developer Guide*.

- For API details, see [CreateProtection](#) in *AWS CLI Command Reference*.

create-subscription

The following code example shows how to use `create-subscription`.

AWS CLI

To enable AWS Shield Advanced protection for an account

The following `create-subscription` example enables Shield Advanced protection for the account.

```
aws shield create-subscription
```

This command produces no output.

For more information, see [Getting Started with AWS Shield Advanced](#) in the *AWS Shield Advanced Developer Guide*.

- For API details, see [CreateSubscription](#) in *AWS CLI Command Reference*.

delete-protection

The following code example shows how to use `delete-protection`.

AWS CLI

To remove AWS Shield Advanced protection from an AWS resource

The following `delete-protection` example removes the specified AWS Shield Advanced protection.

```
aws shield delete-protection \  
  --protection-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

This command produces no output.

For more information, see [Removing AWS Shield Advanced from an AWS Resource](#) in the *AWS Shield Advanced Developer Guide*.

- For API details, see [DeleteProtection](#) in *AWS CLI Command Reference*.

describe-attack

The following code example shows how to use `describe-attack`.

AWS CLI

To retrieve a detailed description of an attack

The following describe-attack example displays details about the DDoS attack with the specified attack ID. You can obtain attack IDs by running the list-attacks command.

```
aws shield describe-attack --attack-id a1b2c3d4-5678-90ab-cdef-EXAMPLE22222
```

Output:

```
{
  "Attack": {
    "AttackId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "ResourceArn": "arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/testElb",
    "SubResources": [
      {
        "Type": "IP",
        "Id": "192.0.2.2",
        "AttackVectors": [
          {
            "VectorType": "SYN_FLOOD",
            "VectorCounters": [
              {
                "Name": "SYN_FLOOD_BPS",
                "Max": 982184.0,
                "Average": 982184.0,
                "Sum": 11786208.0,
                "N": 12,
                "Unit": "BPS"
              }
            ]
          }
        ]
      }
    ],
    "Counters": []
  },
  {
    "Type": "IP",
    "Id": "192.0.2.3",
    "AttackVectors": [
      {
        "VectorType": "SYN_FLOOD",
```

```
        "VectorCounters": [
            {
                "Name": "SYN_FLOOD_BPS",
                "Max": 982184.0,
                "Average": 982184.0,
                "Sum": 9821840.0,
                "N": 10,
                "Unit": "BPS"
            }
        ]
    },
    "Counters": []
},
{
    "Type": "IP",
    "Id": "192.0.2.4",
    "AttackVectors": [
        {
            "VectorType": "SYN_FLOOD",
            "VectorCounters": [
                {
                    "Name": "SYN_FLOOD_BPS",
                    "Max": 982184.0,
                    "Average": 982184.0,
                    "Sum": 7857472.0,
                    "N": 8,
                    "Unit": "BPS"
                }
            ]
        }
    ],
    "Counters": []
},
{
    "Type": "IP",
    "Id": "192.0.2.5",
    "AttackVectors": [
        {
            "VectorType": "SYN_FLOOD",
            "VectorCounters": [
                {
                    "Name": "SYN_FLOOD_BPS",
                    "Max": 982184.0,
```

```
        "Average": 982184.0,
        "Sum": 1964368.0,
        "N": 2,
        "Unit": "BPS"
      }
    ]
  },
  "Counters": []
},
{
  "Type": "IP",
  "Id": "2001:DB8::bcde:4321:8765:0:0",
  "AttackVectors": [
    {
      "VectorType": "SYN_FLOOD",
      "VectorCounters": [
        {
          "Name": "SYN_FLOOD_BPS",
          "Max": 982184.0,
          "Average": 982184.0,
          "Sum": 1964368.0,
          "N": 2,
          "Unit": "BPS"
        }
      ]
    }
  ],
  "Counters": []
},
{
  "Type": "IP",
  "Id": "192.0.2.6",
  "AttackVectors": [
    {
      "VectorType": "SYN_FLOOD",
      "VectorCounters": [
        {
          "Name": "SYN_FLOOD_BPS",
          "Max": 982184.0,
          "Average": 982184.0,
          "Sum": 1964368.0,
          "N": 2,
          "Unit": "BPS"
        }
      ]
    }
  ],
  "Counters": []
}
```

```
        }
      ]
    }
  ],
  "Counters": []
}
],
"StartTime": 1576024927.457,
"EndTime": 1576025647.457,
"AttackCounters": [],
"AttackProperties": [
  {
    "AttackLayer": "NETWORK",
    "AttackPropertyIdentifier": "SOURCE_IP_ADDRESS",
    "TopContributors": [
      {
        "Name": "198.51.100.5",
        "Value": 2024475682
      },
      {
        "Name": "198.51.100.8",
        "Value": 1311380863
      },
      {
        "Name": "203.0.113.4",
        "Value": 900599855
      },
      {
        "Name": "198.51.100.4",
        "Value": 769417366
      },
      {
        "Name": "203.1.113.13",
        "Value": 757992847
      }
    ]
  },
  {
    "AttackLayer": "NETWORK",
    "AttackPropertyIdentifier": "SOURCE_COUNTRY",
    "TopContributors": [
```

```
        "Name": "United States",
        "Value": 80938161764
    },
    {
        "Name": "Brazil",
        "Value": 9929864330
    },
    {
        "Name": "Netherlands",
        "Value": 1635009446
    },
    {
        "Name": "Mexico",
        "Value": 144832971
    },
    {
        "Name": "Japan",
        "Value": 45369000
    }
],
"Unit": "BYTES",
"Total": 92773354841
},
{
    "AttackLayer": "NETWORK",
    "AttackPropertyIdentifier": "SOURCE_ASN",
    "TopContributors": [
        {
            "Name": "12345",
            "Value": 74953625841
        },
        {
            "Name": "12346",
            "Value": 4440087595
        },
        {
            "Name": "12347",
            "Value": 1635009446
        },
        {
            "Name": "12348",
            "Value": 1221230000
        },
    ]
}
```

```

        "Name": "12349",
        "Value": 1199425294
      }
    ],
    "Unit": "BYTES",
    "Total": 92755479921
  }
],
"Mitigations": []
}
}

```

For more information, see [Reviewing DDoS Incidents](#) in the *AWS Shield Advanced Developer Guide*.

- For API details, see [DescribeAttack](#) in *AWS CLI Command Reference*.

describe-drt-access

The following code example shows how to use `describe-drt-access`.

AWS CLI

To retrieve a description of the authorizations the DRT has to mitigate attacks on your behalf

The following `describe-drt-access` example retrieves the role and S3 bucket authorizations that the DRT has, which allow it to respond to potential attacks on your behalf.

```
aws shield describe-drt-access
```

Output:

```

{
  "RoleArn": "arn:aws:iam::123456789012:role/service-role/DrtRole",
  "LogBucketList": [
    "flow-logs-for-website-lb"
  ]
}

```

For more information, see [Authorize the DDoS Response Team](#) in the *AWS Shield Advanced Developer Guide*.

- For API details, see [DescribeDrtAccess](#) in *AWS CLI Command Reference*.

describe-emergency-contact-settings

The following code example shows how to use `describe-emergency-contact-settings`.

AWS CLI

To retrieve emergency e-mail addresses that you have on file with the DRT

The following `describe-emergency-contact-settings` example retrieves the e-mail addresses that are on file with the DRT for the account. These are the addresses the DRT should contact when it's responding to a suspected attack.

```
aws shield describe-emergency-contact-settings
```

Output:

```
{
  "EmergencyContactList": [
    {
      "EmailAddress": "ops@example.com"
    },
    {
      "EmailAddress": "ddos-notifications@example.com"
    }
  ]
}
```

For more information, see [How AWS Shield Works](https://docs.aws.amazon.com/waf/latest/developerguide/ddos-overview.html) in the *AWS Shield Advanced Developer Guide*.

- For API details, see [DescribeEmergencyContactSettings](#) in *AWS CLI Command Reference*.

describe-protection

The following code example shows how to use `describe-protection`.

AWS CLI

To retrieve the details for an AWS Shield Advanced protection

The following `describe-protection` example displays details about the Shield Advanced protection with the specified ID. You can obtain protection IDs by running the `list-protections` command.

```
aws shield describe-protection \  
  --protection-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{  
  "Protection": {  
    "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
    "Name": "1.2.3.4",  
    "ResourceArn": "arn:aws:ec2:us-west-2:123456789012:eip-allocation/  
eipalloc-0ac1537af40742a6d"  
  }  
}
```

For more information, see [Specify Your Resources to Protect](#) in the *AWS Shield Advanced Developer Guide*.

- For API details, see [DescribeProtection](#) in *AWS CLI Command Reference*.

describe-subscription

The following code example shows how to use `describe-subscription`.

AWS CLI

To retrieve the details of the AWS Shield Advanced protection for the account

The following `describe-subscription` example displays details about the Shield Advanced protection provided for the account.:

```
aws shield describe-subscription
```

Output:

```
{  
  "Subscription": {  
    "StartTime": 1534368978.0,
```



```

    "EndTime": 1597613778.0,
    "TimeCommitmentInSeconds": 63244800,
    "AutoRenew": "ENABLED",
    "Limits": [
      {
        "Type": "GLOBAL_ACCELERATOR",
        "Max": 1000
      },
      {
        "Type": "ROUTE53_HOSTED_ZONE",
        "Max": 1000
      },
      {
        "Type": "CF_DISTRIBUTION",
        "Max": 1000
      },
      {
        "Type": "ELB_LOAD_BALANCER",
        "Max": 1000
      },
      {
        "Type": "EC2_ELASTIC_IP_ALLOCATION",
        "Max": 1000
      }
    ]
  }
}

```

For more information, see [How AWS Shield Works](#) in the *AWS Shield Advanced Developer Guide*.

- For API details, see [DescribeSubscription](#) in *AWS CLI Command Reference*.

disassociate-drt-log-bucket

The following code example shows how to use `disassociate-drt-log-bucket`.

AWS CLI

To remove the authorization for DRT to access an Amazon S3 bucket on your behalf

The following `disassociate-drt-log-bucket` example removes the association between the DRT and the specified S3 bucket. After this command completes, the DRT can no longer access the bucket on behalf of the account.

```
aws shield disassociate-drt-log-bucket \  
  --log-bucket flow-logs-for-website-lb
```

This command produces no output.

For more information, see [Authorize the DDoS Response Team](#) in the *AWS Shield Advanced Developer Guide*.

- For API details, see [DisassociateDrtLogBucket](#) in *AWS CLI Command Reference*.

disassociate-drt-role

The following code example shows how to use `disassociate-drt-role`.

AWS CLI

To remove the authorization for DRT to mitigate potential attacks on your behalf

The following `disassociate-drt-role` example removes the association between the DRT and the account. After this call, the DRT can no longer access or manage your account.

```
aws shield disassociate-drt-role
```

This command produces no output.

For more information, see [Authorize the DDoS Response Team](#) in the *AWS Shield Advanced Developer Guide*.

- For API details, see [DisassociateDrtRole](#) in *AWS CLI Command Reference*.

get-subscription-state

The following code example shows how to use `get-subscription-state`.

AWS CLI

To retrieve the current state of the account's AWS Shield Advanced subscription

The following `get-subscription-state` example retrieves the state of the Shield Advanced protection for the account.

```
aws shield get-subscription-state
```

Output:

```
{
  "SubscriptionState": "ACTIVE"
}
```

For more information, see [How AWS Shield Works](#) in the *AWS Shield Advanced Developer Guide*.

- For API details, see [GetSubscriptionState](#) in *AWS CLI Command Reference*.

list-attacks

The following code example shows how to use `list-attacks`.

AWS CLI**To retrieve attack summaries from AWS Shield Advanced**

The following `list-attacks` example retrieves summaries of attacks for the specified AWS CloudFront distribution during the specified time period. The response includes attack IDs that you can provide to the `describe-attack` command for detailed information on an attack.

```
aws shield list-attacks \
  --resource-arns arn:aws:cloudfront::12345678910:distribution/E1PXMP22ZVFAOR \
  --start-time FromInclusive=1529280000,ToExclusive=1529300000
```

Output:

```
{
  "AttackSummaries": [
    {
      "AttackId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "ResourceArn": "arn:aws:cloudfront::123456789012:distribution/
E1PXMP22ZVFAOR",
      "StartTime": 1529280000.0,
      "EndTime": 1529449200.0,
      "AttackVectors": [
        {
          "VectorType": "SYN_FLOOD"
        }
      ]
    }
  ]
}
```

```
]
}
```

For more information, see [Reviewing DDoS Incidents](#) in the *AWS Shield Advanced Developer Guide*.

- For API details, see [ListAttacks](#) in *AWS CLI Command Reference*.

list-protections

The following code example shows how to use `list-protections`.

AWS CLI

To retrieve protection summaries from AWS Shield Advanced

The following `list-protections` example retrieves summaries of the protections that are enabled for the account.

```
aws shield list-protections
```

Output:

```
{
  "Protections": [
    {
      "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "Name": "Protection for CloudFront distribution",
      "ResourceArn": "arn:aws:cloudfront::123456789012:distribution/
E198WC25FX0WY8"
    }
  ]
}
```

For more information, see [Specify Your Resources to Protect](#) in the *AWS Shield Advanced Developer Guide*.

- For API details, see [ListProtections](#) in *AWS CLI Command Reference*.

update-emergency-contact-settings

The following code example shows how to use `update-emergency-contact-settings`.

AWS CLI

To define the emergency e-mail addresses that are on file with the DRT

The following `update-emergency-contact-settings` example defines two e-mail addresses that the DRT should contact when it's responding to a suspected attack.

```
aws shield update-emergency-contact-settings \  
    --emergency-contact-list EmailAddress=ops@example.com EmailAddress=ddos-  
notifications@example.com
```

This command produces no output.

For more information, see [How AWS Shield Works](#) in the *AWS Shield Advanced Developer Guide*.

- For API details, see [UpdateEmergencyContactSettings](#) in *AWS CLI Command Reference*.

update-subscription

The following code example shows how to use `update-subscription`.

AWS CLI

To modify the account's AWS Shield Advanced subscription

The following `update-subscription` example enables auto-renewal of the AWS Shield Advanced subscription for the account.

```
aws shield update-subscription \  
    --auto-renew ENABLED
```

This command produces no output.

For more information, see [How AWS Shield Works](#) in the *AWS Shield Advanced Developer Guide*.

- For API details, see [UpdateSubscription](#) in *AWS CLI Command Reference*.

Signer examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Signer.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

cancel-signing-profile

The following code example shows how to use `cancel-signing-profile`.

AWS CLI

To delete a signing profile

The following `cancel-signing-profile` example removes an existing signing profile from AWS Signer.

```
aws signer cancel-signing-profile \  
  --profile-name MyProfile1
```

This command produces no output.

- For API details, see [CancelSigningProfile](#) in *AWS CLI Command Reference*.

describe-signing-job

The following code example shows how to use `describe-signing-job`.

AWS CLI

To display details about a signing job

The following `describe-signing-job` example displays details about the specified signing job.

```
aws signer describe-signing-job \  
  --job-id 2065c468-73e2-4385-a6c9-0123456789abc
```

Output:

```
{  
  "status": "Succeeded",  
  "completedAt": 1568412037,  
  "platformId": "AmazonFreeRTOS-Default",  
  "signingMaterial": {  
    "certificateArn": "arn:aws:acm:us-  
west-2:123456789012:certificate/6a55389b-306b-4e8c-a95c-0123456789abc"  
  },  
  "statusReason": "Signing Succeeded",  
  "jobId": "2065c468-73e2-4385-a6c9-0123456789abc",  
  "source": {  
    "s3": {  
      "version": "PNyFaUTgsQh5ZdMccoCe6pT1g0pgB_M4",  
      "bucketName": "signer-source",  
      "key": "MyCode.rb"  
    }  
  },  
  "profileName": "MyProfile2",  
  "signedObject": {  
    "s3": {  
      "bucketName": "signer-destination",  
      "key": "signed-2065c468-73e2-4385-a6c9-0123456789abc"  
    }  
  },  
  "requestedBy": "arn:aws:iam::123456789012:user/maria",  
  "createdAt": 1568412036  
}
```

- For API details, see [DescribeSigningJob](#) in *AWS CLI Command Reference*.

get-signing-platform

The following code example shows how to use `get-signing-platform`.

AWS CLI

To display details about a signing platform

The following `get-signing-platform` example displays details about the specified signing platform.

```
aws signer get-signing-platform \  
  --platform-id AmazonFreeRTOS-TI-CC3220SF
```

Output:

```
{  
  "category": "AWS",  
  "displayName": "Amazon FreeRTOS SHA1-RSA CC3220SF-Format",  
  "target": "SHA1-RSA-TISHA1",  
  "platformId": "AmazonFreeRTOS-TI-CC3220SF",  
  "signingConfiguration": {  
    "encryptionAlgorithmOptions": {  
      "defaultValue": "RSA",  
      "allowedValues": [  
        "RSA"  
      ]  
    },  
    "hashAlgorithmOptions": {  
      "defaultValue": "SHA1",  
      "allowedValues": [  
        "SHA1"  
      ]  
    }  
  },  
  "maxSizeInMB": 16,  
  "partner": "AmazonFreeRTOS",  
  "signingImageFormat": {  
    "defaultFormat": "JSONEmbedded",  
    "supportedFormats": [  
      "JSONEmbedded"  
    ]  
  }  
}
```

- For API details, see [GetSigningPlatform](#) in *AWS CLI Command Reference*.

get-signing-profile

The following code example shows how to use `get-signing-profile`.

AWS CLI

To display details about a signing profile

The following `get-signing-profile` example displays details about the specified signing profile.

```
aws signer get-signing-profile \  
  --profile-name MyProfile3
```

Output:

```
{  
  "platformId": "AmazonFreeRTOS-TI-CC3220SF",  
  "profileName": "MyProfile3",  
  "status": "Active",  
  "signingMaterial": {  
    "certificateArn": "arn:aws:acm:us-  
west-2:123456789012:certificate/6a55389b-306b-4e8c-a95c-0123456789abc"  
  }  
}
```

- For API details, see [GetSigningProfile](#) in *AWS CLI Command Reference*.

list-signing-jobs

The following code example shows how to use `list-signing-jobs`.

AWS CLI

To list all signing jobs

The following `list-signing-jobs` example displays details about all signing jobs for the account.

```
aws signer list-signing-jobs
```

In this example, two jobs are returned, one successful, and one failed.

```
{
  "jobs": [
    {
      "status": "Succeeded",
      "signingMaterial": {
        "certificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/6a55389b-306b-4e8c-a95c-0123456789abc"
      },
      "jobId": "2065c468-73e2-4385-a6c9-0123456789abc",
      "source": {
        "s3": {
          "version": "PNyFaUTgsQh5ZdMCcoCe6pT1g0pgB_M4",
          "bucketName": "signer-source",
          "key": "MyCode.rb"
        }
      },
      "signedObject": {
        "s3": {
          "bucketName": "signer-destination",
          "key": "signed-2065c468-73e2-4385-a6c9-0123456789abc"
        }
      },
      "createdAt": 1568412036
    },
    {
      "status": "Failed",
      "source": {
        "s3": {
          "version": "PNyFaUTgsQh5ZdMCcoCe6pT1g0pgB_M4",
          "bucketName": "signer-source",
          "key": "MyOtherCode.rb"
        }
      },
      "signingMaterial": {
        "certificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/6a55389b-306b-4e8c-a95c-0123456789abc"
      },
      "createdAt": 1568402690,
      "jobId": "74d9825e-22fc-4a0d-b962-0123456789abc"
    }
  ]
}
```

```
}
```

- For API details, see [ListSigningJobs](#) in *AWS CLI Command Reference*.

list-signing-platforms

The following code example shows how to use `list-signing-platforms`.

AWS CLI

To list all signing platforms

The following `list-signing-platforms` example displays details about all available signing platforms.

```
aws signer list-signing-platforms
```

Output:

```
{
  "platforms": [
    {
      "category": "AWS",
      "displayName": "AWS IoT Device Management SHA256-ECDSA ",
      "target": "SHA256-ECDSA",
      "platformId": "AWSIoTDeviceManagement-SHA256-ECDSA",
      "signingConfiguration": {
        "encryptionAlgorithmOptions": {
          "defaultValue": "ECDSA",
          "allowedValues": [
            "ECDSA"
          ]
        },
        "hashAlgorithmOptions": {
          "defaultValue": "SHA256",
          "allowedValues": [
            "SHA256"
          ]
        }
      },
      "maxSizeInMB": 2048,
      "partner": "AWSIoTDeviceManagement",
    }
  ]
}
```

```
    "signingImageFormat": {
      "defaultFormat": "JSONDetached",
      "supportedFormats": [
        "JSONDetached"
      ]
    }
  },
  {
    "category": "AWS",
    "displayName": "Amazon FreeRTOS SHA1-RSA CC3220SF-Format",
    "target": "SHA1-RSA-TISHA1",
    "platformId": "AmazonFreeRTOS-TI-CC3220SF",
    "signingConfiguration": {
      "encryptionAlgorithmOptions": {
        "defaultValue": "RSA",
        "allowedValues": [
          "RSA"
        ]
      },
      "hashAlgorithmOptions": {
        "defaultValue": "SHA1",
        "allowedValues": [
          "SHA1"
        ]
      }
    },
    "maxSizeInMB": 16,
    "partner": "AmazonFreeRTOS",
    "signingImageFormat": {
      "defaultFormat": "JSONEmbedded",
      "supportedFormats": [
        "JSONEmbedded"
      ]
    }
  },
  {
    "category": "AWS",
    "displayName": "Amazon FreeRTOS SHA256-ECDSA",
    "target": "SHA256-ECDSA",
    "platformId": "AmazonFreeRTOS-Default",
    "signingConfiguration": {
      "encryptionAlgorithmOptions": {
        "defaultValue": "ECDSA",
        "allowedValues": [
```

```

        "ECDSA"
      ],
    },
    "hashAlgorithmOptions": {
      "defaultValue": "SHA256",
      "allowedValues": [
        "SHA256"
      ]
    },
  },
  "maxSizeInMB": 16,
  "partner": "AmazonFreeRTOS",
  "signingImageFormat": {
    "defaultFormat": "JSONEmbedded",
    "supportedFormats": [
      "JSONEmbedded"
    ]
  }
}
]
}

```

- For API details, see [ListSigningPlatforms](#) in *AWS CLI Command Reference*.

list-signing-profiles

The following code example shows how to use `list-signing-profiles`.

AWS CLI

To list all signing profiles

The following `list-signing-profiles` example displays details about all signing profiles for the account.

```
aws signer list-signing-profiles
```

Output:

```
{
  "profiles": [
    {
```

```

    "platformId": "AmazonFreeRTOS-TI-CC3220SF",
    "profileName": "MyProfile4",
    "status": "Active",
    "signingMaterial": {
      "certificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/6a55389b-306b-4e8c-a95c-0123456789abc"
    }
  },
  {
    "platformId": "AWSIoTDeviceManagement-SHA256-ECDSA",
    "profileName": "MyProfile5",
    "status": "Active",
    "signingMaterial": {
      "certificateArn": "arn:aws:acm:us-
west-2:123456789012:certificate/6a55389b-306b-4e8c-a95c-0123456789abc"
    }
  }
]
}

```

- For API details, see [ListSigningProfiles](#) in *AWS CLI Command Reference*.

put-signing-profile

The following code example shows how to use `put-signing-profile`.

AWS CLI

To create a signing profile

The following `put-signing-profile` example creates a signing profile using the specified certificate and platform.

```

aws signer put-signing-profile \
  --profile-name MyProfile6 \
  --signing-material certificateArn=arn:aws:acm:us-
west-2:123456789012:certificate/6a55389b-306b-4e8c-a95c-0123456789abc \
  --platform AmazonFreeRTOS-TI-CC3220SF

```

Output:

```
{
```

```
"arn": "arn:aws:signer:us-west-2:123456789012:/signing-profiles/MyProfile6"
}
```

- For API details, see [PutSigningProfile](#) in *AWS CLI Command Reference*.

start-signing-job

The following code example shows how to use `start-signing-job`.

AWS CLI

To start a signing job

The following `start-signing-job` example starts a signing job on the code found at the specified source. It uses the specified profile to do the signing and places the signed code in the specified destination.

```
aws signer start-signing-job \
  --source 's3={bucketName=signer-
source,key=MyCode.rb,version=PNyFaUTgsQh5ZdMCcoCe6pT1g0pgB_M4}' \
  --destination 's3={bucketName=signer-destination,prefix=signed-}' \
  --profile-name MyProfile7
```

The output is the ID of the signing job.

```
{
  "jobId": "2065c468-73e2-4385-a6c9-0123456789abc"
}
```

- For API details, see [StartSigningJob](#) in *AWS CLI Command Reference*.

Snowball examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Snowball.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

get-snowball-usage

The following code example shows how to use `get-snowball-usage`.

AWS CLI

To get information about the Snowball service limit for your account

The following `get-snowball-usage` example displays information about the Snowball service limit for your account, and also the number of Snowballs your account has in use.

```
aws snowball get-snowball-usage
```

Output:

```
{
  "SnowballLimit": 1,
  "SnowballsInUse": 0
}
```

FOR more information, see [AWS Snowball Edge Limits](#) in the *AWS Snowball Developer Guide*.

- For API details, see [GetSnowballUsage](#) in *AWS CLI Command Reference*.

list-jobs

The following code example shows how to use `list-jobs`.

AWS CLI

To list the current Snowball jobs in your account

The following `list-jobs` example displays an array of `JobListEntry` objects. In this example, a single job is listed.

```
aws snowball list-jobs
```

Output:

```
{
  "JobListEntries": [
    {
      "CreationDate": 2016-09-27T14:50Z,
      "Description": "Important Photos 2016-08-11",
      "IsMaster": TRUE,
      "JobId": "ABCd1e324fe-022f-488e-a98b-3b0566063db1",
      "JobState": "Complete",
      "JobType": "IMPORT",
      "SnowballType": "EDGE"
    }
  ]
}
```

For more information, see [Jobs for AWS Snowball Edge devices](#) in the *AWS Snowball Developer Guide*.

- For API details, see [ListJobs](#) in *AWS CLI Command Reference*.

Amazon SNS examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon SNS.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)
- [Scenarios](#)

Actions

add-permission

The following code example shows how to use add-permission.

AWS CLI

To add a permission to a topic

The following add-permission example adds the permission for AWS account 987654321098 to use the Publish action with the specified topic under AWS account 123456789012.

```
aws sns add-permission \  
  --topic-arn arn:aws:sns:us-west-2:123456789012:MyTopic \  
  --label Publish-Permission \  
  --aws-account-id 987654321098 \  
  --action-name Publish
```

This command produces no output.

- For API details, see [AddPermission](#) in *AWS CLI Command Reference*.

check-if-phone-number-is-opted-out

The following code example shows how to use check-if-phone-number-is-opted-out.

AWS CLI

To check SMS message opt-out for a phone number

The following check-if-phone-number-is-opted-out example checks whether the specified phone number is opted out of receiving SMS messages from the current AWS account.

```
aws sns check-if-phone-number-is-opted-out \  

```

```
--phone-number +1555550100
```

Output:

```
{
  "isOptedOut": false
}
```

- For API details, see [CheckIfPhoneNumbersOptedOut](#) in *AWS CLI Command Reference*.

confirm-subscription

The following code example shows how to use `confirm-subscription`.

AWS CLI**To confirm a subscription**

The following `confirm-subscription` command completes the confirmation process started when you subscribed to an SNS topic named `my-topic`. The `--token` parameter comes from the confirmation message sent to the notification endpoint specified in the `subscribe` call.

```
aws sns confirm-subscription \
  --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic \
  --token
2336412f37fb687f5d51e6e241d7700ae02f7124d8268910b858cb4db727ceeb2474bb937929d3bdd7ce5d0cce1
```

Output:

```
{
  "SubscriptionArn": "arn:aws:sns:us-west-2:123456789012:my-
topic:8a21d249-4329-4871-acc6-7be709c6ea7f"
}
```

- For API details, see [ConfirmSubscription](#) in *AWS CLI Command Reference*.

create-platform-application

The following code example shows how to use `create-platform-application`.

AWS CLI

To create a platform application

The following `create-platform-application` example creates a Google Firebase platform application using the specified platform credential.

```
aws sns create-platform-application \  
  --name MyApplication \  
  --platform GCM \  
  --attributes PlatformCredential=EXAMPLEabcd12345jklm67890stuv12345bcdef
```

Output:

```
{  
  "PlatformApplicationArn": "arn:aws:sns:us-west-2:123456789012:app/GCM/  
MyApplication"  
}
```

- For API details, see [CreatePlatformApplication](#) in *AWS CLI Command Reference*.

create-topic

The following code example shows how to use `create-topic`.

AWS CLI

To create an SNS topic

The following `create-topic` example creates an SNS topic named `my-topic`.

```
aws sns create-topic \  
  --name my-topic
```

Output:

```
{  
  "ResponseMetadata": {  
    "RequestId": "1469e8d7-1642-564e-b85d-a19b4b341f83"  
  },
```

```
"TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic"
}
```

For more information, see [Using the AWS Command Line Interface with Amazon SQS and Amazon SNS](#) in the *AWS Command Line Interface User Guide*.

- For API details, see [CreateTopic](#) in *AWS CLI Command Reference*.

delete-endpoint

The following code example shows how to use delete-endpoint.

AWS CLI

To delete a platform application endpoint

The following delete-endpoint example deletes the specified platform application endpoint.

```
aws sns delete-endpoint \  
  --endpoint-arn arn:aws:sns:us-west-2:123456789012:endpoint/GCM/  
  MyApplication/12345678-abcd-9012-efgh-345678901234
```

This command produces no output.

- For API details, see [DeleteEndpoint](#) in *AWS CLI Command Reference*.

delete-platform-application

The following code example shows how to use delete-platform-application.

AWS CLI

To delete a platform application

The following delete-platform-application example deletes the specified platform application.

```
aws sns delete-platform-application \  
  --platform-application-arn arn:aws:sns:us-west-2:123456789012:app/ADM/  
  MyApplication
```

This command produces no output.

- For API details, see [DeletePlatformApplication](#) in *AWS CLI Command Reference*.

delete-topic

The following code example shows how to use `delete-topic`.

AWS CLI

To delete an SNS topic

The following `delete-topic` example deletes the specified SNS topic.

```
aws sns delete-topic \  
  --topic-arn "arn:aws:sns:us-west-2:123456789012:my-topic"
```

This command produces no output.

- For API details, see [DeleteTopic](#) in *AWS CLI Command Reference*.

get-endpoint-attributes

The following code example shows how to use `get-endpoint-attributes`.

AWS CLI

To list platform application endpoint attributes

The following `get-endpoint-attributes` example lists the attributes for the specified platform application endpoint.

```
aws sns get-endpoint-attributes \  
  --endpoint-arn arn:aws:sns:us-west-2:123456789012:endpoint/GCM/  
MyApplication/12345678-abcd-9012-efgh-345678901234
```

Output:

```
{  
  "Attributes": {
```

```
    "Enabled": "true",  
    "Token": "EXAMPLE12345..."  
  }  
}
```

- For API details, see [GetEndpointAttributes](#) in *AWS CLI Command Reference*.

get-platform-application-attributes

The following code example shows how to use `get-platform-application-attributes`.

AWS CLI

To list the platform application attributes

The following `get-platform-application-attributes` example lists the attributes for the specified platform application.

```
aws sns get-platform-application-attributes \  
  --platform-application-arn arn:aws:sns:us-west-2:123456789012:app/MPNS/  
  MyApplication
```

Output:

```
{  
  "Attributes": {  
    "Enabled": "true",  
    "SuccessFeedbackSampleRate": "100"  
  }  
}
```

- For API details, see [GetPlatformApplicationAttributes](#) in *AWS CLI Command Reference*.

get-sms-attributes

The following code example shows how to use `get-sms-attributes`.

AWS CLI

To list the default SMS message attributes

The following `get-sms-attributes` example lists the default attributes for sending SMS messages.

```
aws sns get-sms-attributes
```

Output:

```
{
  "attributes": {
    "DefaultSenderId": "MyName"
  }
}
```

- For API details, see [GetSMSAttributes](#) in *AWS CLI Command Reference*.

get-subscription-attributes

The following code example shows how to use `get-subscription-attributes`.

AWS CLI

To retrieve subscription attributes for a topic

The following `get-subscription-attributes` displays the attributes of the specified subscription. You can get the `subscription-arn` from the output of the `list-subscriptions` command.

```
aws sns get-subscription-attributes \
  --subscription-arn "arn:aws:sns:us-west-2:123456789012:my-
  topic:8a21d249-4329-4871-acc6-7be709c6ea7f"
```

Output:

```
{
  "Attributes": {
    "Endpoint": "my-email@example.com",
    "Protocol": "email",
    "RawMessageDelivery": "false",
    "ConfirmationWasAuthenticated": "false",
    "Owner": "123456789012",
```



```

    "SubscriptionArn": "arn:aws:sns:us-west-2:123456789012:my-
topic:8a21d249-4329-4871-acc6-7be709c6ea7f",
    "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic"
  }
}

```

- For API details, see [GetSubscriptionAttributes](#) in *AWS CLI Command Reference*.

get-topic-attributes

The following code example shows how to use get-topic-attributes.

AWS CLI

To retrieve the attributes of a topic

The following get-topic-attributes example displays the attributes for the specified topic.

```

aws sns get-topic-attributes \
  --topic-arn "arn:aws:sns:us-west-2:123456789012:my-topic"

```

Output:

```

{
  "Attributes": {
    "SubscriptionsConfirmed": "1",
    "DisplayName": "my-topic",
    "SubscriptionsDeleted": "0",
    "EffectiveDeliveryPolicy": "{\"http\":{\"defaultHealthyRetryPolicy\":{\"minDelayTarget\":20,\"maxDelayTarget\":20,\"numRetries\":3,\"numMaxDelayRetries\":0,\"numNoDelayRetries\":0,\"numMinDelayRetries\":0,\"backoffFunction\":\"linear\"},\"disableSubscriptionOverrides\":false}}",
    "Owner": "123456789012",
    "Policy": "{\"Version\":\"2008-10-17\",\"Id\":\"__default_policy_ID\",\"Statement\": [{\"Sid\":\"__default_statement_ID\",\"Effect\":\"Allow\",\"Principal\":{\"AWS\":\"*\"},\"Action\": [\"SNS:Subscribe\",\"SNS:ListSubscriptionsByTopic\",\"SNS:DeleteTopic\",\"SNS:GetTopicAttributes\",\"SNS:Publish\",\"SNS:RemovePermission\",\"SNS:AddPermission\",\"SNS:SetTopicAttributes\"],\"Resource\":\"arn:aws:sns:us-west-2:123456789012:my-topic\",\"Condition\": {\"StringEquals\":{\"AWS:SourceOwner\":\"0123456789012\"}}}]}",
    "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic",
    "SubscriptionsPending": "0"
  }
}

```

```
}  
}
```

- For API details, see [GetTopicAttributes](#) in *AWS CLI Command Reference*.

list-endpoints-by-platform-application

The following code example shows how to use `list-endpoints-by-platform-application`.

AWS CLI

To list the endpoints for a platform application

The following `list-endpoints-by-platform-application` example lists the endpoints and endpoint attributes for the specified platform application.

```
aws sns list-endpoints-by-platform-application \  
  --platform-application-arn arn:aws:sns:us-west-2:123456789012:app/GCM/  
  MyApplication
```

Output:

```
{  
  "Endpoints": [  
    {  
      "Attributes": {  
        "Token": "EXAMPLE12345...",  
        "Enabled": "true"  
      },  
      "EndpointArn": "arn:aws:sns:us-west-2:123456789012:endpoint/GCM/  
MyApplication/12345678-abcd-9012-efgh-345678901234"  
    }  
  ]  
}
```

- For API details, see [ListEndpointsByPlatformApplication](#) in *AWS CLI Command Reference*.

list-phone-numbers-opted-out

The following code example shows how to use `list-phone-numbers-opted-out`.

AWS CLI

To list SMS message opt-outs

The following `list-phone-numbers-opted-out` example lists the phone numbers opted out of receiving SMS messages.

```
aws sns list-phone-numbers-opted-out
```

Output:

```
{
  "phoneNumbers": [
    "+15555550100"
  ]
}
```

- For API details, see [ListPhoneNumbersOptedOut](#) in *AWS CLI Command Reference*.

list-platform-applications

The following code example shows how to use `list-platform-applications`.

AWS CLI

To list platform applications

The following `list-platform-applications` example lists the platform applications for ADM and MPNS.

```
aws sns list-platform-applications
```

Output:

```
{
  "PlatformApplications": [
    {
      "PlatformApplicationArn": "arn:aws:sns:us-west-2:123456789012:app/ADM/MyApplication",
      "Attributes": {
        "SuccessFeedbackSampleRate": "100",
        "Enabled": "true"
      }
    }
  ]
}
```

```

    }
  },
  {
    "PlatformApplicationArn": "arn:aws:sns:us-west-2:123456789012:app/MPNS/
MyOtherApplication",
    "Attributes": {
      "SuccessFeedbackSampleRate": "100",
      "Enabled": "true"
    }
  }
]
}

```

- For API details, see [ListPlatformApplications](#) in *AWS CLI Command Reference*.

list-subscriptions-by-topic

The following code example shows how to use `list-subscriptions-by-topic`.

AWS CLI

To list the subscriptions associated with a topic

The following `list-subscriptions-by-topic` retrieves a list of SNS subscriptions associated with the specified topic.

```
aws sns list-subscriptions-by-topic \
  --topic-arn "arn:aws:sns:us-west-2:123456789012:my-topic"
```

Output:

```
{
  "Subscriptions": [
    {
      "Owner": "123456789012",
      "Endpoint": "my-email@example.com",
      "Protocol": "email",
      "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic",
      "SubscriptionArn": "arn:aws:sns:us-west-2:123456789012:my-
topic:8a21d249-4329-4871-acc6-7be709c6ea7f"
    }
  ]
}
```

```
}
```

- For API details, see [ListSubscriptionsByTopic](#) in *AWS CLI Command Reference*.

list-subscriptions

The following code example shows how to use `list-subscriptions`.

AWS CLI

To list your SNS subscriptions

The following `list-subscriptions` example displays a list of the SNS subscriptions in your AWS account.

```
aws sns list-subscriptions
```

Output:

```
{
  "Subscriptions": [
    {
      "Owner": "123456789012",
      "Endpoint": "my-email@example.com",
      "Protocol": "email",
      "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic",
      "SubscriptionArn": "arn:aws:sns:us-west-2:123456789012:my-
topic:8a21d249-4329-4871-acc6-7be709c6ea7f"
    }
  ]
}
```

- For API details, see [ListSubscriptions](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list tags for a topic

The following `list-tags-for-resource` example lists the tags for the specified Amazon SNS topic.

```
aws sns list-tags-for-resource \
  --resource-arn arn:aws:sns:us-west-2:123456789012:MyTopic
```

Output:

```
{
  "Tags": [
    {
      "Key": "Team",
      "Value": "Alpha"
    }
  ]
}
```

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

list-topics

The following code example shows how to use `list-topics`.

AWS CLI

To list your SNS topics

The following `list-topics` example lists all of SNS topics in your AWS account.

```
aws sns list-topics
```

Output:

```
{
  "Topics": [
    {
      "TopicArn": "arn:aws:sns:us-west-2:123456789012:my-topic"
    }
  ]
}
```

- For API details, see [ListTopics](#) in *AWS CLI Command Reference*.

opt-in-phone-number

The following code example shows how to use `opt-in-phone-number`.

AWS CLI

To opt-in for SMS messages

The following `opt-in-phone-number` example opts the specified phone number into receiving SMS messages.

```
aws sns opt-in-phone-number \  
  --phone-number +15555550100
```

This command produces no output.

- For API details, see [OptInPhoneNumber](#) in *AWS CLI Command Reference*.

publish

The following code example shows how to use `publish`.

AWS CLI

Example 1: To publish a message to a topic

The following `publish` example publishes the specified message to the specified SNS topic. The message comes from a text file, which enables you to include line breaks.

```
aws sns publish \  
  --topic-arn "arn:aws:sns:us-west-2:123456789012:my-topic" \  
  --message file://message.txt
```

Contents of `message.txt`:

```
Hello World  
Second Line
```

Output:

```
{
  "MessageId": "123a45b6-7890-12c3-45d6-111122223333"
}
```

Example 2: To publish an SMS message to a phone number

The following `publish` example publishes the message `Hello world!` to the phone number `+1-555-555-0100`.

```
aws sns publish \
  --message "Hello world!" \
  --phone-number +1-555-555-0100
```

Output:

```
{
  "MessageId": "123a45b6-7890-12c3-45d6-333322221111"
}
```

- For API details, see [Publish](#) in *AWS CLI Command Reference*.

put-data-protection-policy

The following code example shows how to use `put-data-protection-policy`.

AWS CLI

To set data protection policy

Example 1: To deny publishers from publishing messages with `CreditCardNumber`

The following `put-data-protection-policy` example denies publishers from publishing messages with `CreditCardNumber`.

```
aws sns put-data-protection-policy \
  --resource-arn arn:aws:sns:us-east-1:123456789012:mytopic \
  --data-protection-policy "{\"Name\":\"data_protection_policy\",\"Description\":"
  "\"Example data protection policy\",\"Version\":\"2021-06-01\",\"Statement\":"
  "\":[{\"DataDirection\":\"Inbound\",\"Principal\":[\"*\"],\"DataIdentifier\":"
  "\":[\"arn:aws:dataprotection::aws:data-identifier/CreditCardNumber\"],\"Operation\":"
  "\":{\"Deny\":{}}}]}"
```


This command produces no output.

Example 2: To load parameters from a file

The following `put-data-protection-policy` loads parameters from a file.

```
aws sns put-data-protection-policy \  
  --resource-arn arn:aws:sns:us-west-2:123456789012:MyTopic \  
  --data-protection-policy file://policy.json
```

This command produces no output.

- For API details, see [PutDataProtectionPolicy](#) in *AWS CLI Command Reference*.

remove-permission

The following code example shows how to use `remove-permission`.

AWS CLI

To remove a permission from a topic

The following `remove-permission` example removes the permission `Publish-Permission` from the specified topic.

```
aws sns remove-permission \  
  --topic-arn arn:aws:sns:us-west-2:123456789012:MyTopic \  
  --label Publish-Permission
```

This command produces no output.

- For API details, see [RemovePermission](#) in *AWS CLI Command Reference*.

set-endpoint-attributes

The following code example shows how to use `set-endpoint-attributes`.

AWS CLI

To set endpoint attributes

The following `set-endpoint-attributes` example disables the specified platform application endpoint.

```
aws sns set-endpoint-attributes \  
  --endpoint-arn arn:aws:sns:us-west-2:123456789012:endpoint/GCM/  
MyApplication/12345678-abcd-9012-efgh-345678901234 \  
  --attributes Enabled=false
```

Output:

```
{  
  "Attributes": {  
    "Enabled": "false",  
    "Token": "EXAMPLE12345..."  
  }  
}
```

- For API details, see [SetEndpointAttributes](#) in *AWS CLI Command Reference*.

set-platform-application-attributes

The following code example shows how to use `set-platform-application-attributes`.

AWS CLI

To set platform application attributes

The following `set-platform-application-attributes` example sets the `EventDeliveryFailure` attribute for the specified platform application to the ARN of the specified Amazon SNS topic.

```
aws sns set-platform-application-attributes \  
  --platform-application-arn arn:aws:sns:us-west-2:123456789012:app/GCM/  
MyApplication \  
  --attributes EventDeliveryFailure=arn:aws:sns:us-  
west-2:123456789012:AnotherTopic
```

This command produces no output.

- For API details, see [SetPlatformApplicationAttributes](#) in *AWS CLI Command Reference*.

set-sms-attributes

The following code example shows how to use `set-sms-attributes`.

AWS CLI

To set SMS message attributes

The following `set-sms-attributes` example sets the default sender ID for SMS messages to `MyName`.

```
aws sns set-sms-attributes \  
  --attributes DefaultSenderId=MyName
```

This command produces no output.

- For API details, see [SetSMSAttributes](#) in *AWS CLI Command Reference*.

set-subscription-attributes

The following code example shows how to use `set-subscription-attributes`.

AWS CLI

To set subscription attributes

The following `set-subscription-attributes` example sets the `RawMessageDelivery` attribute to an SQS subscription.

```
aws sns set-subscription-attributes \  
  --subscription-arn arn:aws:sns:us-  
east-1:123456789012:mytopic:f248de18-2cf6-578c-8592-b6f1eaa877dc \  
  --attribute-name RawMessageDelivery \  
  --attribute-value true
```

This command produces no output.

The following `set-subscription-attributes` example sets a `FilterPolicy` attribute to an SQS subscription.

```
aws sns set-subscription-attributes \  
  --subscription-arn arn:aws:sns:us-  
east-1:123456789012:mytopic:f248de18-2cf6-578c-8592-b6f1eaa877dc \  
  --attribute-name FilterPolicy \  
  --attribute-value true
```

```
--attribute-value "{ \"anyMandatoryKey\": [\"any\", \"of\", \"these\"] }"
```

This command produces no output.

The following `set-subscription-attributes` example removes the `FilterPolicy` attribute from an SQS subscription.

```
aws sns set-subscription-attributes \  
  --subscription-arn arn:aws:sns:us-  
east-1:123456789012:mytopic:f248de18-2cf6-578c-8592-b6f1eaa877dc \  
  --attribute-name FilterPolicy \  
  --attribute-value "{}"
```

This command produces no output.

- For API details, see [SetSubscriptionAttributes](#) in *AWS CLI Command Reference*.

set-topic-attributes

The following code example shows how to use `set-topic-attributes`.

AWS CLI

To set an attribute for a topic

The following `set-topic-attributes` example sets the `DisplayName` attribute for the specified topic.

```
aws sns set-topic-attributes \  
  --topic-arn arn:aws:sns:us-west-2:123456789012:MyTopic \  
  --attribute-name DisplayName \  
  --attribute-value MyTopicDisplayName
```

This command produces no output.

- For API details, see [SetTopicAttributes](#) in *AWS CLI Command Reference*.

subscribe

The following code example shows how to use `subscribe`.

AWS CLI

To subscribe to a topic

The following `subscribe` command subscribes an email address to the specified topic.

```
aws sns subscribe \  
  --topic-arn arn:aws:sns:us-west-2:123456789012:my-topic \  
  --protocol email \  
  --notification-endpoint my-email@example.com
```

Output:

```
{  
  "SubscriptionArn": "pending confirmation"  
}
```

- For API details, see [Subscribe](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add a tag to a topic

The following `tag-resource` example adds a metadata tag to the specified Amazon SNS topic.

```
aws sns tag-resource \  
  --resource-arn arn:aws:sns:us-west-2:123456789012:MyTopic \  
  --tags Key=Team,Value=Alpha
```

This command produces no output.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

unsubscribe

The following code example shows how to use `unsubscribe`.

AWS CLI

To unsubscribe from a topic

The following `unsubscribe` example deletes the specified subscription from a topic.

```
aws sns unsubscribe \  
  --subscription-arn arn:aws:sns:us-west-2:0123456789012:my-  
  topic:8a21d249-4329-4871-acc6-7be709c6ea7f
```

This command produces no output.

- For API details, see [Unsubscribe](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove a tag from a topic

The following `untag-resource` example removes any tags with the specified keys from the specified Amazon SNS topic.

```
aws sns untag-resource \  
  --resource-arn arn:aws:sns:us-west-2:123456789012:MyTopic \  
  --tag-keys Team
```

This command produces no output.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

Scenarios

Create a platform endpoint for push notifications

The following code example shows how to create a platform endpoint for Amazon SNS push notifications.

AWS CLI

To create a platform application endpoint

The following `create-platform-endpoint` example creates an endpoint for the specified platform application using the specified token.

```
aws sns create-platform-endpoint \  
  --platform-application-arn arn:aws:sns:us-west-2:123456789012:app/GCM/  
MyApplication \  
  --token EXAMPLE12345...
```

Output:

```
{  
  "EndpointArn": "arn:aws:sns:us-west-2:1234567890:endpoint/GCM/  
MyApplication/12345678-abcd-9012-efgh-345678901234"  
}
```

Amazon SQS examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon SQS.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-permission

The following code example shows how to use `add-permission`.

AWS CLI

To add a permission to a queue

This example enables the specified AWS account to send messages to the specified queue.

Command:

```
aws sqs add-permission --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue --label SendMessagesFromMyQueue --aws-account-ids 12345EXAMPLE --actions SendMessage
```

Output:

```
None.
```

- For API details, see [AddPermission](#) in *AWS CLI Command Reference*.

cancel-message-move-task

The following code example shows how to use `cancel-message-move-task`.

AWS CLI

To cancel a message move task

The following `cancel-message-move-task` example cancels the specified message move task.

```
aws sqs cancel-message-move-task \  
  --task-handle AQEB6nR4...Hz1vZQ==
```

Output:

```
{  
  "ApproximateNumberOfMessagesMoved": 102  
}
```

For more information, see [Amazon SQS API permissions: Actions and resource reference](#) in the *Developer Guide*.

- For API details, see [CancelMessageMoveTask](#) in *AWS CLI Command Reference*.

change-message-visibility-batch

The following code example shows how to use `change-message-visibility-batch`.

AWS CLI

To change multiple messages' timeout visibilities as a batch

This example changes the 2 specified messages' timeout visibilities to 10 hours (10 hours * 60 minutes * 60 seconds).

Command:

```
aws sqs change-message-visibility-batch --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue --entries file://change-message-visibility-batch.json
```

Input file (`change-message-visibility-batch.json`):

```
[
  {
    "Id": "FirstMessage",
    "ReceiptHandle": "AQEBhz2q...Jf3kaw==",
    "VisibilityTimeout": 36000
  },
  {
    "Id": "SecondMessage",
    "ReceiptHandle": "AQEBkTUH...HifSnw==",
    "VisibilityTimeout": 36000
  }
]
```

Output:

```
{
  "Successful": [
    {
      "Id": "SecondMessage"
    },
    {
```

```
    "Id": "FirstMessage"
  }
]
}
```

- For API details, see [ChangeMessageVisibilityBatch](#) in *AWS CLI Command Reference*.

change-message-visibility

The following code example shows how to use `change-message-visibility`.

AWS CLI

To change a message's timeout visibility

This example changes the specified message's timeout visibility to 10 hours (10 hours * 60 minutes * 60 seconds).

Command:

```
aws sqs change-message-visibility --queue-url https://sqs.us-
east-1.amazonaws.com/80398EXAMPLE/MyQueue --receipt-handle AQEBTpyI...t6HyQg== --
visibility-timeout 36000
```

Output:

```
None.
```

- For API details, see [ChangeMessageVisibility](#) in *AWS CLI Command Reference*.

create-queue

The following code example shows how to use `create-queue`.

AWS CLI

To create a queue

This example creates a queue with the specified name, sets the message retention period to 3 days (3 days * 24 hours * 60 minutes * 60 seconds), and sets the queue's dead letter queue to the specified queue with a maximum receive count of 1,000 messages.

Command:

```
aws sqs create-queue --queue-name MyQueue --attributes file://create-queue.json
```

Input file (create-queue.json):

```
{
  "RedrivePolicy": "{\"deadLetterTargetArn\":\"arn:aws:sqs:us-east-1:80398EXAMPLE:MyDeadLetterQueue\", \"maxReceiveCount\":\"1000\"}\",
  "MessageRetentionPeriod": "259200"
}
```

Output:

```
{
  "QueueUrl": "https://queue.amazonaws.com/80398EXAMPLE/MyQueue"
}
```

- For API details, see [CreateQueue](#) in *AWS CLI Command Reference*.

delete-message-batch

The following code example shows how to use `delete-message-batch`.

AWS CLI**To delete multiple messages as a batch**

This example deletes the specified messages.

Command:

```
aws sqs delete-message-batch --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue --entries file://delete-message-batch.json
```

Input file (delete-message-batch.json):

```
[
  {
    "Id": "FirstMessage",
    "ReceiptHandle": "AQEB1mg1...Z4GuLw=="
  }
]
```

```
  },
  {
    "Id": "SecondMessage",
    "ReceiptHandle": "AQEBLsYM...VQubAA=="
  }
]
```

Output:

```
{
  "Successful": [
    {
      "Id": "FirstMessage"
    },
    {
      "Id": "SecondMessage"
    }
  ]
}
```

- For API details, see [DeleteMessageBatch](#) in *AWS CLI Command Reference*.

delete-message

The following code example shows how to use delete-message.

AWS CLI

To delete a message

This example deletes the specified message.

Command:

```
aws sqs delete-message --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue --receipt-handle AQEBRXTo...q2doVA==
```

Output:

```
None.
```

- For API details, see [DeleteMessage](#) in *AWS CLI Command Reference*.

delete-queue

The following code example shows how to use `delete-queue`.

AWS CLI

To delete a queue

This example deletes the specified queue.

Command:

```
aws sqs delete-queue --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyNewerQueue
```

Output:

```
None.
```

- For API details, see [DeleteQueue](#) in *AWS CLI Command Reference*.

get-queue-attributes

The following code example shows how to use `get-queue-attributes`.

AWS CLI

To get a queue's attributes

This example gets all of the specified queue's attributes.

Command:

```
aws sqs get-queue-attributes --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue --attribute-names All
```

Output:

```
{
  "Attributes": {
    "ApproximateNumberOfMessagesNotVisible": "0",
```

```

    "RedrivePolicy": "{\\"deadLetterTargetArn\\":\\"arn:aws:sqs:us-
east-1:80398EXAMPLE:MyDeadLetterQueue\\",\\"maxReceiveCount\\":1000}",
    "MessageRetentionPeriod": "345600",
    "ApproximateNumberOfMessagesDelayed": "0",
    "MaximumMessageSize": "262144",
    "CreatedTimestamp": "1442426968",
    "ApproximateNumberOfMessages": "0",
    "ReceiveMessageWaitTimeSeconds": "0",
    "DelaySeconds": "0",
    "VisibilityTimeout": "30",
    "LastModifiedTimestamp": "1442426968",
    "QueueArn": "arn:aws:sqs:us-east-1:80398EXAMPLE:MyNewQueue"
  }
}

```

This example gets only the specified queue's maximum message size and visibility timeout attributes.

Command:

```

aws sqs get-queue-attributes --queue-url https://sqs.us-
east-1.amazonaws.com/80398EXAMPLE/MyNewQueue --attribute-names MaximumMessageSize
VisibilityTimeout

```

Output:

```

{
  "Attributes": {
    "VisibilityTimeout": "30",
    "MaximumMessageSize": "262144"
  }
}

```

- For API details, see [GetQueueAttributes](#) in *AWS CLI Command Reference*.

get-queue-url

The following code example shows how to use `get-queue-url`.

AWS CLI

To get a queue URL

This example gets the specified queue's URL.

Command:

```
aws sqs get-queue-url --queue-name MyQueue
```

Output:

```
{
  "QueueUrl": "https://queue.amazonaws.com/80398EXAMPLE/MyQueue"
}
```

- For API details, see [GetQueueUrl](#) in *AWS CLI Command Reference*.

list-dead-letter-source-queues

The following code example shows how to use `list-dead-letter-source-queues`.

AWS CLI

To list dead letter source queues

This example lists the queues that are associated with the specified dead letter source queue.

Command:

```
aws sqs list-dead-letter-source-queues --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyDeadLetterQueue
```

Output:

```
{
  "queueUrls": [
    "https://queue.amazonaws.com/80398EXAMPLE/MyQueue",
    "https://queue.amazonaws.com/80398EXAMPLE/MyOtherQueue"
  ]
}
```

- For API details, see [ListDeadLetterSourceQueues](#) in *AWS CLI Command Reference*.

list-message-move-tasks

The following code example shows how to use `list-message-move-tasks`.

AWS CLI

To list the message move tasks

The following `list-message-move-tasks` example lists the 2 most recent message move tasks in the specified queue.

```
aws sqs list-message-move-tasks \  
  --source-arn arn:aws:sqs:us-west-2:80398EXAMPLE:MyQueue \  
  --max-results 2
```

Output:

```
{  
  "Results": [  
    {  
      "TaskHandle": "AQEB6nR4...HzlvZQ==",  
      "Status": "RUNNING",  
      "SourceArn": "arn:aws:sqs:us-west-2:80398EXAMPLE:MyQueue1",  
      "DestinationArn": "arn:aws:sqs:us-west-2:80398EXAMPLE:MyQueue2",  
      "MaxNumberOfMessagesPerSecond": 50,  
      "ApproximateNumberOfMessagesMoved": 203,  
      "ApproximateNumberOfMessagesToMove": 30,  
      "StartedTimestamp": 1442428276921  
    },  
    {  
      "Status": "COMPLETED",  
      "SourceArn": "arn:aws:sqs:us-west-2:80398EXAMPLE:MyQueue1",  
      "DestinationArn": "arn:aws:sqs:us-west-2:80398EXAMPLE:MyQueue2",  
      "ApproximateNumberOfMessagesMoved": 29,  
      "ApproximateNumberOfMessagesToMove": 0,  
      "StartedTimestamp": 1342428272093  
    }  
  ]  
}
```

For more information, see [Amazon SQS API permissions: Actions and resource reference](#) in the *Developer Guide*.

- For API details, see [ListMessageMoveTasks](#) in *AWS CLI Command Reference*.

list-queue-tags

The following code example shows how to use `list-queue-tags`.

AWS CLI

To list all cost allocation tags for a queue

The following `list-queue-tags` example displays all of the cost allocation tags associated with the specified queue.

```
aws sqs list-queue-tags \  
  --queue-url https://sqs.us-west-2.amazonaws.com/123456789012/MyQueue
```

Output:

```
{  
  "Tags": {  
    "Team": "Alpha"  
  }  
}
```

For more information, see [Listing Cost Allocation Tags](#) in the *Amazon Simple Queue Service Developer Guide*.

- For API details, see [ListQueueTags](#) in *AWS CLI Command Reference*.

list-queues

The following code example shows how to use `list-queues`.

AWS CLI

To list queues

This example lists all queues.

Command:

```
aws sqs list-queues
```

Output:

```
{
  "QueueUrls": [
    "https://queue.amazonaws.com/80398EXAMPLE/MyDeadLetterQueue",
    "https://queue.amazonaws.com/80398EXAMPLE/MyQueue",
    "https://queue.amazonaws.com/80398EXAMPLE/MyOtherQueue",
    "https://queue.amazonaws.com/80398EXAMPLE/TestQueue1",
    "https://queue.amazonaws.com/80398EXAMPLE/TestQueue2"
  ]
}
```

This example lists only queues that start with "My".

Command:

```
aws sqs list-queues --queue-name-prefix My
```

Output:

```
{
  "QueueUrls": [
    "https://queue.amazonaws.com/80398EXAMPLE/MyDeadLetterQueue",
    "https://queue.amazonaws.com/80398EXAMPLE/MyQueue",
    "https://queue.amazonaws.com/80398EXAMPLE/MyOtherQueue"
  ]
}
```

- For API details, see [ListQueues](#) in *AWS CLI Command Reference*.

purge-queue

The following code example shows how to use `purge-queue`.

AWS CLI**To purge a queue**

This example deletes all messages in the specified queue.

Command:

```
aws sqs purge-queue --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyNewQueue
```

Output:

```
None.
```

- For API details, see [PurgeQueue](#) in *AWS CLI Command Reference*.

receive-message

The following code example shows how to use `receive-message`.

AWS CLI

To receive a message

This example receives up to 10 available messages, returning all available attributes.

Command:

```
aws sqs receive-message --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue --attribute-names All --message-attribute-names All --max-number-of-messages 10
```

Output:

```
{
  "Messages": [
    {
      "Body": "My first message.",
      "ReceiptHandle": "AQEBzbVv...fqNzFw==",
      "MD5ofBody": "1000f835...a35411fa",
      "MD5ofMessageAttributes": "9424c491...26bc3ae7",
      "MessageId": "d6790f8d-d575-4f01-bc51-40122EXAMPLE",
      "Attributes": {
        "ApproximateFirstReceiveTimestamp": "1442428276921",
        "SenderId": "AIDAIAZKMSNQ7TEXAMPLE",
        "ApproximateReceiveCount": "5",
        "SentTimestamp": "1442428276921"
      },
      "MessageAttributes": {
```

```

    "PostalCode": {
      "DataType": "String",
      "StringValue": "ABC123"
    },
    "City": {
      "DataType": "String",
      "StringValue": "Any City"
    }
  }
}
]
}

```

This example receives the next available message, returning only the `SenderId` and `SentTimestamp` attributes as well as the `PostalCode` message attribute.

Command:

```

aws sqs receive-message --queue-url https://sqs.us-
east-1.amazonaws.com/80398EXAMPLE/MyQueue --attribute-names SenderId SentTimestamp
--message-attribute-names PostalCode

```

Output:

```

{
  "Messages": [
    {
      "Body": "My first message.",
      "ReceiptHandle": "AQEB6nR4...HzlvZQ==",
      "MD5OfBody": "1000f835...a35411fa",
      "MD5OfMessageAttributes": "b8e89563...e088e74f",
      "MessageId": "d6790f8d-d575-4f01-bc51-40122EXAMPLE",
      "Attributes": {
        "SenderId": "AIDAIKMSNQ7TEXAMPLE",
        "SentTimestamp": "1442428276921"
      },
      "MessageAttributes": {
        "PostalCode": {
          "DataType": "String",
          "StringValue": "ABC123"
        }
      }
    }
  ]
}

```

```
]
}
```

- For API details, see [ReceiveMessage](#) in *AWS CLI Command Reference*.

remove-permission

The following code example shows how to use `remove-permission`.

AWS CLI

To remove a permission

This example removes the permission with the specified label from the specified queue.

Command:

```
aws sqs remove-permission --queue-url https://sqs.us-
east-1.amazonaws.com/80398EXAMPLE/MyQueue --label SendMessagesFromMyQueue
```

Output:

```
None.
```

- For API details, see [RemovePermission](#) in *AWS CLI Command Reference*.

send-message-batch

The following code example shows how to use `send-message-batch`.

AWS CLI

To send multiple messages as a batch

This example sends 2 messages with the specified message bodies, delay periods, and message attributes, to the specified queue.

Command:

```
aws sqs send-message-batch --queue-url https://sqs.us-
east-1.amazonaws.com/80398EXAMPLE/MyQueue --entries file://send-message-batch.json
```

Input file (send-message-batch.json):

```
[
  {
    "Id": "FuelReport-0001-2015-09-16T140731Z",
    "MessageBody": "Fuel report for account 0001 on 2015-09-16 at 02:07:31 PM.",
    "DelaySeconds": 10,
    "MessageAttributes": {
      "SellerName": {
        "DataType": "String",
        "StringValue": "Example Store"
      },
      "City": {
        "DataType": "String",
        "StringValue": "Any City"
      },
      "Region": {
        "DataType": "String",
        "StringValue": "WA"
      },
      "PostalCode": {
        "DataType": "String",
        "StringValue": "99065"
      },
      "PricePerGallon": {
        "DataType": "Number",
        "StringValue": "1.99"
      }
    }
  },
  {
    "Id": "FuelReport-0002-2015-09-16T140930Z",
    "MessageBody": "Fuel report for account 0002 on 2015-09-16 at 02:09:30 PM.",
    "DelaySeconds": 10,
    "MessageAttributes": {
      "SellerName": {
        "DataType": "String",
        "StringValue": "Example Fuels"
      },
      "City": {
        "DataType": "String",
        "StringValue": "North Town"
      },
      "Region": {
```

```

        "DataType": "String",
        "StringValue": "WA"
    },
    "PostalCode": {
        "DataType": "String",
        "StringValue": "99123"
    },
    "PricePerGallon": {
        "DataType": "Number",
        "StringValue": "1.87"
    }
}
]

```

Output:

```

{
  "Successful": [
    {
      "MD50fMessageBody": "203c4a38...7943237e",
      "MD50fMessageAttributes": "10809b55...baf283ef",
      "Id": "FuelReport-0001-2015-09-16T140731Z",
      "MessageId": "d175070c-d6b8-4101-861d-adeb3EXAMPLE"
    },
    {
      "MD50fMessageBody": "2cf0159a...c1980595",
      "MD50fMessageAttributes": "55623928...ae354a25",
      "Id": "FuelReport-0002-2015-09-16T140930Z",
      "MessageId": "f9b7d55d-0570-413e-b9c5-a9264EXAMPLE"
    }
  ]
}

```

- For API details, see [SendMessageBatch](#) in *AWS CLI Command Reference*.

send-message

The following code example shows how to use send-message.

AWS CLI

To send a message

This example sends a message with the specified message body, delay period, and message attributes, to the specified queue.

Command:

```
aws sqs send-message --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyQueue --message-body "Information about the largest city in Any Region." --delay-seconds 10 --message-attributes file://send-message.json
```

Input file (send-message.json):

```
{
  "City": {
    "DataType": "String",
    "StringValue": "Any City"
  },
  "Greeting": {
    "DataType": "Binary",
    "BinaryValue": "Hello, World!"
  },
  "Population": {
    "DataType": "Number",
    "StringValue": "1250800"
  }
}
```

Output:

```
{
  "MD5ofMessageBody": "51b0a325...39163aa0",
  "MD5ofMessageAttributes": "00484c68...59e48f06",
  "MessageId": "da68f62c-0c07-4bee-bf5f-7e856EXAMPLE"
}
```

- For API details, see [SendMessage](#) in *AWS CLI Command Reference*.

set-queue-attributes

The following code example shows how to use `set-queue-attributes`.

AWS CLI

To set queue attributes

This example sets the specified queue to a delivery delay of 10 seconds, a maximum message size of 128 KB (128 KB * 1,024 bytes), a message retention period of 3 days (3 days * 24 hours * 60 minutes * 60 seconds), a receive message wait time of 20 seconds, and a default visibility timeout of 60 seconds. This example also associates the specified dead letter queue with a maximum receive count of 1,000 messages.

Command:

```
aws sqs set-queue-attributes --queue-url https://sqs.us-east-1.amazonaws.com/80398EXAMPLE/MyNewQueue --attributes file://set-queue-attributes.json
```

Input file (set-queue-attributes.json):

```
{
  "DelaySeconds": "10",
  "MaximumMessageSize": "131072",
  "MessageRetentionPeriod": "259200",
  "ReceiveMessageWaitTimeSeconds": "20",
  "RedrivePolicy": "{\"deadLetterTargetArn\":\"arn:aws:sqs:us-east-1:80398EXAMPLE:MyDeadLetterQueue\",\"maxReceiveCount\":\"1000\"}",
  "VisibilityTimeout": "60"
}
```

Output:

```
None.
```

- For API details, see [SetQueueAttributes](#) in *AWS CLI Command Reference*.

start-message-move-task

The following code example shows how to use `start-message-move-task`.

AWS CLI

*Example 1: *To start a message move task**

The following `start-message-move-task` example starts a message move task to redrive messages from the specified dead-letter queue to the source queue.

```
aws sqs start-message-move-task \  
  --source-arn arn:aws:sqs:us-west-2:80398EXAMPLE:MyQueue
```

Output:

```
{  
  "TaskHandle": "AQEB6nR4...Hz1vZQ=="  
}
```

For more information, see [This is the topic title](#) in the *Name of your guide*.

*Example 2: *To start a message move task with a maximum rate**

The following `start-message-move-task` example starts a message move task to redrive messages from the specified dead-letter queue to the specified destination queue at a maximum rate of 50 messages per second.

```
aws sqs start-message-move-task \  
  --source-arn arn:aws:sqs:us-west-2:80398EXAMPLE:MyQueue1 \  
  --destination-arn arn:aws:sqs:us-west-2:80398EXAMPLE:MyQueue2 \  
  --max-number-of-messages-per-second 50
```

Output:

```
{  
  "TaskHandle": "AQEB6nR4...Hz1vZQ=="  
}
```

For more information, see [Amazon SQS API permissions: Actions and resource reference](#) in the *Developer Guide*.

- For API details, see [StartMessageMoveTask](#) in *AWS CLI Command Reference*.

tag-queue

The following code example shows how to use `tag-queue`.

AWS CLI

To add cost allocation tags to a queue

The following `tag-queue` example adds a cost allocation tag to the specified Amazon SQS queue.

```
aws sqs tag-queue \  
  --queue-url https://sqs.us-west-2.amazonaws.com/123456789012/MyQueue \  
  --tags Priority=Highest
```

This command produces no output.

For more information, see [Adding Cost Allocation Tags](#) in the *Amazon Simple Queue Service Developer Guide*.

- For API details, see [TagQueue](#) in *AWS CLI Command Reference*.

untag-queue

The following code example shows how to use `untag-queue`.

AWS CLI

To remove cost allocation tags from a queue

The following `untag-queue` example removes a cost allocation tag from the specified Amazon SQS queue.

```
aws sqs untag-queue \  
  --queue-url https://sqs.us-west-2.amazonaws.com/123456789012/MyQueue \  
  --tag-keys "Priority"
```

This command produces no output.

For more information, see [Adding Cost Allocation Tags](#) in the *Amazon Simple Queue Service Developer Guide*.

- For API details, see [UntagQueue](#) in *AWS CLI Command Reference*.

Storage Gateway examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Storage Gateway.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

describe-gateway-information

The following code example shows how to use `describe-gateway-information`.

AWS CLI

To describe a gateway

The following `describe-gateway-information` command returns metadata about the specified gateway. To specify which gateway to describe, use the Amazon Resource Name (ARN) of the gateway in the command.

This examples specifies a gateway with the id `sgw-12A3456B` in account `123456789012`:

```
aws storagegateway describe-gateway-information --gateway-arn
"arn:aws:storagegateway:us-west-2:123456789012:gateway/sgw-12A3456B"
```

This command outputs a JSON block that contains metadata about about the gateway such as its name, network interfaces, configured time zone, and the state (whether the gateway is running or not).

- For API details, see [DescribeGatewayInformation](#) in *AWS CLI Command Reference*.

list-file-shares

The following code example shows how to use `list-file-shares`.

AWS CLI

To list file shares

The following command-name example lists the available widgets in your AWS account.

```
aws storagegateway list-file-shares \  
  --gateway-arn arn:aws:storagegateway:us-east-1:209870788375:gateway/sgw-FB02E292
```

Output:

```
{  
  "FileShareInfoList": [  
    {  
      "FileShareType": "NFS",  
      "FileShareARN": "arn:aws:storagegateway:us-east-1:111122223333:share/  
share-2FA12345",  
      "FileShareId": "share-2FA12345",  
      "FileShareStatus": "AVAILABLE",  
      "GatewayARN": "arn:aws:storagegateway:us-east-1:111122223333:gateway/  
sgw-FB0AAAAA"  
    }  
  ],  
  "Marker": null  
}
```

For more information, see [ListFileShares](#) in the *AWS Storage Gateway Service API Reference*.

- For API details, see [ListFileShares](#) in *AWS CLI Command Reference*.

list-gateways

The following code example shows how to use `list-gateways`.

AWS CLI

To list gateways for an account

The following `list-gateways` command lists all the gateways defined for an account:

```
aws storagegateway list-gateways
```

This command outputs a JSON block that contains a list of gateway Amazon Resource Names (ARNs).

- For API details, see [ListGateways](#) in *AWS CLI Command Reference*.

list-volumes

The following code example shows how to use `list-volumes`.

AWS CLI

To list the volumes configured for a gateway

The following `list-volumes` command returns a list of volumes configured for the specified gateway. To specify which gateway to describe, use the Amazon Resource Name (ARN) of the gateway in the command.

This examples specifies a gateway with the id `sgw-12A3456B` in account `123456789012`:

```
aws storagegateway list-volumes --gateway-arn "arn:aws:storagegateway:us-west-2:123456789012:gateway/sgw-12A3456B"
```

This command outputs a JSON block that a list of volumes that includes the type and ARN for each volume.

- For API details, see [ListVolumes](#) in *AWS CLI Command Reference*.

refresh-cache

The following code example shows how to use `refresh-cache`.

AWS CLI

To refresh the file share cache

The following `refresh-cache` example refreshes the cache for the specified file share.

```
aws storagegateway refresh-cache \  
  --file-share-arn arn:aws:storagegateway:us-east-1:111122223333:share/  
share-2FA12345
```

Output:

```
{
  "FileShareARN": "arn:aws:storagegateway:us-east-1:111122223333:share/
share-2FA12345",
  "NotificationId": "4954d4b1-abcd-ef01-1234-97950a7d3483"
}
```

For more information, see [ListFileShares](#) in the *AWS Storage Gateway Service API Reference*.

- For API details, see [RefreshCache](#) in *AWS CLI Command Reference*.

AWS STS examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS STS.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

assume-role-with-saml

The following code example shows how to use `assume-role-with-saml`.

AWS CLI

To get short-term credentials for a role authenticated with SAML

The following `assume-role-with-saml` command retrieves a set of short-term credentials for the IAM role `TestSaml`. The request in this example is authenticated by using the SAML assertion supplied by your identity provider when you authenticate to it.

```
aws sts assume-role-with-saml \
  --role-arn arn:aws:iam::123456789012:role/TestSaml \
  --principal-arn arn:aws:iam::123456789012:saml-provider/SAML-test \
  --saml-assertion
"VERYLONGENCODEDASSERTIONEXAMPLExzYw1s0kF1ZG11bmN1PmJsYW5rPC9zYW1s0kF1ZG11bmN1Pjwvc2FtbDpBd
+PHNhbWw6TmFtZU1EIEZvcmlhdD0idXJu0m9hc2lz0m5hbWVzOnRj01NBTUw6Mi4w0m5hbWVpZC1mb3JtYXQ6dHJhbnN
+PHNhbWw6U3ViamVjdENvbWZpcmlhdGlvbiBNZXRob2Q9InVybjpvcyXNpczpuYW1lc3p0YzpzTQU1M0jIuMDpjbTpiZWw="
```

Output:

```
{
  "Issuer": "https://integ.example.com/idp/shibboleth</Issuer",
  "AssumedRoleUser": {
    "Arn": "arn:aws:sts::123456789012:assumed-role/TestSaml",
    "AssumedRoleId": "AR0456EXAMPLE789:TestSaml"
  },
  "Credentials": {
    "AccessKeyId": "ASIAV3ZUEFP6EXAMPLE",
    "SecretAccessKey": "8P+SQvWIuLnKhh8d++jpw0nNmQRBZvNEXAMPLEKEY",
    "SessionToken": "IQoJb3JpZ2luX2VjE0z//////////
wEXAMPLEtMSJHMEUCIDoKK3JH9uGQE1z0sINr5M4jk
+Na8KHDcCYRVjJCZEv0AiEA30vJGtw1EcVi0leS2vhs8VdCKFJQWPQrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburED
+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
+scqKmlzm8FDrypNC9Yjc8fP0Ln9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCR/oLxBA==",
    "Expiration": "2019-11-01T20:26:47Z"
  },
  "Audience": "https://signin.aws.amazon.com/saml",
  "SubjectType": "transient",
  "PackedPolicySize": "6",
  "NameQualifier": "SbdG0nUkh1i4+EXAMPLExL/jEvs=",
  "Subject": "SamlExample"
}
```

For more information, see [Requesting Temporary Security Credentials](#) in the *AWS IAM User Guide*.

- For API details, see [AssumeRoleWithSaml](#) in *AWS CLI Command Reference*.

assume-role-with-web-identity

The following code example shows how to use `assume-role-with-web-identity`.

AWS CLI

To get short-term credentials for a role authenticated with Web Identity (OAuth 2.0)

The following `assume-role-with-web-identity` command retrieves a set of short-term credentials for the IAM role `app1`. The request is authenticated by using the web identity token supplied by the specified web identity provider. Two additional policies are applied to the session to further restrict what the user can do. The returned credentials expire one hour after they are generated.

```
aws sts assume-role-with-web-identity \
  --duration-seconds 3600 \
  --role-session-name "app1" \
  --provider-id "www.amazon.com" \
  --policy-arns "arn:aws:iam::123456789012:policy/
q=webidentitydemopolicy1","arn:aws:iam::123456789012:policy/webidentitydemopolicy2"
  \
  --role-arn arn:aws:iam::123456789012:role/FederatedWebIdentityRole \
  --web-identity-token "Atza
%7CIQEBLjAsAhRFiXuWpUXuRvQ9PZL3GMFcYevydwIUFAHZwXZXXXXXXXXXJnrulxKDHwy87oGKPznh0D6bEQZTSCzyoC
CrKqjG7nPBjNIL016GGvuS5gSvPRUxWES3VYfm1w17WTI7jn-Pcb6M-
buCgHhF0zTQxod27L9Cqn0Lio7N3gZAGpsp6n1-
AJB0CJckcyXe2c6uD0sr0JeZ1KUm2eTDVMf8IehDVI0r1Q0nTV6KzzAI30Y87Vd_cVMQ"
```

Output:

```
{
  "SubjectFromWebIdentityToken": "amzn1.account.AF6RH07KZU5XRVQJGXX6HB56KR2A"
  "Audience": "client.5498841531868486423.1548@apps.example.com",
  "AssumedRoleUser": {
    "Arn": "arn:aws:sts::123456789012:assumed-role/FederatedWebIdentityRole/
app1",
    "AssumedRoleId": "AROACLKWSQRAOEXAMPLE:app1"
  }
  "Credentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJa1rXUtnFEMI/K7MDENG/bPxrFiCYzEXAMPLEKEY",
    "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvqwqKwRc0IfrRh3c/LTo6UDdyJw00vEVPvLXCrrrUtdnniCEXAMPLE/"
  }
}
```

```
IvU1dYUg2RVAJBanLiHb4IgrmpRV3zrkuWJ0gQs8IZZaIv2BXIa2R40l9kBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wn0E49mRl/+0tkIKG07fAE",
  "Expiration": "2020-05-19T18:06:10+00:00"
},
"Provider": "www.amazon.com"
}
```

For more information, see [Requesting Temporary Security Credentials](#) in the *AWS IAM User Guide*.

- For API details, see [AssumeRoleWithWebIdentity](#) in *AWS CLI Command Reference*.

assume-role

The following code example shows how to use `assume-role`.

AWS CLI

To assume a role

The following `assume-role` command retrieves a set of short-term credentials for the IAM role `s3-access-example`.

```
aws sts assume-role \
  --role-arn arn:aws:iam::123456789012:role/xaccounts3access \
  --role-session-name s3-access-example
```

Output:

```
{
  "AssumedRoleUser": {
    "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:s3-access-example",
    "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-access-
example"
  },
  "Credentials": {
    "SecretAccessKey": "9drTJvcXLB89EXAMPLEELB8923FB892xMFI",
    "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/qwjzP2iEXAMPLEbw/
m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtY1FVgAUj
+7Indz3LU0aTWk1WKIjHmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
```

```
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFIPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8BRi2
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo50ytwU=",
    "Expiration": "2016-03-15T00:05:07Z",
    "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
  }
}
```

The output of the command contains an access key, secret key, and session token that you can use to authenticate to AWS.

For AWS CLI use, you can set up a named profile associated with a role. When you use the profile, the AWS CLI will call `assume-role` and manage credentials for you. For more information, see [Use an IAM role in the AWS CLI](#) in the *AWS CLI User Guide*.

- For API details, see [AssumeRole](#) in *AWS CLI Command Reference*.

decode-authorization-message

The following code example shows how to use `decode-authorization-message`.

AWS CLI

To decode an encoded authorization message returned in response to a request

The following `decode-authorization-message` example decodes additional information about the authorization status of a request from an encoded message returned in response to an Amazon Web Services request.

```
aws sts decode-authorization-message \
  --encoded-message EXAMPLEwodyRNrtlQARDip-
eTA6i6Dr1UhhPQrLWB_lAb15pAKx19mPDlexYcGBreyIKQC1BGBIPBKr3dFDkwqe07e2NMk5j_hmzAiChJN-8oy3Ewi
Ojau7BMj0TWw0tHPHv_Zaz87yENDipr745EjQwRd5LaoL3vN8_5ZfA9UiBMKDgVh1gjqZJFUiQoubv78V1RbHNYnK44E
p0u3FZjwYStfvTb3GHs3-6rLribG09jZ0ktkfE6vqx1FzLyeDr4P2ihC1wty9tArCvvGzIAUNmARQJ2VWVPxioqgoqCz
JWP5pwe_mAyqh0NLw-r1S56YC_90onj9A80sNrHlI-
tIiNd7tgNTYzDuPQYD2FMDNBp82V9eVmYGtPp5NIeSpuf3f0HanFuBZgENxZQZ2d1H3xJGMTtYayzZrRXjiq_SfX9zeB
FaoPIb8LmmKVBLpIB0iFhU9sEHPqKHVPi6jdxXqKaZaFGvYVmV0iuQdNQKuyk0p067P0FrZECLjj0tNPB0ZCcuEKEXAM
```

Output:

```
{
  "DecodedMessage": "{\"allowed\":false,\"explicitDeny\":true,\"matchedStatements
\":{\"items\":[{\"statementId\":\"VisualEditor0\",\"effect\":\"DENY\",\"principals
```

```

\":{\\"items\\":[{\\"value\\":\\"ARO123456789EXAMPLE\\"}]},\\"principalGroups
\\":{\\"items\\":[]},\\"actions\\":{\\"items\\":[{\\"value\\":\\"ec2:RunInstances
\\"}]},\\"resources\\":{\\"items\\":[{\\"value\\":\\"*\\"}]},\\"conditions\\":{\\"items
\\":[[]]}},\\"failures\\":{\\"items\\":[]},\\"context\\":{\\"principal\\":{\\"id\\":
\\"ARO123456789EXAMPLE:Ana\\",\\"arn\\":\\"arn:aws:sts:111122223333:assumed-role/
Developer/Ana\\"},\\"action\\":\\"RunInstances\\",\\"resource\\":\\"arn:aws:ec2:us-
east-1:111122223333:instance/*\\"},\\"conditions\\":{\\"items\\":[{\\"key\\":
\\"ec2:MetadataHttpPutResponseHopLimit\\",\\"values\\":{\\"items\\":[{\\"value\\":
\\"2\\"}]},{\\"key\\":\\"ec2:InstanceMarketType\\",\\"values\\":{\\"items\\":[{\\"value
\\":\\"on-demand\\"}]},{\\"key\\":\\"aws:Resource\\",\\"values\\":{\\"items\\":[{\\"value
\\":\\"instance/*\\"}]},{\\"key\\":\\"aws:Account\\",\\"values\\":{\\"items\\":[{\\"value
\\":\\"111122223333\\"}]},{\\"key\\":\\"ec2:AvailabilityZone\\",\\"values\\":{\\"items\\":
[{\\"value\\":\\"us-east-1f\\"}]},{\\"key\\":\\"ec2:ecsOptimized\\",\\"values\\":{\\"items
\\":[{\\"value\\":\\"false\\"}]},{\\"key\\":\\"ec2:IsLaunchTemplateResource\\",\\"values
\\":{\\"items\\":[{\\"value\\":\\"false\\"}]},{\\"key\\":\\"ec2:InstanceType\\",\\"values\\":
{\\"items\\":[{\\"value\\":\\"t2.micro\\"}]},{\\"key\\":\\"ec2:RootDeviceType\\",\\"values
\\":{\\"items\\":[{\\"value\\":\\"efs\\"}]},{\\"key\\":\\"aws:Region\\",\\"values\\":{\\"items
\\":[{\\"value\\":\\"us-east-1\\"}]},{\\"key\\":\\"ec2:MetadataHttpEndpoint\\",\\"values
\\":{\\"items\\":[{\\"value\\":\\"enabled\\"}]},{\\"key\\":\\"aws:Service\\",\\"values\\":
{\\"items\\":[{\\"value\\":\\"ec2\\"}]},{\\"key\\":\\"ec2:InstanceID\\",\\"values\\":{\\"items
\\":[{\\"value\\":\\"*\\"}]},{\\"key\\":\\"ec2:MetadataHttpTokens\\",\\"values\\":{\\"items
\\":[{\\"value\\":\\"required\\"}]},{\\"key\\":\\"aws:Type\\",\\"values\\":{\\"items\\":
[{\\"value\\":\\"instance\\"}]},{\\"key\\":\\"ec2:Tenancy\\",\\"values\\":{\\"items\\":
[{\\"value\\":\\"default\\"}]},{\\"key\\":\\"ec2:Region\\",\\"values\\":{\\"items\\":[{\\"value
\\":\\"us-east-1\\"}]},{\\"key\\":\\"aws:ARN\\",\\"values\\":{\\"items\\":[{\\"value\\":
\\"arn:aws:ec2:us-east-1:111122223333:instance/*\\"}]}}]}]}"}
}

```

For more information, see [Policy evaluation logic](#) in the *AWS IAM User Guide*.

- For API details, see [DecodeAuthorizationMessage](#) in *AWS CLI Command Reference*.

get-caller-identity

The following code example shows how to use `get-caller-identity`.

AWS CLI

To get details about the current IAM identity

The following `get-caller-identity` command displays information about the IAM identity used to authenticate the request. The caller is an IAM user.

```
aws sts get-caller-identity
```

Output:

```
{
  "UserId": "AIDASAMPLEUSERID",
  "Account": "123456789012",
  "Arn": "arn:aws:iam::123456789012:user/DevAdmin"
}
```

- For API details, see [GetCallerIdentity](#) in *AWS CLI Command Reference*.

get-federation-token

The following code example shows how to use `get-federation-token`.

AWS CLI

To return a set of temporary security credentials using IAM user access key credentials

The following `get-federation-token` example returns a set of temporary security credentials (consisting of an access key ID, a secret access key, and a security token) for a user. You must call the `GetFederationToken` operation using the long-term security credentials of an IAM user.

```
aws sts get-federation-token \
  --name Bob \
  --policy file://myfile.json \
  --policy-arns arn=arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess \
  --duration-seconds 900
```

Contents of `myfile.json`:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:Describe*",

```

```

    "Resource": "*"
  },
  {
    "Effect": "Allow",
    "Action": "elasticloadbalancing:Describe*",
    "Resource": "*"
  },
  {
    "Effect": "Allow",
    "Action": [
      "cloudwatch:ListMetrics",
      "cloudwatch:GetMetricStatistics",
      "cloudwatch:Describe*"
    ],
    "Resource": "*"
  },
  {
    "Effect": "Allow",
    "Action": "autoscaling:Describe*",
    "Resource": "*"
  }
]
}

```

Output:

```

{
  "Credentials": {
    "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "EXAMPLEpZ21uX2VjEGoaCXVzLXdlc3QtMiJIMEYCIQC/
W9pL5ArQyDD5JwFL3/h5+WGopQ24GEXweNctwhi9sgIhAMkg
+MZE35iWM8s4r5Lr25f9rSTVPFH98G42QQunWMTfKq0DCOP/////////
wEQAxoMNDUy0TI1MTcwNTA3Igxuy3A0puuoLsk3MJwqgQPg8Q0d9HuoClUxq26wnc/nm
+eZLjHDyGf2KUAHK2DuaS/nrGSEXAMPLE",
    "Expiration": "2023-12-20T02:06:07+00:00"
  },
  "FederatedUser": {
    "FederatedUserId": "111122223333:Bob",
    "Arn": "arn:aws:sts::111122223333:federated-user/Bob"
  },
  "PackedPolicySize": 36
}

```

For more information, see [Requesting Temporary Security Credentials](#) in the *AWS IAM User Guide*.

- For API details, see [GetFederationToken](#) in *AWS CLI Command Reference*.

get-session-token

The following code example shows how to use `get-session-token`.

AWS CLI

To get a set of short term credentials for an IAM identity

The following `get-session-token` command retrieves a set of short-term credentials for the IAM identity making the call. The resulting credentials can be used for requests where multi-factor authentication (MFA) is required by policy. The credentials expire 15 minutes after they are generated.

```
aws sts get-session-token \  
  --duration-seconds 900 \  
  --serial-number "YourMFADeviceSerialNumber" \  
  --token-code 123456
```

Output:

```
{  
  "Credentials": {  
    "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",  
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",  
    "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4B1CFFxWNE1OPTgk5TthT  
+FvwqnKwRc0IfrrRh3c/LTo6UDdyJw00vEVPvLXCrrrUtdnniCEXAMPLE/  
IvU1dYUg2RVAJBanLiHb4IgrmpRV3zrkuWJ0gQs8IZZaIv2BXIa2R40lgkBN9bkUDNCJiBeb/  
AX1zBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wn0E49mRl/+0tkIKG07fAE",  
    "Expiration": "2020-05-19T18:06:10+00:00"  
  }  
}
```

For more information, see [Requesting Temporary Security Credentials](#) in the *AWS IAM User Guide*.

- For API details, see [GetSessionToken](#) in *AWS CLI Command Reference*.

AWS Support examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS Support.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-attachments-to-set

The following code example shows how to use `add-attachments-to-set`.

AWS CLI

To add an attachment to a set

The following `add-attachments-to-set` example adds an image to a set that you can then specify for a support case in your AWS account.

```
aws support add-attachments-to-set \  
  --attachment-set-id "as-2f5a6faa2a4a1e600-mu-nk5xQ1Br70-  
G1cUos5LZkd38K0AHZa9BMDVzNEXAMPLE" \  
  --attachments fileName=troubleshoot-screenshot.png,data=base64-encoded-string
```

Output:

```
{  
  "attachmentSetId": "as-2f5a6faa2a4a1e600-mu-nk5xQ1Br70-  
G1cUos5LZkd38K0AHZa9BMDVzNEXAMPLE",
```



```
"expiryTime": "2020-05-14T17:04:40.790+0000"  
}
```

For more information, see [Case management](#) in the *AWS Support User Guide*.

- For API details, see [AddAttachmentsToSet](#) in *AWS CLI Command Reference*.

add-communication-to-case

The following code example shows how to use `add-communication-to-case`.

AWS CLI

To add communication to a case

The following `add-communication-to-case` example adds communications to a support case in your AWS account.

```
aws support add-communication-to-case \  
  --case-id "case-12345678910-2013-c4c1d2bf33c5cf47" \  
  --communication-body "I'm attaching a set of images to this case." \  
  --cc-email-addresses "myemail@example.com" \  
  --attachment-set-id "as-2f5a6faa2a4a1e600-mu-nk5xQ1Br70-  
G1cUos5LZkd38K0AHZa9BMDVzNEXAMPLE"
```

Output:

```
{  
  "result": true  
}
```

For more information, see [Case management](#) in the *AWS Support User Guide*.

- For API details, see [AddCommunicationToCase](#) in *AWS CLI Command Reference*.

create-case

The following code example shows how to use `create-case`.

AWS CLI

To create a case

The following `create-case` example creates a support case for your AWS account.

```
aws support create-case \  
  --category-code "using-aws" \  
  --cc-email-addresses "myemail@example.com" \  
  --communication-body "I want to learn more about an AWS service." \  
  --issue-type "technical" \  
  --language "en" \  
  --service-code "general-info" \  
  --severity-code "low" \  
  --subject "Question about my account"
```

Output:

```
{  
  "caseId": "case-12345678910-2013-c4c1d2bf33c5cf47"  
}
```

For more information, see [Case management](#) in the *AWS Support User Guide*.

- For API details, see [CreateCase](#) in *AWS CLI Command Reference*.

describe-attachment

The following code example shows how to use `describe-attachment`.

AWS CLI

To describe an attachment

The following `describe-attachment` example returns information about the attachment with the specified ID.

```
aws support describe-attachment \  
  --attachment-id "attachment-KBnjRNrePd9D6Jx0-Mm00xZuDEaL2JAj_0-  
gJv9qqDooTipsz3V1Nb19rCfkZneeQeDPgp8X1iVJyHH7UuhZDdNeqGoduZsPrAhyMakqlc60-  
iJjL5HqyYGiT1FG8EXAMPLE"
```

Output:

```
{  
  "attachment": {
```

```
    "fileName": "troubleshoot-screenshot.png",
    "data": "base64-blob"
  }
}
```

For more information, see [Case management](#) in the *AWS Support User Guide*.

- For API details, see [DescribeAttachment](#) in *AWS CLI Command Reference*.

describe-cases

The following code example shows how to use `describe-cases`.

AWS CLI

To describe a case

The following `describe-cases` example returns information about the specified support case in your AWS account.

```
aws support describe-cases \
  --display-id "1234567890" \
  --after-time "2020-03-23T21:31:47.774Z" \
  --include-resolved-cases \
  --language "en" \
  --no-include-communications \
  --max-item 1
```

Output:

```
{
  "cases": [
    {
      "status": "resolved",
      "ccEmailAddresses": [],
      "timeCreated": "2020-03-23T21:31:47.774Z",
      "caseId": "case-12345678910-2013-c4c1d2bf33c5cf47",
      "severityCode": "low",
      "language": "en",
      "categoryCode": "using-aws",
      "serviceCode": "general-info",
      "submittedBy": "myemail@example.com",
      "displayId": "1234567890",
    }
  ]
}
```

```

      "subject": "Question about my account"
    }
  ]
}

```

For more information, see [Case management](#) in the *AWS Support User Guide*.

- For API details, see [DescribeCases](#) in *AWS CLI Command Reference*.

describe-communications

The following code example shows how to use describe-communications.

AWS CLI

To describe the latest communication for a case

The following describe-communications example returns the latest communication for the specified support case in your AWS account.

```

aws support describe-communications \
  --case-id "case-12345678910-2013-c4c1d2bf33c5cf47" \
  --after-time "2020-03-23T21:31:47.774Z" \
  --max-item 1

```

Output:

```

{
  "communications": [
    {
      "body": "I want to learn more about an AWS service.",
      "attachmentSet": [],
      "caseId": "case-12345678910-2013-c4c1d2bf33c5cf47",
      "timeCreated": "2020-05-12T23:12:35.000Z",
      "submittedBy": "Amazon Web Services"
    }
  ],
  "NextToken": "eyJmZXRhZG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQEXAMPLE=="
}

```

For more information, see [Case management](#) in the *AWS Support User Guide*.

- For API details, see [DescribeCommunications](#) in *AWS CLI Command Reference*.

describe-services

The following code example shows how to use describe-services.

AWS CLI

To list AWS services and service categories

The following describe-services example lists the available service categories for requesting general information.

```
aws support describe-services \  
  --service-code-list "general-info"
```

Output:

```
{  
  "services": [  
    {  
      "code": "general-info",  
      "name": "General Info and Getting Started",  
      "categories": [  
        {  
          "code": "charges",  
          "name": "How Will I Be Charged?"  
        },  
        {  
          "code": "gdpr-queries",  
          "name": "Data Privacy Query"  
        },  
        {  
          "code": "reserved-instances",  
          "name": "Reserved Instances"  
        },  
        {  
          "code": "resource",  
          "name": "Where is my Resource?"  
        },  
        {  
          "code": "using-aws",  
          "name": "Using AWS & Services"  
        },  
        {
```

```
        "code": "free-tier",
        "name": "Free Tier"
    },
    {
        "code": "security-and-compliance",
        "name": "Security & Compliance"
    },
    {
        "code": "account-structure",
        "name": "Account Structure"
    }
]
}
```

For more information, see [Case management](#) in the *AWS Support User Guide*.

- For API details, see [DescribeServices](#) in *AWS CLI Command Reference*.

describe-severity-levels

The following code example shows how to use `describe-severity-levels`.

AWS CLI

To list the available severity levels

The following `describe-severity-levels` example lists the available severity levels for a support case.

```
aws support describe-severity-levels
```

Output:

```
{
  "severityLevels": [
    {
      "code": "low",
      "name": "Low"
    },
    {
      "code": "normal",
```

```
        "name": "Normal"
    },
    {
        "code": "high",
        "name": "High"
    },
    {
        "code": "urgent",
        "name": "Urgent"
    },
    {
        "code": "critical",
        "name": "Critical"
    }
]
}
```

For more information, see [Choosing a severity](#) in the *AWS Support User Guide*.

- For API details, see [DescribeSeverityLevels](#) in *AWS CLI Command Reference*.

describe-trusted-advisor-check-refresh-statuses

The following code example shows how to use `describe-trusted-advisor-check-refresh-statuses`.

AWS CLI

To list the refresh statuses of AWS Trusted Advisor checks

The following `describe-trusted-advisor-check-refresh-statuses` example lists the refresh statuses for two Trusted Advisor checks: Amazon S3 Bucket Permissions and IAM Use.

```
aws support describe-trusted-advisor-check-refresh-statuses \
  --check-id "Pfx0RwqBli" "zXCkfM1nI3"
```

Output:

```
{
  "statuses": [
    {
      "checkId": "Pfx0RwqBli",
```

```
    "status": "none",
    "millisUntilNextRefreshable": 0
  },
  {
    "checkId": "zXCkfM1nI3",
    "status": "none",
    "millisUntilNextRefreshable": 0
  }
]
```

For more information, see [AWS Trusted Advisor](#) in the *AWS Support User Guide*.

- For API details, see [DescribeTrustedAdvisorCheckRefreshStatuses](#) in *AWS CLI Command Reference*.

describe-trusted-advisor-check-result

The following code example shows how to use `describe-trusted-advisor-check-result`.

AWS CLI

To list the results of an AWS Trusted Advisor check

The following `describe-trusted-advisor-check-result` example lists the results of the IAM Use check.

```
aws support describe-trusted-advisor-check-result \
  --check-id "zXCkfM1nI3"
```

Output:

```
{
  "result": {
    "checkId": "zXCkfM1nI3",
    "timestamp": "2020-05-13T21:38:05Z",
    "status": "ok",
    "resourcesSummary": {
      "resourcesProcessed": 1,
      "resourcesFlagged": 0,
      "resourcesIgnored": 0,
      "resourcesSuppressed": 0
    }
  },
```



```
    "categorySpecificSummary": {
      "costOptimizing": {
        "estimatedMonthlySavings": 0.0,
        "estimatedPercentMonthlySavings": 0.0
      }
    },
    "flaggedResources": [
      {
        "status": "ok",
        "resourceId": "47DEQpj8HBSa-_TImW-5JCeuQeRkm5NMpJWZEXAMPLE",
        "isSuppressed": false
      }
    ]
  }
}
```

For more information, see [AWS Trusted Advisor](#) in the *AWS Support User Guide*.

- For API details, see [DescribeTrustedAdvisorCheckResult](#) in *AWS CLI Command Reference*.

describe-trusted-advisor-check-summaries

The following code example shows how to use `describe-trusted-advisor-check-summaries`.

AWS CLI

To list the summaries of AWS Trusted Advisor checks

The following `describe-trusted-advisor-check-summaries` example lists the results for two Trusted Advisor checks: Amazon S3 Bucket Permissions and IAM Use.

```
aws support describe-trusted-advisor-check-summaries \
  --check-ids "Pfx0RwqBli" "zXCkfM1nI3"
```

Output:

```
{
  "summaries": [
    {
      "checkId": "Pfx0RwqBli",
      "timestamp": "2020-05-13T21:38:12Z",
```

```

    "status": "ok",
    "hasFlaggedResources": true,
    "resourcesSummary": {
      "resourcesProcessed": 44,
      "resourcesFlagged": 0,
      "resourcesIgnored": 0,
      "resourcesSuppressed": 0
    },
    "categorySpecificSummary": {
      "costOptimizing": {
        "estimatedMonthlySavings": 0.0,
        "estimatedPercentMonthlySavings": 0.0
      }
    }
  },
  {
    "checkId": "zXCkfM1nI3",
    "timestamp": "2020-05-13T21:38:05Z",
    "status": "ok",
    "hasFlaggedResources": true,
    "resourcesSummary": {
      "resourcesProcessed": 1,
      "resourcesFlagged": 0,
      "resourcesIgnored": 0,
      "resourcesSuppressed": 0
    },
    "categorySpecificSummary": {
      "costOptimizing": {
        "estimatedMonthlySavings": 0.0,
        "estimatedPercentMonthlySavings": 0.0
      }
    }
  }
]
}

```

For more information, see [AWS Trusted Advisor](#) in the *AWS Support User Guide*.

- For API details, see [DescribeTrustedAdvisorCheckSummaries](#) in *AWS CLI Command Reference*.

describe-trusted-advisor-checks

The following code example shows how to use `describe-trusted-advisor-checks`.

AWS CLI

To list the available AWS Trusted Advisor checks

The following `describe-trusted-advisor-checks` example lists the available Trusted Advisor checks in your AWS account. This information includes the check name, ID, description, category, and metadata. Note that the output is shortened for readability.

```
aws support describe-trusted-advisor-checks \  
  --language "en"
```

Output:

```
{  
  "checks": [  
    {  
      "id": "zXCkFM1nI3",  
      "name": "IAM Use",  
      "description": "Checks for your use of AWS Identity and Access  
Management (IAM). You can use IAM to create users, groups, and roles in AWS, and  
you can use permissions to control access to AWS resources. \n<br>\n<br>\n<b>Alert  
Criteria</b><br>\nYellow: No IAM users have been created for this account.\n<br>  
<br>\n<b>Recommended Action</b><br>\nCreate one or more IAM users and groups in  
your account. You can then create additional users whose permissions are limited  
to perform specific tasks in your AWS environment. For more information, see <a  
href=\"https://docs.aws.amazon.com/IAM/latest/UserGuide/IAMGettingStarted.html\"  
target=\"_blank\">Getting Started</a>. \n<br><br>\n<b>Additional Resources</b><br>  
<a href=\"https://docs.aws.amazon.com/IAM/latest/UserGuide/IAM_Introduction.html\"  
target=\"_blank\">What Is IAM?</a>",  
      "category": "security",  
      "metadata": []  
    }  
  ]  
}
```

For more information, see [AWS Trusted Advisor](#) in the *AWS Support User Guide*.

- For API details, see [DescribeTrustedAdvisorChecks](#) in *AWS CLI Command Reference*.

refresh-trusted-advisor-check

The following code example shows how to use `refresh-trusted-advisor-check`.

AWS CLI

To refresh an AWS Trusted Advisor check

The following `refresh-trusted-advisor-check` example refreshes the Amazon S3 Bucket Permissions Trusted Advisor check in your AWS account.

```
aws support refresh-trusted-advisor-check \  
  --check-id "Pfx0RwqBli"
```

Output:

```
{  
  "status": {  
    "checkId": "Pfx0RwqBli",  
    "status": "enqueued",  
    "millisUntilNextRefreshable": 3599992  
  }  
}
```

For more information, see [AWS Trusted Advisor](#) in the *AWS Support User Guide*.

- For API details, see [RefreshTrustedAdvisorCheck](#) in *AWS CLI Command Reference*.

resolve-case

The following code example shows how to use `resolve-case`.

AWS CLI

To resolve a support case

The following `resolve-case` example resolves a support case in your AWS account.

```
aws support resolve-case \  
  --case-id "case-12345678910-2013-c4c1d2bf33c5cf47"
```

Output:

```
{  
  "finalCaseStatus": "resolved",
```

```
"initialCaseStatus": "work-in-progress"  
}
```

For more information, see [Case management](#) in the *AWS Support User Guide*.

- For API details, see [ResolveCase](#) in *AWS CLI Command Reference*.

Amazon SWF examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon SWF.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

count-closed-workflow-executions

The following code example shows how to use `count-closed-workflow-executions`.

AWS CLI

Counting Closed Workflow Executions

You can use `swf count-closed-workflow-executions` to retrieve the number of closed workflow executions for a given domain. You can specify filters to count specific classes of executions.

The `--domain` and *either* `--close-time-filter` or `--start-time-filter` arguments are required. All other arguments are optional.

```
aws swf count-closed-workflow-executions \  
  --domain DataFrobtzz \  
  --close-time-filter "{ \"latestDate\" : 1377129600, \"oldestDate\" :  
1370044800 }"
```

Output:

```
{  
  "count": 2,  
  "truncated": false  
}
```

If "truncated" is true, then "count" represents the maximum number that can be returned by Amazon SWF. Any further results are truncated.

To reduce the number of results returned, you can:

modify the `--close-time-filter` or `--start-time-filter` values to narrow the time range that is searched. Each of these is mutually exclusive: You can specify *only one of these* in a request. Use the `--close-status-filter`, `--execution-filter`, `--tag-filter` or `--type-filter` arguments to further filter the results. However, these arguments are also mutually exclusive.

See Also [CountClosedWorkflowExecutions](#) in the *Amazon Simple Workflow Service API Reference*

- For API details, see [CountClosedWorkflowExecutions](#) in *AWS CLI Command Reference*.

count-open-workflow-executions

The following code example shows how to use `count-open-workflow-executions`.

AWS CLI

Counting Open Workflow Executions

You can use `swf count-open-workflow-executions` to retrieve the number of open workflow executions for a given domain. You can specify filters to count specific classes of executions.

The `--domain` and `--start-time-filter` arguments are required. All other arguments are optional.

```
aws swf count-open-workflow-executions \  
  --domain DataFrobtzz \  
  --start-time-filter "{ \"latestDate\" : 1377129600, \"oldestDate\" :  
1370044800 }"
```

Output:

```
{  
  "count": 4,  
  "truncated": false  
}
```

If "truncated" is true, then "count" represents the maximum number that can be returned by Amazon SWF. Any further results are truncated.

To reduce the number of results returned, you can:

modify the `--start-time-filter` values to narrow the time range that is searched. Use the `--close-status-filter`, `--execution-filter`, `--tag-filter` or `--type-filter` arguments to further filter the results. Each of these is mutually exclusive: You can specify *only one of these* in a request.

For more information, see `CountOpenWorkflowExecutions` in the *Amazon Simple Workflow Service API Reference*

- For API details, see [CountOpenWorkflowExecutions](#) in *AWS CLI Command Reference*.

deprecate-domain

The following code example shows how to use `deprecate-domain`.

AWS CLI

Deprecating a Domain

To deprecate a domain (you can still see it, but cannot create new workflow executions or register types on it), use `swf deprecate-domain`. It has a sole required parameter, `--name`, which takes the name of the domain to deprecate.

```
aws swf deprecate-domain \  
  --name DataFrobtzz
```

```
--name MyNeatNewDomain ""
```

As with `register-domain`, no output is returned. If you use `list-domains` to view the registered domains, however, you will see that the domain has been deprecated and no longer appears in the returned data.

```
aws swf list-domains \  
  --registration-status REGISTERED  
  {  
    "domainInfos": [  
      {  
        "status": "REGISTERED",  
        "name": "DataFrobotz"  
      },  
      {  
        "status": "REGISTERED",  
        "name": "erontest"  
      }  
    ]  
  }
```

If you use `--registration-status DEPRECATED` with `list-domains`, you will see your deprecated domain.

```
aws swf list-domains \  
  --registration-status DEPRECATED  
  {  
    "domainInfos": [  
      {  
        "status": "DEPRECATED",  
        "name": "MyNeatNewDomain"  
      }  
    ]  
  }
```

You can still use `describe-domain` to get information about a deprecated domain.

```
aws swf describe-domain \  
  --name MyNeatNewDomain  
  {  
    "domainInfo": {
```



```
        "status": "DEPRECATED",
        "name": "MyNeatNewDomain"
    },
    "configuration": {
        "workflowExecutionRetentionPeriodInDays": "0"
    }
}
```

See Also [DeprecateDomain](#) in the *Amazon Simple Workflow Service API Reference*

- For API details, see [DeprecateDomain](#) in *AWS CLI Command Reference*.

describe-domain

The following code example shows how to use `describe-domain`.

AWS CLI

Getting Information About a Domain

To get detailed information about a particular domain, use the `swf describe-domain` command. There is one required parameter: `--name`, which takes the name of the domain you want information about.

```
aws swf describe-domain \
  --name DataFrobotz
  {
    "domainInfo": {
      "status": "REGISTERED",
      "name": "DataFrobotz"
    },
    "configuration": {
      "workflowExecutionRetentionPeriodInDays": "1"
    }
  }
```

You can also use `describe-domain` to get information about deprecated domains.

```
aws swf describe-domain \
  --name MyNeatNewDomain
  {
    "domainInfo": {
```

```
        "status": "DEPRECATED",
        "name": "MyNeatNewDomain"
    },
    "configuration": {
        "workflowExecutionRetentionPeriodInDays": "0"
    }
}
```

See Also [DescribeDomain](#) in the *Amazon Simple Workflow Service API Reference*

- For API details, see [DescribeDomain](#) in *AWS CLI Command Reference*.

list-activity-types

The following code example shows how to use `list-activity-types`.

AWS CLI

Listing Activity Types

To get a list of the activity types for a domain, use `swf list-activity-types`. The `--domain` and `--registration-status` arguments are required.

```
aws swf list-activity-types \
  --domain DataFrobtzz \
  --registration-status REGISTERED
```

Output:

```
{
  "typeInfos": [
    {
      "status": "REGISTERED",
      "creationDate": 1371454150.451,
      "activityType": {
        "version": "1",
        "name": "confirm-user-email"
      },
      "description": "subscribe confirm-user-email activity"
    },
    {
      "status": "REGISTERED",
```

```

    "creationDate": 1371454150.709,
    "activityType": {
      "version": "1",
      "name": "confirm-user-phone"
    },
    "description": "subscribe confirm-user-phone activity"
  },
  {
    "status": "REGISTERED",
    "creationDate": 1371454149.871,
    "activityType": {
      "version": "1",
      "name": "get-subscription-info"
    },
    "description": "subscribe get-subscription-info activity"
  },
  {
    "status": "REGISTERED",
    "creationDate": 1371454150.909,
    "activityType": {
      "version": "1",
      "name": "send-subscription-success"
    },
    "description": "subscribe send-subscription-success activity"
  },
  {
    "status": "REGISTERED",
    "creationDate": 1371454150.085,
    "activityType": {
      "version": "1",
      "name": "subscribe-user-sns"
    },
    "description": "subscribe subscribe-user-sns activity"
  }
]
}

```

You can use the `--name` argument to select only activity types with a particular name:

```

aws swf list-activity-types \
  --domain DataFrobtzz \
  --registration-status REGISTERED \
  --name "send-subscription-success"

```

Output:

```
{
  "typeInfos": [
    {
      "status": "REGISTERED",
      "creationDate": 1371454150.909,
      "activityType": {
        "version": "1",
        "name": "send-subscription-success"
      },
      "description": "subscribe send-subscription-success activity"
    }
  ]
}
```

To retrieve results in pages, you can set the `--maximum-page-size` argument. If more results are returned than will fit in a page of results, a `"nextPageToken"` will be returned in the result set:

```
aws swf list-activity-types \
  --domain DataFrobtzz \
  --registration-status REGISTERED \
  --maximum-page-size 2
```

Output:

```
{
  "nextPageToken": "AAAAKgAAAAEAAAAAAAAAAAAA1Gp1BelJq
+PmHvAnDxJYbup8+0R4LVtbXLDL17QNY7C30pHo9Ssz06D/GuFz10yC73umBQ1t0PJ/gC/
aYpzDMqUIWIA1T9W0s2DryyZX40C/6Lhk9/
o5kdsuWMSBkHhgaZjgwp3WJINIFJFdaSMxY2vYAX7AtRtpcqJuBDDRE9RaRqDGyqIYUMltarki qpSY1ZVveBasBvlyvU
WGAaqehiDz7/JzLT/wWNNUM0d+Nhe",
  "typeInfos": [
    {
      "status": "REGISTERED",
      "creationDate": 1371454150.451,
      "activityType": {
        "version": "1",
        "name": "confirm-user-email"
      },
      "description": "subscribe confirm-user-email activity"
    }
  ]
}
```

```

    },
    {
      "status": "REGISTERED",
      "creationDate": 1371454150.709,
      "activityType": {
        "version": "1",
        "name": "confirm-user-phone"
      },
      "description": "subscribe confirm-user-phone activity"
    }
  ]
}

```

You can pass the `nextPageToken` value to the next call to `list-activity-types` in the `--next-page-token` argument, retrieving the next page of results:

```

aws swf list-activity-types \
  --domain DataFrobtzz \
  --registration-status REGISTERED \
  --maximum-page-size 2 \
  --next-page-token "AAAAKgAAAAEAAAAAAAAAAAAA1Gp1Be1Jq
+PmHvAnDxJYbup8+0R4LVtbXLD17QNY7C30pHo9Ssz06D/GuFz10yC73umBQ1t0PJ/gC/
aYpzDMqUIWIA1T9W0s2DryyZX40C/6Lhk9/
o5kdsuWMSBkHhgaZjgwp3WJINIFJFdaSMxY2vYAX7AtRtpcqJuBDDRE9RaRqDGyqIYUMltarki qpSY1ZVveBasBv1vyU
WGAaqehiDz7/JzLT/wWNNUM0d+Nhe"

```

Output:

```

{
  "nextPageToken": "AAAAKgAAAAEAAAAAAAAAAAAAw+7LZ4GRZPzTqBHsp2wBxWB8m1sgLCclgCuq3J+h/
m3+v0fFqtkcjLwV5cc40jNAzTCuq/
XcylPumGwkjbajtqpZpbq0cVNfjFxFgoi0LB201bvV0krbUISBvlpFPmSwpDSZJsxg5UxCcweteS1Fn1PNSZ/
MoinBZo80TkjMuzcsTuK0zH9wCaR8ITcALJ3SaqHU3pyIRS5hPmFA30LIc8zaAepjlaujo6hntNSCruB4"
  "typeInfos": [
    {
      "status": "REGISTERED",
      "creationDate": 1371454149.871,
      "activityType": {
        "version": "1",
        "name": "get-subscription-info"
      },
      "description": "subscribe get-subscription-info activity"
    },
  ],
}

```

```

    {
      "status": "REGISTERED",
      "creationDate": 1371454150.909,
      "activityType": {
        "version": "1",
        "name": "send-subscription-success"
      },
      "description": "subscribe send-subscription-success activity"
    }
  ]
}

```

If there are still more results to return, "nextPageToken" will be returned with the results. When there are no more pages of results to return, "nextPageToken" will *not* be returned in the result set.

You can use the `--reverse-order` argument to reverse the order of the returned results. This also affects paged results.

```

aws swf list-activity-types \
  --domain DataFrobtzz \
  --registration-status REGISTERED \
  --maximum-page-size 2 \
  --reverse-order

```

Output:

```

{
  "nextPageToken": "AAAAKgAAAAEAAAAAAAAAAwXcpu5ePSyQkrC
+8WMbmSrenuZC2ZkIXQYBPB/b9xIOVkj+bMEFhGj0KmmJ4rF7iddhjf7UMYCsfGkEn7mk
+yMCgVc1JxDWmB0EH46bhcmclmYNQihMDmUwocpr7To6/R7CLu0St1gkFayx0idJXErQW0zdNfQaIWAnF/
cwioBbXlkz1fQzmDeU3M5oYGMPQIrUqkPq7pMEW0q0lK5eDN97NzFYdZZ/r1cLDWPZhUjY",
  "typeInfos": [
    {
      "status": "REGISTERED",
      "creationDate": 1371454150.085,
      "activityType": {
        "version": "1",
        "name": "subscribe-user-sns"
      },
      "description": "subscribe subscribe-user-sns activity"
    },
  ],
}

```

```
{
  "status": "REGISTERED",
  "creationDate": 1371454150.909,
  "activityType": {
    "version": "1",
    "name": "send-subscription-success"
  },
  "description": "subscribe send-subscription-success activity"
}
]
```

See Also [ListActivityTypes](#) in the *Amazon Simple Workflow Service API Reference*

- For API details, see [ListActivityTypes](#) in *AWS CLI Command Reference*.

list-domains

The following code example shows how to use `list-domains`.

AWS CLI

Example 1: To list your registered domains

The following `list-domains` command example lists the REGISTERED SWF domains that you have registered for your account.

```
aws swf list-domains \
  --registration-status REGISTERED
```

Output:

```
{
  "domainInfos": [
    {
      "status": "REGISTERED",
      "name": "DataFrobotz"
    },
    {
      "status": "REGISTERED",
      "name": "erontest"
    }
  ]
}
```

```
]
}
```

For more information, see [ListDomains](#) in the *Amazon Simple Workflow Service API Reference*

Example 2: To list your deprecated domains

The following `list-domains` command example lists the DEPRECATED SWF domains that you have registered for your account. Deprecated domains are domains that can not register new workflows or activities, but that can still be queried.

```
aws swf list-domains \
  --registration-status DEPRECATED
```

Output:

```
{
  "domainInfos": [
    {
      "status": "DEPRECATED",
      "name": "MyNeatNewDomain"
    }
  ]
}
```

For more information, see [ListDomains](#) in the *Amazon Simple Workflow Service API Reference*

Example 3: To list the first page of registered domains

The following `list-domains` command example lists the first page REGISTERED SWF domains that you have registered for your account using the `--maximum-page-size` option.

```
aws swf list-domains \
  --registration-status REGISTERED \
  --maximum-page-size 1
```

Output:

```
{
  "domainInfos": [
```



```

    {
      "status": "REGISTERED",
      "name": "DataFrobotz"
    }
  ],
  "nextPageToken": "AAAAKgAAAAEAAAAAAAAAAAAA2QJKNtidVgd49TTeNwYcpD
+QKT2ynuEbibcQWe2QKrs1MGe63gpS0MgZGpcpoKttL40CXRFn98Xif557it
+wSZUsvUDtImjDLvguyuyyFdIZtvIxIKE0Pm3k2r40jAGaFsG0uVbrK1jv1a7wdU7FYH301kNCP8b7PBj9SBkUyGoiAg
}

```

For more information, see [ListDomains](#) in the *Amazon Simple Workflow Service API Reference*

Example 4: To list the specified single page of registered domains

The following `list-domains` command example lists the first page REGISTERED SWF domains that you have registered for your account using the `--maximum-page-size` option.

When you make the call again, this time supplying the value of `nextPageToken` in the `--next-page-token` argument, you'll get another page of results.

```

aws swf list-domains \
  --registration-status REGISTERED \
  --maximum-page-size 1 \
  --next-page-token "AAAAKgAAAAEAAAAAAAAAAAAA2QJKNtidVgd49TTeNwYcpD
+QKT2ynuEbibcQWe2QKrs1MGe63gpS0MgZGpcpoKttL40CXRFn98Xif557it
+wSZUsvUDtImjDLvguyuyyFdIZtvIxIKE0Pm3k2r40jAGaFsG0uVbrK1jv1a7wdU7FYH301kNCP8b7PBj9SBkUyGoiAg

```

Output:

```

{
  "domainInfos": [
    {
      "status": "REGISTERED",
      "name": "erontest"
    }
  ]
}

```

When there are no further pages of results to retrieve, `nextPageToken` will not be returned in the results.

For more information, see [ListDomains](#) in the *Amazon Simple Workflow Service API Reference*

- For API details, see [ListDomains](#) in *AWS CLI Command Reference*.

list-workflow-types

The following code example shows how to use `list-workflow-types`.

AWS CLI

Listing Workflow Types

To get a list of the workflow types for a domain, use `swf list-workflow-types`. The `--domain` and `--registration-status` arguments are required. Here's a simple example.

```
aws swf list-workflow-types \  
  --domain DataFrobtzz \  
  --registration-status REGISTERED
```

Output:

```
{  
  "typeInfos": [  
    {  
      "status": "REGISTERED",  
      "creationDate": 1371454149.598,  
      "description": "DataFrobtzz subscribe workflow",  
      "workflowType": {  
        "version": "v3",  
        "name": "subscribe"  
      }  
    }  
  ]  
}
```

As with `list-activity-types`, you can use the `--name` argument to select only workflow types with a particular name, and use the `--maximum-page-size` argument in coordination with `--next-page-token` to page results. To reverse the order in which results are returned, use `--reverse-order`.

See Also [ListWorkflowTypes](#) in the *Amazon Simple Workflow Service API Reference*

- For API details, see [ListWorkflowTypes](#) in *AWS CLI Command Reference*.

register-domain

The following code example shows how to use `register-domain`.

AWS CLI

Registering a Domain

You can use the AWS CLI to register new domains. Use the `swf register-domain` command. There are two required parameters, `--name`, which takes the domain name, and `--workflow-execution-retention-period-in-days`, which takes an integer to specify the number of days to retain workflow execution data on this domain, up to a maximum period of 90 days (for more information, see the SWF FAQ <https://aws.amazon.com/swf/faqs/#retain_limit>). Workflow execution data will not be retained after the specified number of days have passed.

```
aws swf register-domain \  
  --name MyNeatNewDomain \  
  --workflow-execution-retention-period-in-days 0  
  ""
```

When you register a domain, nothing is returned (""), but you can use `swf list-domains` or `swf describe-domain` to see the new domain.

```
aws swf list-domains \  
  --registration-status REGISTERED  
  {  
    "domainInfos": [  
      {  
        "status": "REGISTERED",  
        "name": "DataFrobotz"  
      },  
      {  
        "status": "REGISTERED",  
        "name": "MyNeatNewDomain"  
      },  
      {  
        "status": "REGISTERED",  
        "name": "erontest"  
      }  
    ]  
  }
```

Using `swf describe-domain`:

```
aws swf describe-domain --name MyNeatNewDomain
{
  "domainInfo": {
    "status": "REGISTERED",
    "name": "MyNeatNewDomain"
  },
  "configuration": {
    "workflowExecutionRetentionPeriodInDays": "0"
  }
}
```

See Also [RegisterDomain](#) in the *Amazon Simple Workflow Service API Reference*

- For API details, see [RegisterDomain](#) in *AWS CLI Command Reference*.

register-workflow-type

The following code example shows how to use `register-workflow-type`.

AWS CLI

Registering a Workflow Type

To register a Workflow type with the AWS CLI, use the `swf register-workflow-type` command.

```
aws swf register-workflow-type \
  --domain DataFrobtzz \
  --name "MySimpleWorkflow" \
  --workflow-version "v1"
```

If successful, the command produces no output.

On an error (for example, if you try to register the same workflow type twice, or specify a domain that doesn't exist) you will get a response in JSON.

```
{
  "message": "WorkflowType=[name=MySimpleWorkflow, version=v1]",
  "__type": "com.amazonaws.swf.base.model#TypeAlreadyExistsFault"
}
```

The `--domain`, `--name` and `--workflow-version` are required. You can also set the workflow description, timeouts, and child workflow policy.

For more information, see [RegisterWorkflowType](#) in the *Amazon Simple Workflow Service API Reference*

- For API details, see [RegisterWorkflowType](#) in *AWS CLI Command Reference*.

Systems Manager examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Systems Manager.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

add-tags-to-resource

The following code example shows how to use `add-tags-to-resource`.

AWS CLI

Example 1: To add tags to a maintenance window

The following `add-tags-to-resource` example adds a tag to the specified maintenance window.

```
aws ssm add-tags-to-resource \  
    --resource-type "MaintenanceWindow" \  
    --tags "tag-key=tag-value"
```

```
--resource-id "mw-03eb9db428EXAMPLE" \  
--tags "Key=Stack,Value=Production"
```

This command produces no output.

Example 2: To add tags to a parameter

The following `add-tags-to-resource` example adds two tags to to the specified parameter.

```
aws ssm add-tags-to-resource \  
  --resource-type "Parameter" \  
  --resource-id "My-Parameter" \  
  --tags '[{"Key":"Region","Value":"East"}, {"Key":"Environment",  
  "Value":"Production"}]'
```

This command produces no output.

Example 3: To add tags to an SSM document

The following `add-tags-to-resource` example adds a tag to to the specified document.

```
aws ssm add-tags-to-resource \  
  --resource-type "Document" \  
  --resource-id "My-Document" \  
  --tags "Key=Quarter,Value=Q322"
```

This command produces no output.

For more information, see [Tagging Systems Manager resources](#) in the *AWS Systems Manager User Guide*.

- For API details, see [AddTagsToResource](#) in *AWS CLI Command Reference*.

associate-ops-item-related-item

The following code example shows how to use `associate-ops-item-related-item`.

AWS CLI

To associate a related item

The following `associate-ops-item-related-item` example associates a related item to the OpsItem.

```
aws ssm associate-ops-item-related-item \  
  --ops-item-id "oi-649fExample" \  
  --association-type "RelatesTo" \  
  --resource-type "AWS::SSMIncidents::IncidentRecord" \  
  --resource-uri "arn:aws:ssm-incidents::111122223333:incident-record/Example-  
Response-Plan/c2bde883-f7d5-343a-b13a-bf5fe9ea689f"
```

Output:

```
{  
  "AssociationId": "61d7178d-a30d-4bc5-9b4e-a9e74EXAMPLE"  
}
```

For more information, see [Working with Incident Manager incidents in OpsCenter](#) in the *AWS Systems Manager User Guide*.

- For API details, see [AssociateOpsItemRelatedItem](#) in *AWS CLI Command Reference*.

cancel-command

The following code example shows how to use `cancel-command`.

AWS CLI

Example 1: To cancel a command for all instances

The following `cancel-command` example attempts to cancel the specified command that is already running for all instances.

```
aws ssm cancel-command \  
  --command-id "662add3d-5831-4a10-b64a-f2ff3EXAMPLE"
```

This command produces no output.

Example 2: To cancel a command for specific instances

The following `cancel-command` example attempts to cancel a command for the specified instance only.

```
aws ssm cancel-command \  
  --command-id "662add3d-5831-4a10-b64a-f2ff3EXAMPLE"
```

```
--instance-ids "i-02573cafcfEXAMPLE"
```

This command produces no output.

For more information, see [Tagging Systems Manager Parameters](#) in the *AWS Systems Manager User Guide*.

- For API details, see [CancelCommand](#) in *AWS CLI Command Reference*.

cancel-maintenance-window-execution

The following code example shows how to use `cancel-maintenance-window-execution`.

AWS CLI

To cancel a maintenance window execution

This `cancel-maintenance-window-execution` example stops the specified maintenance window execution that is already in progress.

```
aws ssm cancel-maintenance-window-execution \  
  --window-execution-id j218d5b5c-mw66-tk4d-r3g9-1d4d1EXAMPLE
```

Output:

```
{  
  "WindowExecutionId": "j218d5b5c-mw66-tk4d-r3g9-1d4d1EXAMPLE"  
}
```

For more information, see [Systems Manager Maintenance Windows Tutorials \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [CancelMaintenanceWindowExecution](#) in *AWS CLI Command Reference*.

create-activation

The following code example shows how to use `create-activation`.

AWS CLI

To create a managed instance activation

The following `create-activation` example creates a managed instance activation.

```
aws ssm create-activation \  
  --default-instance-name "HybridWebServers" \  
  --iam-role "HybridWebServersRole" \  
  --registration-limit 5
```

Output:

```
{  
  "ActivationId": "5743558d-563b-4457-8682-d16c3EXAMPLE",  
  "ActivationCode": "dRmgnYaFv567vEXAMPLE"  
}
```

For more information, see [Step 4: Create a Managed-Instance Activation for a Hybrid Environment](#) in the *AWS Systems Manager User Guide*.

- For API details, see [CreateActivation](#) in *AWS CLI Command Reference*.

create-association-batch

The following code example shows how to use `create-association-batch`.

AWS CLI

To create multiple associations

This example associates a configuration document with multiple instances. The output returns a list of successful and failed operations, if applicable.

Command:

```
aws ssm create-association-batch --entries "Name=AWS-  
UpdateSSMAgent,InstanceId=i-1234567890abcdef0" "Name=AWS-  
UpdateSSMAgent,InstanceId=i-9876543210abcdef0"
```

Output:

```
{  
  "Successful": [  
    {  
      "Name": "AWS-UpdateSSMAgent",
```

```
"InstanceId": "i-1234567890abcdef0",
"AssociationVersion": "1",
"Date": 1550504725.007,
"LastUpdateAssociationDate": 1550504725.007,
"Status": {
  "Date": 1550504725.007,
  "Name": "Associated",
  "Message": "Associated with AWS-UpdateSSMAgent"
},
"Overview": {
  "Status": "Pending",
  "DetailedStatus": "Creating"
},
"DocumentVersion": "$DEFAULT",
"AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
"Targets": [
  {
    "Key": "InstanceIds",
    "Values": [
      "i-1234567890abcdef0"
    ]
  }
]
},
{
  "Name": "AWS-UpdateSSMAgent",
  "InstanceId": "i-9876543210abcdef0",
  "AssociationVersion": "1",
  "Date": 1550504725.057,
  "LastUpdateAssociationDate": 1550504725.057,
  "Status": {
    "Date": 1550504725.057,
    "Name": "Associated",
    "Message": "Associated with AWS-UpdateSSMAgent"
  },
  "Overview": {
    "Status": "Pending",
    "DetailedStatus": "Creating"
  },
  "DocumentVersion": "$DEFAULT",
  "AssociationId": "9c9f7f20-5154-4fed-a83e-0123456789ab",
  "Targets": [
    {
      "Key": "InstanceIds",
```

```

        "Values": [
            "i-9876543210abcdef0"
        ]
    }
]
},
"Failed": []
}

```

- For API details, see [CreateAssociationBatch](#) in *AWS CLI Command Reference*.

create-association

The following code example shows how to use create-association.

AWS CLI

Example 1: To associate a document using instance IDs

This example associates a configuration document with an instance, using instance IDs.

```

aws ssm create-association \
  --instance-id "i-0cb2b964d3e14fd9f" \
  --name "AWS-UpdateSSMAgent"

```

Output:

```

{
  "AssociationDescription": {
    "Status": {
      "Date": 1487875500.33,
      "Message": "Associated with AWS-UpdateSSMAgent",
      "Name": "Associated"
    },
    "Name": "AWS-UpdateSSMAgent",
    "InstanceId": "i-0cb2b964d3e14fd9f",
    "Overview": {
      "Status": "Pending",
      "DetailedStatus": "Creating"
    },
    "AssociationId": "b7c3266e-a544-44db-877e-b20d3a108189",
  }
}

```

```

    "DocumentVersion": "$DEFAULT",
    "LastUpdateAssociationDate": 1487875500.33,
    "Date": 1487875500.33,
    "Targets": [
      {
        "Values": [
          "i-0cb2b964d3e14fd9f"
        ],
        "Key": "InstanceIds"
      }
    ]
  }
}

```

For more information, see [CreateAssociation](#) in the *AWS Systems Manager API Reference*.

Example 2: To associate a document using targets

This example associates a configuration document with an instance, using targets.

```

aws ssm create-association \
  --name "AWS-UpdateSSMAgent" \
  --targets "Key=instanceids,Values=i-0cb2b964d3e14fd9f"

```

Output:

```

{
  "AssociationDescription": {
    "Status": {
      "Date": 1487875500.33,
      "Message": "Associated with AWS-UpdateSSMAgent",
      "Name": "Associated"
    },
    "Name": "AWS-UpdateSSMAgent",
    "InstanceId": "i-0cb2b964d3e14fd9f",
    "Overview": {
      "Status": "Pending",
      "DetailedStatus": "Creating"
    },
    "AssociationId": "b7c3266e-a544-44db-877e-b20d3a108189",
    "DocumentVersion": "$DEFAULT",
    "LastUpdateAssociationDate": 1487875500.33,

```

```

    "Date": 1487875500.33,
    "Targets": [
      {
        "Values": [
          "i-0cb2b964d3e14fd9f"
        ],
        "Key": "InstanceIds"
      }
    ]
  }
}

```

For more information, see [CreateAssociation](#) in the *AWS Systems Manager API Reference*.

Example 3: To create an association that runs only once

This example creates a new association that only runs once on the specified date and time. Associations created with a date in the past or present (by the time it is processed the date is in the past) run immediately.

```

aws ssm create-association \
  --name "AWS-UpdateSSMAgent" \
  --targets "Key=instanceids,Values=i-0cb2b964d3e14fd9f" \
  --schedule-expression "at(2020-05-14T15:55:00)" \
  --apply-only-at-cron-interval

```

Output:

```

{
  "AssociationDescription": {
    "Status": {
      "Date": 1487875500.33,
      "Message": "Associated with AWS-UpdateSSMAgent",
      "Name": "Associated"
    },
    "Name": "AWS-UpdateSSMAgent",
    "InstanceId": "i-0cb2b964d3e14fd9f",
    "Overview": {
      "Status": "Pending",
      "DetailedStatus": "Creating"
    },
    "AssociationId": "b7c3266e-a544-44db-877e-b20d3a108189",
  }
}

```

```
"DocumentVersion": "$DEFAULT",
"LastUpdateAssociationDate": 1487875500.33,
"Date": 1487875500.33,
"Targets": [
  {
    "Values": [
      "i-0cb2b964d3e14fd9f"
    ],
    "Key": "InstanceIds"
  }
]
```

For more information, see [CreateAssociation](#) in the *AWS Systems Manager API Reference* or [Reference: Cron and rate expressions for Systems Manager](#) in the *AWS Systems Manager User Guide*.

- For API details, see [CreateAssociation](#) in *AWS CLI Command Reference*.

create-document

The following code example shows how to use create-document.

AWS CLI

To create a document

The following create-document example creates a Systems Manager document.

```
aws ssm create-document \
  --content file://exampleDocument.yml \
  --name "Example" \
  --document-type "Automation" \
  --document-format YAML
```

Output:

```
{
  "DocumentDescription": {
    "Hash": "fc2410281f40779e694a8b95975d0f9f316da8a153daa94e3d9921102EXAMPLE",
    "HashType": "Sha256",
```

```

    "Name": "Example",
    "Owner": "29884EXAMPLE",
    "CreateDate": 1583256349.452,
    "Status": "Creating",
    "DocumentVersion": "1",
    "Description": "Document Example",
    "Parameters": [
      {
        "Name": "AutomationAssumeRole",
        "Type": "String",
        "Description": "(Required) The ARN of the role that allows
Automation to perform the actions on your behalf. If no role is specified, Systems
Manager Automation uses your IAM permissions to execute this document.",
        "DefaultValue": ""
      },
      {
        "Name": "InstanceId",
        "Type": "String",
        "Description": "(Required) The ID of the Amazon EC2 instance.",
        "DefaultValue": ""
      }
    ],
    "PlatformTypes": [
      "Windows",
      "Linux"
    ],
    "DocumentType": "Automation",
    "SchemaVersion": "0.3",
    "LatestVersion": "1",
    "DefaultVersion": "1",
    "DocumentFormat": "YAML",
    "Tags": []
  }
}

```

For more information, see [Creating Systems Manager Documents](#) in the *AWS Systems Manager User Guide*.

- For API details, see [CreateDocument](#) in *AWS CLI Command Reference*.

create-maintenance-window

The following code example shows how to use create-maintenance-window.

AWS CLI

Example 1: To create a maintenance window

The following `create-maintenance-window` example creates a new maintenance window that every five minutes for up to two hours (as needed), prevents new tasks from starting within one hour of the end of the maintenance window execution, allows unassociated targets (instances that you haven't registered with the maintenance window), and indicates through the use of custom tags that its creator intends to use it in a tutorial.

```
aws ssm create-maintenance-window \  
  --name "My-Tutorial-Maintenance-Window" \  
  --schedule "rate(5 minutes)" \  
  --duration 2 --cutoff 1 \  
  --allow-unassociated-targets \  
  --tags "Key=Purpose,Value=Tutorial"
```

Output:

```
{  
  "WindowId": "mw-0c50858d01EXAMPLE"  
}
```

Example 2: To create a maintenance window that runs only once

The following `create-maintenance-window` example creates a new maintenance window that only runs one time on the specified date and time.

```
aws ssm create-maintenance-window \  
  --name My-One-Time-Maintenance-Window \  
  --schedule "at(2020-05-14T15:55:00)" \  
  --duration 5 \  
  --cutoff 2 \  
  --allow-unassociated-targets \  
  --tags "Key=Environment,Value=Production"
```

Output:

```
{  
  "WindowId": "mw-01234567890abcdef"
```



```
}
```

For more information, see [Maintenance Windows](#) in the *AWS Systems Manager User Guide*.

- For API details, see [CreateMaintenanceWindow](#) in *AWS CLI Command Reference*.

create-ops-item

The following code example shows how to use `create-ops-item`.

AWS CLI

To create an OpsItems

The following `create-ops-item` example uses the `/aws/resources` key in `OperationalData` to create an OpsItem with an Amazon DynamoDB related resource.

```
aws ssm create-ops-item \  
  --title "EC2 instance disk full" \  
  --description "Log clean up may have failed which caused the disk to be full" \  
  --priority 2 \  
  --source ec2 \  
  --operational-data '{"aws/resources":{"Value":["arn\  
\": \"arn:aws:dynamodb:us-west-2:12345678:table/OpsItems\  
\"]},"Type":"SearchableString"}' \  
  --notifications Arn="arn:aws:sns:us-west-2:12345678:TestUser"
```

Output:

```
{  
  "OpsItemId": "oi-1a2b3c4d5e6f"  
}
```

For more information, see [Creating OpsItems](#) in the *AWS Systems Manager User Guide*.

- For API details, see [CreateOpsItem](#) in *AWS CLI Command Reference*.

create-patch-baseline

The following code example shows how to use `create-patch-baseline`.

AWS CLI

Example 1: To create a patch baseline with auto-approval

The following `create-patch-baseline` example creates a patch baseline for Windows Server that approves patches for a production environment seven days after they are released by Microsoft.

```
aws ssm create-patch-baseline \  
  --name "Windows-Production-Baseline-AutoApproval" \  
  --operating-system "WINDOWS" \  
  --approval-rules  
  "PatchRules=[{PatchFilterGroup={PatchFilters=[{Key=MSRC_SEVERITY,Values=[Critical,Important  
{Key=CLASSIFICATION,Values=[SecurityUpdates,Updates,UpdateRollups,CriticalUpdates]}]},Approv  
  \  
  --description "Baseline containing all updates approved for Windows Server  
  production systems"
```

Output:

```
{  
  "BaselineId": "pb-045f10b4f3EXAMPLE"  
}
```

Example 2: To create a patch baseline with an approval cutoff date

The following `create-patch-baseline` example creates a patch baseline for Windows Server that approves all patches for a production environment that are released on or before July 7, 2020.

```
aws ssm create-patch-baseline \  
  --name "Windows-Production-Baseline-AutoApproval" \  
  --operating-system "WINDOWS" \  
  --approval-rules  
  "PatchRules=[{PatchFilterGroup={PatchFilters=[{Key=MSRC_SEVERITY,Values=[Critical,Important  
{Key=CLASSIFICATION,Values=[SecurityUpdates,Updates,UpdateRollups,CriticalUpdates]}]},Approv  
  \  
  --description "Baseline containing all updates approved for Windows Server  
  production systems"
```

Output:

```
{
  "BaselineId": "pb-045f10b4f3EXAMPLE"
}
```

Example 3: To create a patch baseline with approval rules stored in a JSON file

The following `create-patch-baseline` example creates a patch baseline for Amazon Linux 2017.09 that approves patches for a production environment seven days after they are released, specifies approval rules for the patch baseline, and specifies a custom repository for patches.

```
aws ssm create-patch-baseline \
  --cli-input-json file://my-amazon-linux-approval-rules-and-repo.json
```

Contents of `my-amazon-linux-approval-rules-and-repo.json`:

```
{
  "Name": "Amazon-Linux-2017.09-Production-Baseline",
  "Description": "My approval rules patch baseline for Amazon Linux 2017.09
instances",
  "OperatingSystem": "AMAZON_LINUX",
  "Tags": [
    {
      "Key": "Environment",
      "Value": "Production"
    }
  ],
  "ApprovalRules": {
    "PatchRules": [
      {
        "ApproveAfterDays": 7,
        "EnableNonSecurity": true,
        "PatchFilterGroup": {
          "PatchFilters": [
            {
              "Key": "SEVERITY",
              "Values": [
                "Important",
                "Critical"
              ]
            }
          ]
        }
      }
    ]
  },
}
```

```

        {
            "Key": "CLASSIFICATION",
            "Values": [
                "Security",
                "Bugfix"
            ]
        },
        {
            "Key": "PRODUCT",
            "Values": [
                "AmazonLinux2017.09"
            ]
        }
    ]
}
}
}
],
"Sources": [
    {
        "Name": "My-AL2017.09",
        "Products": [
            "AmazonLinux2017.09"
        ],
        "Configuration": "[amzn-main] \nname=amzn-main-Base
\nmirrorlist=http://repo./$awsregion./$awsdomain//$releasever/main/mirror.list //
\nmirrorlist_expire=300//\nmetadata_expire=300 \npriority=10 \nfailovermethod=priority
\nfastestmirror_enabled=0 \ngpgcheck=1 \ngpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-
KEY-amazon-ga \nenabled=1 \nretries=3 \ntimeout=5\nreport_instanceid=yes"
    }
]
}

```

Example 4: To create a patch baseline that specifies approved and rejected patches

The following `create-patch-baseline` example explicitly specifies patches to approve and reject as exception to the default approval rules.

```

aws ssm create-patch-baseline \
  --name "Amazon-Linux-2017.09-Alpha-Baseline" \
  --description "My custom approve/reject patch baseline for Amazon Linux 2017.09
instances" \
  --operating-system "AMAZON_LINUX" \

```

```
--approved-patches "CVE-2018-1234567,example-pkg-EE-2018*.amzn1.noarch" \  
--approved-patches-compliance-level "HIGH" \  
--approved-patches-enable-non-security \  
--tags "Key=Environment,Value=Alpha"
```

For more information, see [Create a Custom Patch Baseline](#) in the *AWS Systems Manager User Guide*.

- For API details, see [CreatePatchBaseline](#) in *AWS CLI Command Reference*.

create-resource-data-sync

The following code example shows how to use `create-resource-data-sync`.

AWS CLI

To create a resource data sync

This example creates a resource data sync. There is no output if the command succeeds.

Command:

```
aws ssm create-resource-data-sync --sync-name "ssm-resource-data-sync" --s3-  
destination "BucketName=ssm-bucket,Prefix=inventory,SyncFormat=JsonSerDe,Region=us-  
east-1"
```

- For API details, see [CreateResourceDataSync](#) in *AWS CLI Command Reference*.

delete-activation

The following code example shows how to use `delete-activation`.

AWS CLI

To delete a managed instance activation

The following `delete-activation` example deletes a managed instance activation.

```
aws ssm delete-activation \  
--activation-id "aa673477-d926-42c1-8757-1358cEXAMPLE"
```

This command produces no output.

For more information, see [Setting Up AWS Systems Manager for Hybrid Environments](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DeleteActivation](#) in *AWS CLI Command Reference*.

delete-association

The following code example shows how to use delete-association.

AWS CLI

Example 1: To delete an association using the association ID

The following delete-association example deletes the association for the specified association ID. There is no output if the command succeeds.

```
aws ssm delete-association \  
  --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

This command produces no output.

For more information, see [Editing and creating a new version of an association](#) in the *AWS Systems Manager User Guide*.

Example 2: To delete an association

The following delete-association example deletes the association between an instance and a document. There is no output if the command succeeds.

```
aws ssm delete-association \  
  --instance-id "i-1234567890abcdef0" \  
  --name "AWS-UpdateSSMAgent"
```

This command produces no output.

For more information, see [Working with associations in Systems Manager](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DeleteAssociation](#) in *AWS CLI Command Reference*.

delete-document

The following code example shows how to use delete-document.

AWS CLI

To delete a document

The following delete-document example deletes a Systems Manager document.

```
aws ssm delete-document \  
  --name "Example"
```

This command produces no output.

For more information, see [Creating Systems Manager Documents](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DeleteDocument](#) in *AWS CLI Command Reference*.

delete-inventory

The following code example shows how to use delete-inventory.

AWS CLI

To delete a custom inventory type

This example deletes a custom inventory schema.

Command:

```
aws ssm delete-inventory --type-name "Custom:RackInfo" --schema-delete-option  
  "DeleteSchema"
```

Output:

```
{  
  "DeletionId": "d72ac9e8-1f60-4d40-b1c6-bf8c78c68c4d",  
  "TypeName": "Custom:RackInfo",  
  "DeletionSummary": {  
    "TotalCount": 1,  
  }  
}
```

```
    "RemainingCount": 1,
    "SummaryItems": [
      {
        "Version": "1.0",
        "Count": 1,
        "RemainingCount": 1
      }
    ]
  }
}
```

To disable a custom inventory type

This example disables a custom inventory schema.

Command:

```
aws ssm delete-inventory --type-name "Custom:RackInfo" --schema-delete-option
"DisableSchema"
```

Output:

```
{
  "DeletionId": "6961492a-8163-44ec-aa1e-923364dd0850",
  "TypeName": "Custom:RackInformation",
  "DeletionSummary": {
    "TotalCount": 0,
    "RemainingCount": 0,
    "SummaryItems": []
  }
}
```

- For API details, see [DeleteInventory](#) in *AWS CLI Command Reference*.

delete-maintenance-window

The following code example shows how to use `delete-maintenance-window`.

AWS CLI

To delete a maintenance window

This `delete-maintenance-window` example removes the specified maintenance window.

```
aws ssm delete-maintenance-window \  
  --window-id "mw-1a2b3c4d5e6f7g8h9"
```

Output:

```
{  
  "WindowId": "mw-1a2b3c4d5e6f7g8h9"  
}
```

For more information, see [Delete a Maintenance Window \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DeleteMaintenanceWindow](#) in *AWS CLI Command Reference*.

delete-parameter

The following code example shows how to use `delete-parameter`.

AWS CLI

To delete a parameter

The following `delete-parameter` example deletes the specified single parameter.

```
aws ssm delete-parameter \  
  --name "MyParameter"
```

This command produces no output.

For more information, see [Working with Parameter Store](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DeleteParameter](#) in *AWS CLI Command Reference*.

delete-parameters

The following code example shows how to use `delete-parameters`.

AWS CLI

To delete a list of parameters

The following `delete-parameters` example deletes the specified parameters.

```
aws ssm delete-parameters \  
  --names "MyFirstParameter" "MySecondParameter" "MyInvalidParameterName"
```

Output:

```
{  
  "DeletedParameters": [  
    "MyFirstParameter",  
    "MySecondParameter"  
  ],  
  "InvalidParameters": [  
    "MyInvalidParameterName"  
  ]  
}
```

For more information, see [Working with Parameter Store](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DeleteParameters](#) in *AWS CLI Command Reference*.

delete-patch-baseline

The following code example shows how to use `delete-patch-baseline`.

AWS CLI

To delete a patch baseline

The following `delete-patch-baseline` example deletes the specified patch baseline.

```
aws ssm delete-patch-baseline \  
  --baseline-id "pb-045f10b4f382baeda"
```

Output:

```
{
  "BaselineId": "pb-045f10b4f382baeda"
}
```

For more information, see [Update or Delete a Patch Baseline \(Console\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DeletePatchBaseline](#) in *AWS CLI Command Reference*.

delete-resource-data-sync

The following code example shows how to use `delete-resource-data-sync`.

AWS CLI

To delete a resource data sync

This example deletes a resource data sync. There is no output if the command succeeds.

Command:

```
aws ssm delete-resource-data-sync --sync-name "ssm-resource-data-sync"
```

- For API details, see [DeleteResourceDataSync](#) in *AWS CLI Command Reference*.

deregister-managed-instance

The following code example shows how to use `deregister-managed-instance`.

AWS CLI

To deregister a managed instance

The following `deregister-managed-instance` example deregisters the specified managed instance.

```
aws ssm deregister-managed-instance
  --instance-id "mi-08ab247cdfEXAMPLE"
```

This command produces no output.

For more information, see [Deregistering Managed Instances in a Hybrid Environment](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DeregisterManagedInstance](#) in *AWS CLI Command Reference*.

deregister-patch-baseline-for-patch-group

The following code example shows how to use `deregister-patch-baseline-for-patch-group`.

AWS CLI

To deregister a patch group from a patch baseline

The following `deregister-patch-baseline-for-patch-group` example deregisters the specified patch group from the specified patch baseline.

```
aws ssm deregister-patch-baseline-for-patch-group \  
  --patch-group "Production" \  
  --baseline-id "pb-0ca44a362fEXAMPLE"
```

Output:

```
{  
  "PatchGroup": "Production",  
  "BaselineId": "pb-0ca44a362fEXAMPLE"  
}
```

For more information, see [Add a Patch Group to a Patch Baseline](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DeregisterPatchBaselineForPatchGroup](#) in *AWS CLI Command Reference*.

deregister-target-from-maintenance-window

The following code example shows how to use `deregister-target-from-maintenance-window`.

AWS CLI

To remove a target from a maintenance window

The following `deregister-target-from-maintenance-window` example removes the specified target from the specified maintenance window.

```
aws ssm deregister-target-from-maintenance-window \  
  --window-id "mw-ab12cd34ef56gh78" \  
  --window-target-id "1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
```

Output:

```
{  
  "WindowId": "mw-ab12cd34ef56gh78",  
  "WindowTargetId": "1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"  
}
```

For more information, see [Update a Maintenance Window \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DeregisterTargetFromMaintenanceWindow](#) in *AWS CLI Command Reference*.

deregister-task-from-maintenance-window

The following code example shows how to use `deregister-task-from-maintenance-window`.

AWS CLI

To remove a task from a maintenance window

The following `deregister-task-from-maintenance-window` example removes the specified task from the specified maintenance window.

```
aws ssm deregister-task-from-maintenance-window \  
  --window-id "mw-ab12cd34ef56gh78" \  
  --window-task-id "1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d5e6c"
```

Output:

```
{  
  "WindowTaskId": "1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d5e6c",  
  "WindowId": "mw-ab12cd34ef56gh78"  
}
```

For more information, see [Systems Manager Maintenance Windows Tutorials \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DeregisterTaskFromMaintenanceWindow](#) in *AWS CLI Command Reference*.

describe-activations

The following code example shows how to use describe-activations.

AWS CLI

To describe activations

The following describe-activations example lists details about the activations in your AWS account.

```
aws ssm describe-activations
```

Output:

```
{
  "ActivationList": [
    {
      "ActivationId": "5743558d-563b-4457-8682-d16c3EXAMPLE",
      "Description": "Example1",
      "IamRole": "HybridWebServersRole",
      "RegistrationLimit": 5,
      "RegistrationsCount": 5,
      "ExpirationDate": 1584316800.0,
      "Expired": false,
      "CreatedDate": 1581954699.792
    },
    {
      "ActivationId": "3ee0322b-f62d-40eb-b672-13ebfEXAMPLE",
      "Description": "Example2",
      "IamRole": "HybridDatabaseServersRole",
      "RegistrationLimit": 5,
      "RegistrationsCount": 5,
      "ExpirationDate": 1580515200.0,
      "Expired": true,
      "CreatedDate": 1578064132.002
    },
  ],
}
```

```
]
}
```

For more information, see [Step 4: Create a Managed-Instance Activation for a Hybrid Environment](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeActivations](#) in *AWS CLI Command Reference*.

describe-association-execution-targets

The following code example shows how to use `describe-association-execution-targets`.

AWS CLI

To get details of an association execution

The following `describe-association-execution-targets` example describes the specified association execution.

```
aws ssm describe-association-execution-targets \
  --association-id "8dfe3659-4309-493a-8755-0123456789ab" \
  --execution-id "7abb6378-a4a5-4f10-8312-0123456789ab"
```

Output:

```
{
  "AssociationExecutionTargets": [
    {
      "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
      "AssociationVersion": "1",
      "ExecutionId": "7abb6378-a4a5-4f10-8312-0123456789ab",
      "ResourceId": "i-1234567890abcdef0",
      "ResourceType": "ManagedInstance",
      "Status": "Success",
      "DetailedStatus": "Success",
      "LastExecutionDate": 1550505538.497,
      "OutputSource": {
        "OutputSourceId": "97fff367-fc5a-4299-aed8-0123456789ab",
        "OutputSourceType": "RunCommand"
      }
    }
  ]
}
```

```
}
```

For more information, see [Viewing association histories](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeAssociationExecutionTargets](#) in *AWS CLI Command Reference*.

describe-association-executions

The following code example shows how to use `describe-association-executions`.

AWS CLI

Example 1: To get details of all executions for an association

The following `describe-association-executions` example describes all executions of the specified association.

```
aws ssm describe-association-executions \  
  --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

Output:

```
{  
  "AssociationExecutions": [  
    {  
      "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",  
      "AssociationVersion": "1",  
      "ExecutionId": "474925ef-1249-45a2-b93d-0123456789ab",  
      "Status": "Success",  
      "DetailedStatus": "Success",  
      "CreatedTime": 1550505827.119,  
      "ResourceCountByStatus": "{Success=1}"  
    },  
    {  
      "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",  
      "AssociationVersion": "1",  
      "ExecutionId": "7abb6378-a4a5-4f10-8312-0123456789ab",  
      "Status": "Success",  
      "DetailedStatus": "Success",  
      "CreatedTime": 1550505536.843,  
      "ResourceCountByStatus": "{Success=1}"  
    },  
    ...  
  ]  
}
```



```
]
}
```

For more information, see [Viewing association histories](#) in the *AWS Systems Manager User Guide*.

Example 2: To get details of all executions for an association after a specific date and time

The following `describe-association-executions` example describes all executions of an association after the specified date and time.

```
aws ssm describe-association-executions \
  --association-id "8dfe3659-4309-493a-8755-0123456789ab" \
  --filters "Key=CreatedTime,Value=2019-02-18T16:00:00Z,Type=GREATER_THAN"
```

Output:

```
{
  "AssociationExecutions": [
    {
      "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
      "AssociationVersion": "1",
      "ExecutionId": "474925ef-1249-45a2-b93d-0123456789ab",
      "Status": "Success",
      "DetailedStatus": "Success",
      "CreatedTime": 1550505827.119,
      "ResourceCountByStatus": "{Success=1}"
    },
    {
      "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
      "AssociationVersion": "1",
      "ExecutionId": "7abb6378-a4a5-4f10-8312-0123456789ab",
      "Status": "Success",
      "DetailedStatus": "Success",
      "CreatedTime": 1550505536.843,
      "ResourceCountByStatus": "{Success=1}"
    },
    ...
  ]
}
```

For more information, see [Viewing association histories](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeAssociationExecutions](#) in *AWS CLI Command Reference*.

describe-association

The following code example shows how to use describe-association.

AWS CLI

Example 1: To get details of an association

The following describe-association example describes the association for the specified association ID.

```
aws ssm describe-association \  
  --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

Output:

```
{  
  "AssociationDescription": {  
    "Name": "AWS-GatherSoftwareInventory",  
    "AssociationVersion": "1",  
    "Date": 1534864780.995,  
    "LastUpdateAssociationDate": 1543235759.81,  
    "Overview": {  
      "Status": "Success",  
      "AssociationStatusAggregatedCount": {  
        "Success": 2  
      }  
    },  
    "DocumentVersion": "$DEFAULT",  
    "Parameters": {  
      "applications": [  
        "Enabled"  
      ],  
      "awsComponents": [  
        "Enabled"  
      ],  
      "customInventory": [  
        "Enabled"  
      ],  
      "files": [  
        ""  
      ],  
      "instanceDetailedInformation": [  
        ""  
      ]  
    }  
  }  
}
```

```

        "Enabled"
    ],
    "networkConfig": [
        "Enabled"
    ],
    "services": [
        "Enabled"
    ],
    "windowsRegistry": [
        ""
    ],
    "windowsRoles": [
        "Enabled"
    ],
    "windowsUpdates": [
        "Enabled"
    ]
  },
  "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
  "Targets": [
    {
      "Key": "InstanceIds",
      "Values": [
        "*"
      ]
    }
  ],
  "ScheduleExpression": "rate(24 hours)",
  "LastExecutionDate": 1550501886.0,
  "LastSuccessfulExecutionDate": 1550501886.0,
  "AssociationName": "Inventory-Association"
}
}

```

For more information, see [Editing and creating a new version of an association](#) in the *AWS Systems Manager User Guide*.

Example 2: To get details of an association for a specific instance and document

The following describe-association example describes the association between an instance and a document.

```
aws ssm describe-association \
```

```
--instance-id "i-1234567890abcdef0" \  
--name "AWS-UpdateSSMAgent"
```

Output:

```
{  
  "AssociationDescription": {  
    "Status": {  
      "Date": 1487876122.564,  
      "Message": "Associated with AWS-UpdateSSMAgent",  
      "Name": "Associated"  
    },  
    "Name": "AWS-UpdateSSMAgent",  
    "InstanceId": "i-1234567890abcdef0",  
    "Overview": {  
      "Status": "Pending",  
      "DetailedStatus": "Associated",  
      "AssociationStatusAggregatedCount": {  
        "Pending": 1  
      }  
    },  
    "AssociationId": "d8617c07-2079-4c18-9847-1234567890ab",  
    "DocumentVersion": "$DEFAULT",  
    "LastUpdateAssociationDate": 1487876122.564,  
    "Date": 1487876122.564,  
    "Targets": [  
      {  
        "Values": [  
          "i-1234567890abcdef0"  
        ],  
        "Key": "InstanceIds"  
      }  
    ]  
  }  
}
```

For more information, see [Editing and creating a new version of an association](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeAssociation](#) in *AWS CLI Command Reference*.

describe-automation-executions

The following code example shows how to use describe-automation-executions.

AWS CLI

To describe an automation execution

The following describe-automation-executions example displays details about an Automation execution.

```
aws ssm describe-automation-executions \
  --filters Key=ExecutionId,Values=73c8eef8-f4ee-4a05-820c-e354fEXAMPLE
```

Output:

```
{
  "AutomationExecutionMetadataList": [
    {
      "AutomationExecutionId": "73c8eef8-f4ee-4a05-820c-e354fEXAMPLE",
      "DocumentName": "AWS-StartEC2Instance",
      "DocumentVersion": "1",
      "AutomationExecutionStatus": "Success",
      "ExecutionStartTime": 1583737233.748,
      "ExecutionEndTime": 1583737234.719,
      "ExecutedBy": "arn:aws:sts::29884EXAMPLE:assumed-role/mw_service_role/
OrchestrationService",
      "LogFile": "",
      "Outputs": {},
      "Mode": "Auto",
      "Targets": [],
      "ResolvedTargets": {
        "ParameterValues": [],
        "Truncated": false
      },
      "AutomationType": "Local"
    }
  ]
}
```

For more information, see [Running a Simple Automation Workflow](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeAutomationExecutions](#) in *AWS CLI Command Reference*.

describe-automation-step-executions

The following code example shows how to use describe-automation-step-executions.

AWS CLI

Example 1: To describe all steps for an automation execution

The following describe-automation-step-executions example displays details about the steps of an Automation execution.

```
aws ssm describe-automation-step-executions \
  --automation-execution-id 73c8eef8-f4ee-4a05-820c-e354fEXAMPLE
```

Output:

```
{
  "StepExecutions": [
    {
      "StepName": "startInstances",
      "Action": "aws:changeInstanceState",
      "ExecutionStartTime": 1583737234.134,
      "ExecutionEndTime": 1583737234.672,
      "StepStatus": "Success",
      "Inputs": {
        "DesiredState": "\"running\"",
        "InstanceIds": "[\"i-0cb99161f6EXAMPLE\"]"
      },
      "Outputs": {
        "InstanceStates": [
          "running"
        ]
      },
      "StepExecutionId": "95e70479-cf20-4d80-8018-7e4e2EXAMPLE",
      "OverriddenParameters": {}
    }
  ]
}
```

Example 2: To describe a specific step for an automation execution

The following `describe-automation-step-executions` example displays details about a specific step of an Automation execution.

```
aws ssm describe-automation-step-executions \  
  --automation-execution-id 73c8eef8-f4ee-4a05-820c-e354fEXAMPLE \  
  --filters Key=StepExecutionId,Values=95e70479-cf20-4d80-8018-7e4e2EXAMPLE
```

For more information, see [Running an Automation Workflow Step by Step \(Command Line\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeAutomationStepExecutions](#) in *AWS CLI Command Reference*.

describe-available-patches

The following code example shows how to use `describe-available-patches`.

AWS CLI

To get available patches

The following `describe-available-patches` example retrieves details about all available patches for Windows Server 2019 that have a MSRC severity of Critical.

```
aws ssm describe-available-patches \  
  --filters "Key=PRODUCT,Values=WindowsServer2019"  
  "Key=MSRC_SEVERITY,Values=Critical"
```

Output:

```
{  
  "Patches": [  
    {  
      "Id": "fe6bd8c2-3752-4c8b-ab3e-1a7ed08767ba",  
      "ReleaseDate": 1544047205.0,  
      "Title": "2018-11 Update for Windows Server 2019 for x64-based Systems  
(KB4470788)",  
      "Description": "Install this update to resolve issues in Windows. For a  
complete listing of the issues that are included in this update, see the associated  
Microsoft Knowledge Base article for more information. After you install this item,  
you may have to restart your computer.",  
      "ContentUrl": "https://support.microsoft.com/en-us/kb/4470788",  
      "Vendor": "Microsoft",
```

```

        "ProductFamily": "Windows",
        "Product": "WindowsServer2019",
        "Classification": "SecurityUpdates",
        "MsrcSeverity": "Critical",
        "KbNumber": "KB4470788",
        "MsrcNumber": "",
        "Language": "All"
    },
    {
        "Id": "c96115e1-5587-4115-b851-22baa46a3f11",
        "ReleaseDate": 1549994410.0,
        "Title": "2019-02 Security Update for Adobe Flash Player for Windows
Server 2019 for x64-based Systems (KB4487038)",
        "Description": "A security issue has been identified in a Microsoft
software product that could affect your system. You can help protect your system
by installing this update from Microsoft. For a complete listing of the issues that
are included in this update, see the associated Microsoft Knowledge Base article.
After you install this update, you may have to restart your system.",
        "ContentUrl": "https://support.microsoft.com/en-us/kb/4487038",
        "Vendor": "Microsoft",
        "ProductFamily": "Windows",
        "Product": "WindowsServer2019",
        "Classification": "SecurityUpdates",
        "MsrcSeverity": "Critical",
        "KbNumber": "KB4487038",
        "MsrcNumber": "",
        "Language": "All"
    },
    ...
]
}

```

To get details of a specific patch

The following `describe-available-patches` example retrieves details about the specified patch.

```
aws ssm describe-available-patches \
  --filters "Key=PATCH_ID,Values=KB4480979"
```

Output:

```
{
```



```
"Patches": [
  {
    "Id": "680861e3-fb75-432e-818e-d72e5f2be719",
    "ReleaseDate": 1546970408.0,
    "Title": "2019-01 Security Update for Adobe Flash Player for Windows
Server 2016 for x64-based Systems (KB4480979)",
    "Description": "A security issue has been identified in a Microsoft
software product that could affect your system. You can help protect your system
by installing this update from Microsoft. For a complete listing of the issues that
are included in this update, see the associated Microsoft Knowledge Base article.
After you install this update, you may have to restart your system.",
    "ContentUrl": "https://support.microsoft.com/en-us/kb/4480979",
    "Vendor": "Microsoft",
    "ProductFamily": "Windows",
    "Product": "WindowsServer2016",
    "Classification": "SecurityUpdates",
    "MsrcSeverity": "Critical",
    "KbNumber": "KB4480979",
    "MsrcNumber": "",
    "Language": "All"
  }
]
```

For more information, see [How Patch Manager Operations Work](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeAvailablePatches](#) in *AWS CLI Command Reference*.

describe-document-permission

The following code example shows how to use `describe-document-permission`.

AWS CLI

To describe document permissions

The following `describe-document-permission` example displays permission details about a Systems Manager document that is shared publicly.

```
aws ssm describe-document-permission \
  --name "Example" \
```

```
--permission-type "Share"
```

Output:

```
{
  "AccountIds": [
    "all"
  ],
  "AccountSharingInfoList": [
    {
      "AccountId": "all",
      "SharedDocumentVersion": "$DEFAULT"
    }
  ]
}
```

For more information, see [Share a Systems Manager Document](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeDocumentPermission](#) in *AWS CLI Command Reference*.

describe-document

The following code example shows how to use `describe-document`.

AWS CLI**To display details of a document**

The following `describe-document` example displays details about a Systems Manager document in your AWS account.

```
aws ssm describe-document \
  --name "Example"
```

Output:

```
{
  "Document": {
    "Hash": "fc2410281f40779e694a8b95975d0f9f316da8a153daa94e3d9921102EXAMPLE",
    "HashType": "Sha256",
    "Name": "Example",
  }
}
```

```
"Owner": "29884EXAMPLE",
"CreateDate": 1583257938.266,
"Status": "Active",
"DocumentVersion": "1",
"Description": "Document Example",
"Parameters": [
  {
    "Name": "AutomationAssumeRole",
    "Type": "String",
    "Description": "(Required) The ARN of the role that allows
Automation to perform the actions on your behalf. If no role is specified, Systems
Manager Automation uses your IAM permissions to execute this document.",
    "DefaultValue": ""
  },
  {
    "Name": "InstanceId",
    "Type": "String",
    "Description": "(Required) The ID of the Amazon EC2 instance.",
    "DefaultValue": ""
  }
],
"PlatformTypes": [
  "Windows",
  "Linux"
],
"DocumentType": "Automation",
"SchemaVersion": "0.3",
"LatestVersion": "1",
"DefaultVersion": "1",
"DocumentFormat": "YAML",
"Tags": []
}
```

For more information, see [Creating Systems Manager Documents](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeDocument](#) in *AWS CLI Command Reference*.

describe-effective-instance-associations

The following code example shows how to use `describe-effective-instance-associations`.

AWS CLI

To get details of the effective associations for an instance

The following `describe-effective-instance-associations` example retrieves details about the effective associations for an instance.

Command:

```
aws ssm describe-effective-instance-associations --instance-id "i-1234567890abcdef0"
```

Output:

```
{
  "Associations": [
    {
      "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
      "InstanceId": "i-1234567890abcdef0",
      "Content": "{\n  \"schemaVersion\": \"1.2\",\n  \"description\":\n  \"Update the Amazon SSM Agent to the latest version or specified version.\",\n  \"parameters\": {\n    \"version\": {\n      \"default\": \"\",\n      \"description\": \"(Optional) A specific version of the Amazon SSM Agent\n  to install. If not specified, the agent will be updated to the latest version.\",\n      \"type\": \"String\"\n    },\n    \"allowDowngrade\": {\n      \"default\": \"false\",\n      \"description\": \"(Optional)\n  Allow the Amazon SSM Agent service to be downgraded to an earlier version. If\n  set to false, the service can be upgraded to newer versions only (default). If\n  set to true, specify the earlier version.\",\n      \"type\": \"String\",\n      \"allowedValues\": [\n        \"true\",\n        \"false\"\n      ]\n    },\n    \"runtimeConfig\": {\n      \"aws:updateSsmAgent\": {\n        \"properties\": [\n          {\n            \"agentName\": \"amazon-ssm-agent\",\n            \"source\":\n            \"https://s3.{Region}.amazonaws.com/amazon-ssm-{Region}/ssm-agent-manifest.json\",\n            \"allowDowngrade\": \"{{ allowDowngrade }}\",\n            \"targetVersion\": \"{{ version }}\"\n          }\n        ]\n      }\n    }\n  }\n  \"AssociationVersion\": \"1\"\n    }
  ]
}
```

- For API details, see [DescribeEffectiveInstanceAssociations](#) in *AWS CLI Command Reference*.

describe-effective-patches-for-patch-baseline

The following code example shows how to use `describe-effective-patches-for-patch-baseline`.

AWS CLI

Example 1: To get all patches defined by a custom patch baseline

The following `describe-effective-patches-for-patch-baseline` example returns the patches defined by a custom patch baseline in the current AWS account. Note that for a custom baseline, only the ID is required for `--baseline-id`.

```
aws ssm describe-effective-patches-for-patch-baseline \
  --baseline-id "pb-08b654cf9b9681f04"
```

Output:

```
{
  "EffectivePatches": [
    {
      "Patch": {
        "Id": "fe6bd8c2-3752-4c8b-ab3e-1a7ed08767ba",
        "ReleaseDate": 1544047205.0,
        "Title": "2018-11 Update for Windows Server 2019 for x64-based
Systems (KB4470788)",
        "Description": "Install this update to resolve issues in Windows.
For a complete listing of the issues that are included in this update, see the
associated Microsoft Knowledge Base article for more information. After you install
this item, you may have to restart your computer.",
        "ContentUrl": "https://support.microsoft.com/en-us/kb/4470788",
        "Vendor": "Microsoft",
        "ProductFamily": "Windows",
        "Product": "WindowsServer2019",
        "Classification": "SecurityUpdates",
        "MsrcSeverity": "Critical",
        "KbNumber": "KB4470788",
        "MsrcNumber": "",
        "Language": "All"
      },
      "PatchStatus": {
        "DeploymentStatus": "APPROVED",
```

```

        "ComplianceLevel": "CRITICAL",
        "ApprovalDate": 1544047205.0
    },
    {
        "Patch": {
            "Id": "915a6b1a-f556-4d83-8f50-b2e75a9a7e58",
            "ReleaseDate": 1549994400.0,
            "Title": "2019-02 Cumulative Update for .NET Framework 3.5 and 4.7.2
for Windows Server 2019 for x64 (KB4483452)",
            "Description": "A security issue has been identified in a Microsoft
software product that could affect your system. You can help protect your system by
installing this update from Microsoft. For a complete listing of the issues that
are included in this update, see the associated Microsoft Knowledge Base article.
After you install this update, you may have to restart your system.",
            "ContentUrl": "https://support.microsoft.com/en-us/kb/4483452",
            "Vendor": "Microsoft",
            "ProductFamily": "Windows",
            "Product": "WindowsServer2019",
            "Classification": "SecurityUpdates",
            "MsrcSeverity": "Important",
            "KbNumber": "KB4483452",
            "MsrcNumber": "",
            "Language": "All"
        },
        "PatchStatus": {
            "DeploymentStatus": "APPROVED",
            "ComplianceLevel": "CRITICAL",
            "ApprovalDate": 1549994400.0
        }
    },
    ...
],
"NextToken": "--token string truncated--"
}

```

Example 2: To get all patches defined by an AWS managed patch baseline

The following `describe-effective-patches-for-patch-baseline` example returns the patches defined by an AWS managed patch baseline. Note that for an AWS managed baseline, the complete baseline ARN is required for `--baseline-id`

```
aws ssm describe-effective-patches-for-patch-baseline \
```

```
--baseline-id "arn:aws:ssm:us-east-2:733109147000:patchbaseline/  
pb-020d361a05defe4ed"
```

See example 1 for sample output.

For more information, see [How Security Patches Are Selected](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeEffectivePatchesForPatchBaseline](#) in *AWS CLI Command Reference*.

describe-instance-associations-status

The following code example shows how to use `describe-instance-associations-status`.

AWS CLI

To describe the status of an instance's associations

This example shows details of the associations for an instance.

Command:

```
aws ssm describe-instance-associations-status --instance-id "i-1234567890abcdef0"
```

Output:

```
{  
  "InstanceAssociationStatusInfos": [  
    {  
      "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",  
      "Name": "AWS-GatherSoftwareInventory",  
      "DocumentVersion": "1",  
      "AssociationVersion": "1",  
      "InstanceId": "i-1234567890abcdef0",  
      "ExecutionDate": 1550501886.0,  
      "Status": "Success",  
      "ExecutionSummary": "1 out of 1 plugin processed, 1 success, 0 failed, 0  
      timedout, 0 skipped. ",  
      "AssociationName": "Inventory-Association"  
    },  
    {  
      "AssociationId": "5c5a31f6-6dae-46f9-944c-0123456789ab",
```

```
    "Name": "AWS-UpdateSSMAgent",
    "DocumentVersion": "1",
    "AssociationVersion": "1",
    "InstanceId": "i-1234567890abcdef0",
    "ExecutionDate": 1550505828.548,
    "Status": "Success",
    "DetailedStatus": "Success",
    "AssociationName": "UpdateSSMAgent"
  }
]
```

- For API details, see [DescribeInstanceAssociationsStatus](#) in *AWS CLI Command Reference*.

describe-instance-information

The following code example shows how to use `describe-instance-information`.

AWS CLI

Example 1: To describe managed instance information

The following `describe-instance-information` example retrieves details of each of your managed instances.

```
aws ssm describe-instance-information
```

Example 2: To describe information about a specific managed instance

The following `describe-instance-information` example shows details of the managed instance `i-028ea792daEXAMPLE`.

```
aws ssm describe-instance-information \
  --filters "Key=InstanceIds,Values=i-028ea792daEXAMPLE"
```

Example 3: To describe information about managed instances with a specific tag key

The following `describe-instance-information` example shows details for managed instances that have the tag key `DEV`.

```
aws ssm describe-instance-information \
```



```
--filters "Key=tag-key,Values=DEV"
```

Output:

```
{
  "InstanceInformationList": [
    {
      "InstanceId": "i-028ea792daEXAMPLE",
      "PingStatus": "Online",
      "LastPingDateTime": 1582221233.421,
      "AgentVersion": "2.3.842.0",
      "IsLatestVersion": true,
      "PlatformType": "Linux",
      "PlatformName": "SLES",
      "PlatformVersion": "15.1",
      "ResourceType": "EC2Instance",
      "IPAddress": "192.0.2.0",
      "ComputerName": "ip-198.51.100.0.us-east-2.compute.internal",
      "AssociationStatus": "Success",
      "LastAssociationExecutionDate": 1582220806.0,
      "LastSuccessfulAssociationExecutionDate": 1582220806.0,
      "AssociationOverview": {
        "DetailedStatus": "Success",
        "InstanceAssociationStatusAggregatedCount": {
          "Success": 2
        }
      }
    }
  ]
}
```

For more information, see [Managed Instances](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeInstanceInformation](#) in *AWS CLI Command Reference*.

describe-instance-patch-states-for-patch-group

The following code example shows how to use `describe-instance-patch-states-for-patch-group`.

AWS CLI

Example 1: To get the instance states for a patch group

The following `describe-instance-patch-states-for-patch-group` example retrieves details about the patch summary states per-instance for the specified patch group.

```
aws ssm describe-instance-patch-states-for-patch-group \  
  --patch-group "Production"
```

Output:

```
{  
  "InstancePatchStates": [  
    {  
      "InstanceId": "i-02573cafcfEXAMPLE",  
      "PatchGroup": "Production",  
      "BaselineId": "pb-0c10e65780EXAMPLE",  
      "SnapshotId": "a3f5ff34-9bc4-4d2c-a665-4d1c1EXAMPLE",  
      "OwnerInformation": "",  
      "InstalledCount": 32,  
      "InstalledOtherCount": 1,  
      "InstalledPendingRebootCount": 0,  
      "InstalledRejectedCount": 0,  
      "MissingCount": 2,  
      "FailedCount": 0,  
      "UnreportedNotApplicableCount": 2671,  
      "NotApplicableCount": 400,  
      "OperationStartTime": "2021-08-04T11:03:50.590000-07:00",  
      "OperationEndTime": "2021-08-04T11:04:21.555000-07:00",  
      "Operation": "Scan",  
      "RebootOption": "NoReboot",  
      "CriticalNonCompliantCount": 0,  
      "SecurityNonCompliantCount": 1,  
      "OtherNonCompliantCount": 0  
    },  
    {  
      "InstanceId": "i-0471e04240EXAMPLE",  
      "PatchGroup": "Production",  
      "BaselineId": "pb-09ca3fb51fEXAMPLE",  
      "SnapshotId": "05d8ffb0-1bbe-4812-ba2d-d9b7bEXAMPLE",  
      "OwnerInformation": "",  
      "InstalledCount": 32,  
      "InstalledOtherCount": 1,  
      "InstalledPendingRebootCount": 0,  
      "InstalledRejectedCount": 0,  
      "MissingCount": 2,  
      "FailedCount": 0,  
      "UnreportedNotApplicableCount": 2671,  
      "NotApplicableCount": 400,  
      "OperationStartTime": "2021-08-04T11:03:50.590000-07:00",  
      "OperationEndTime": "2021-08-04T11:04:21.555000-07:00",  
      "Operation": "Scan",  
      "RebootOption": "NoReboot",  
      "CriticalNonCompliantCount": 0,  
      "SecurityNonCompliantCount": 1,  
      "OtherNonCompliantCount": 0  
    }  
  ]  
}
```

```

    "FailedCount": 0,
    "UnreportedNotApplicableCount": 2671,
    "NotApplicableCount": 400,
    "OperationStartTime": "2021-08-04T22:06:20.340000-07:00",
    "OperationEndTime": "2021-08-04T22:07:11.220000-07:00",
    "Operation": "Scan",
    "RebootOption": "NoReboot",
    "CriticalNonCompliantCount": 0,
    "SecurityNonCompliantCount": 1,
    "OtherNonCompliantCount": 0
  }
]
}

```

Example 2: To get the instance states for a patch group with more than five missing patches

The following `describe-instance-patch-states-for-patch-group` example retrieves details about the patch summary states for the specified patch group for instances with more than five missing patches.

```

aws ssm describe-instance-patch-states-for-patch-group \
  --filters Key=MissingCount,Type=GreaterThan,Values=5 \
  --patch-group "Production"

```

Output:

```

{
  "InstancePatchStates": [
    {
      "InstanceId": "i-02573cafcfEXAMPLE",
      "PatchGroup": "Production",
      "BaselineId": "pb-0c10e65780EXAMPLE",
      "SnapshotId": "a3f5ff34-9bc4-4d2c-a665-4d1c1EXAMPLE",
      "OwnerInformation": "",
      "InstalledCount": 46,
      "InstalledOtherCount": 4,
      "InstalledPendingRebootCount": 1,
      "InstalledRejectedCount": 1,
      "MissingCount": 7,
      "FailedCount": 0,
      "UnreportedNotApplicableCount": 232,
      "NotApplicableCount": 654,

```

```

    "OperationStartTime": "2021-08-04T11:03:50.590000-07:00",
    "OperationEndTime": "2021-08-04T11:04:21.555000-07:00",
    "Operation": "Scan",
    "RebootOption": "NoReboot",
    "CriticalNonCompliantCount": 0,
    "SecurityNonCompliantCount": 1,
    "OtherNonCompliantCount": 1
  }
]
}

```

Example 3: To get the instance states for a patch group with fewer than ten instances that require a reboot

The following `describe-instance-patch-states-for-patch-group` example retrieves details about the patch summary states for the specified patch group for instances with fewer than ten instances requiring a reboot.

```

aws ssm describe-instance-patch-states-for-patch-group \
  --filters Key=InstalledPendingRebootCount,Type=LessThan,Values=10 \
  --patch-group "Production"

```

Output:

```

{
  "InstancePatchStates": [
    {
      "InstanceId": "i-02573cafcfEXAMPLE",
      "BaselineId": "pb-0c10e65780EXAMPLE",
      "SnapshotId": "a3f5ff34-9bc4-4d2c-a665-4d1c1EXAMPLE",
      "PatchGroup": "Production",
      "OwnerInformation": "",
      "InstalledCount": 32,
      "InstalledOtherCount": 1,
      "InstalledPendingRebootCount": 4,
      "InstalledRejectedCount": 0,
      "MissingCount": 2,
      "FailedCount": 0,
      "UnreportedNotApplicableCount": 846,
      "NotApplicableCount": 212,
      "OperationStartTime": "2021-08-04T11:03:50.590000-07:00",
      "OperationEndTime": "2021-08-06T11:04:21.555000-07:00",
    }
  ]
}

```

```
        "Operation": "Scan",
        "RebootOption": "NoReboot",
        "CriticalNonCompliantCount": 0,
        "SecurityNonCompliantCount": 1,
        "OtherNonCompliantCount": 0
    }
]
}
```

For more information, see [Understanding patch compliance state values](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeInstancePatchStatesForPatchGroup](#) in *AWS CLI Command Reference*.

describe-instance-patch-states

The following code example shows how to use `describe-instance-patch-states`.

AWS CLI

To get the patch summary states for instances

This `describe-instance-patch-states` example gets the patch summary states for an instance.

```
aws ssm describe-instance-patch-states \
  --instance-ids "i-1234567890abcdef0"
```

Output:

```
{
  "InstancePatchStates": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "PatchGroup": "my-patch-group",
      "BaselineId": "pb-0713accee01234567",
      "SnapshotId": "521c3536-930c-4aa9-950e-01234567abcd",
      "CriticalNonCompliantCount": 2,
      "SecurityNonCompliantCount": 2,
      "OtherNonCompliantCount": 1,
    }
  ]
}
```

```

    "InstalledCount": 123,
    "InstalledOtherCount": 334,
    "InstalledPendingRebootCount": 0,
    "InstalledRejectedCount": 0,
    "MissingCount": 1,
    "FailedCount": 2,
    "UnreportedNotApplicableCount": 11,
    "NotApplicableCount": 2063,
    "OperationStartTime": "2021-05-03T11:00:56-07:00",
    "OperationEndTime": "2021-05-03T11:01:09-07:00",
    "Operation": "Scan",
    "LastNoRebootInstallOperationTime": "2020-06-14T12:17:41-07:00",
    "RebootOption": "RebootIfNeeded"
  }
]
}

```

For more information, see [About Patch Compliance](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeInstancePatchStates](#) in *AWS CLI Command Reference*.

describe-instance-patches

The following code example shows how to use `describe-instance-patches`.

AWS CLI

Example 1: To get the patch state details for an instance

The following `describe-instance-patches` example retrieves details about the patches for the specified instance.

```
aws ssm describe-instance-patches \
  --instance-id "i-1234567890abcdef0"
```

Output:

```
{
  "Patches": [
    {
      "Title": "2019-01 Security Update for Adobe Flash Player for Windows
Server 2016 for x64-based Systems (KB4480979)",
```

```

        "KBId": "KB4480979",
        "Classification": "SecurityUpdates",
        "Severity": "Critical",
        "State": "Installed",
        "InstalledTime": "2019-01-09T00:00:00+00:00"
    },
    {
        "Title": "",
        "KBId": "KB4481031",
        "Classification": "",
        "Severity": "",
        "State": "InstalledOther",
        "InstalledTime": "2019-02-08T00:00:00+00:00"
    },
    ...
],
"NextToken": "--token string truncated--"
}

```

Example 2: To get a list of patches in the Missing state for an instance

The following `describe-instance-patches` example retrieves information about patches that are in the Missing state for the specified instance.

```

aws ssm describe-instance-patches \
  --instance-id "i-1234567890abcdef0" \
  --filters Key=State,Values=Missing

```

Output:

```

{
  "Patches": [
    {
      "Title": "Windows Malicious Software Removal Tool x64 - February 2019 (KB890830)",
      "KBId": "KB890830",
      "Classification": "UpdateRollups",
      "Severity": "Unspecified",
      "State": "Missing",
      "InstalledTime": "1970-01-01T00:00:00+00:00"
    },
    ...
  ],
}

```

```
"NextToken": "--token string truncated--"
}
```

For more information, see [About Patch Compliance States](#) in the *AWS Systems Manager User Guide*.

Example 3: To get a list of patches installed since a specified `InstalledTime` for an instance

The following `describe-instance-patches` example retrieves information about patches installed since a specified time for the specified instance by combining the use of `--filters` and `--query`.

```
aws ssm describe-instance-patches \
  --instance-id "i-1234567890abcdef0" \
  --filters Key=State,Values=Installed \
  --query "Patches[?InstalledTime >= `2023-01-01T16:00:00`]"
```

Output:

```
{
  "Patches": [
    {
      "Title": "2023-03 Cumulative Update for Windows Server 2019 (1809) for
x64-based Systems (KB5023702)",
      "KBId": "KB5023702",
      "Classification": "SecurityUpdates",
      "Severity": "Critical",
      "State": "Installed",
      "InstalledTime": "2023-03-16T11:00:00+00:00"
    },
    ...
  ],
  "NextToken": "--token string truncated--"
}
```

- For API details, see [DescribeInstancePatches](#) in *AWS CLI Command Reference*.

describe-inventory-deletions

The following code example shows how to use `describe-inventory-deletions`.

AWS CLI

To get inventory deletions

This example retrieves details for inventory deletion operations.

Command:

```
aws ssm describe-inventory-deletions
```

Output:

```
{
  "InventoryDeletions": [
    {
      "DeletionId": "6961492a-8163-44ec-aa1e-01234567850",
      "TypeName": "Custom:RackInformation",
      "DeletionStartTime": 1550254911.0,
      "LastStatus": "InProgress",
      "LastStatusMessage": "The Delete is in progress",
      "DeletionSummary": {
        "TotalCount": 0,
        "RemainingCount": 0,
        "SummaryItems": []
      },
      "LastStatusUpdateTime": 1550254911.0
    },
    {
      "DeletionId": "d72ac9e8-1f60-4d40-b1c6-987654321c4d",
      "TypeName": "Custom:RackInfo",
      "DeletionStartTime": 1550254859.0,
      "LastStatus": "InProgress",
      "LastStatusMessage": "The Delete is in progress",
      "DeletionSummary": {
        "TotalCount": 1,
        "RemainingCount": 1,
        "SummaryItems": [
          {
            "Version": "1.0",
            "Count": 1,
            "RemainingCount": 1
          }
        ]
      }
    }
  ]
}
```

```
    },
    "LastStatusUpdateTime": 1550254859.0
  }
]
}
```

To get details of a specific inventory deletion

This example retrieves details for a specific inventory deletion operation.

Command:

```
aws ssm describe-inventory-deletions --deletion-id "d72ac9e8-1f60-4d40-
b1c6-987654321c4d"
```

Output:

```
{
  "InventoryDeletions": [
    {
      "DeletionId": "d72ac9e8-1f60-4d40-b1c6-987654321c4d",
      "TypeName": "Custom:RackInfo",
      "DeletionStartTime": 1550254859.0,
      "LastStatus": "InProgress",
      "LastStatusMessage": "The Delete is in progress",
      "DeletionSummary": {
        "TotalCount": 1,
        "RemainingCount": 1,
        "SummaryItems": [
          {
            "Version": "1.0",
            "Count": 1,
            "RemainingCount": 1
          }
        ]
      },
      "LastStatusUpdateTime": 1550254859.0
    }
  ]
}
```

- For API details, see [DescribeInventoryDeletions](#) in *AWS CLI Command Reference*.

describe-maintenance-window-execution-task-invocations

The following code example shows how to use `describe-maintenance-window-execution-task-invocations`.

AWS CLI

To get the specific task invocations performed for a maintenance window task execution

The following `describe-maintenance-window-execution-task-invocations` example lists the invocations for the specified task executed as part of the specified maintenance window execution.

```
aws ssm describe-maintenance-window-execution-task-invocations \
  --window-execution-id "518d5565-5969-4cca-8f0e-da3b2a638355" \
  --task-id "ac0c6ae1-daa3-4a89-832e-d384503b6586"
```

Output:

```
{
  "WindowExecutionTaskInvocationIdentities": [
    {
      "Status": "SUCCESS",
      "Parameters": "{\"documentName\":\"AWS-RunShellScript\",\"instanceIds\":[\"i-0000293ffd8c57862\"],\"parameters\":{\"commands\":[\"df\"]},\"maxConcurrency\":\"1\",\"maxErrors\":\"1\"}",
      "InvocationId": "e274b6e1-fe56-4e32-bd2a-8073c6381d8b",
      "StartTime": 1487692834.723,
      "EndTime": 1487692834.871,
      "WindowExecutionId": "518d5565-5969-4cca-8f0e-da3b2a638355",
      "TaskExecutionId": "ac0c6ae1-daa3-4a89-832e-d384503b6586"
    }
  ]
}
```

For more information, see [View Information About Tasks and Task Executions \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeMaintenanceWindowExecutionTaskInvocations](#) in *AWS CLI Command Reference*.

describe-maintenance-window-execution-tasks

The following code example shows how to use `describe-maintenance-window-execution-tasks`.

AWS CLI

To list all tasks associated with a maintenance window execution

The following `ssm describe-maintenance-window-execution-tasks` example lists the tasks associated with the specified maintenance window execution.

```
aws ssm describe-maintenance-window-execution-tasks \  
  --window-execution-id "518d5565-5969-4cca-8f0e-da3b2EXAMPLE"
```

Output:

```
{  
  "WindowExecutionTaskIdentities": [  
    {  
      "Status": "SUCCESS",  
      "TaskArn": "AWS-RunShellScript",  
      "StartTime": 1487692834.684,  
      "TaskType": "RUN_COMMAND",  
      "EndTime": 1487692835.005,  
      "WindowExecutionId": "518d5565-5969-4cca-8f0e-da3b2EXAMPLE",  
      "TaskExecutionId": "ac0c6ae1-daa3-4a89-832e-d3845EXAMPLE"  
    }  
  ]  
}
```

For more information, see [View Information About Tasks and Task Executions \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeMaintenanceWindowExecutionTasks](#) in *AWS CLI Command Reference*.

describe-maintenance-window-executions

The following code example shows how to use `describe-maintenance-window-executions`.

AWS CLI

Example 1: To list all executions for a maintenance window

The following `describe-maintenance-window-executions` example lists all of the executions for the specified maintenance window.

```
aws ssm describe-maintenance-window-executions \  
  --window-id "mw-ab12cd34eEXAMPLE"
```

Output:

```
{  
  "WindowExecutions": [  
    {  
      "WindowId": "mw-ab12cd34eEXAMPLE",  
      "WindowExecutionId": "6027b513-64fe-4cf0-be7d-1191aEXAMPLE",  
      "Status": "IN_PROGRESS",  
      "StartTime": "2021-08-04T11:00:00.000000-07:00"  
    },  
    {  
      "WindowId": "mw-ab12cd34eEXAMPLE",  
      "WindowExecutionId": "ff75b750-4834-4377-8f61-b3cadEXAMPLE",  
      "Status": "SUCCESS",  
      "StartTime": "2021-08-03T11:00:00.000000-07:00",  
      "EndTime": "2021-08-03T11:37:21.450000-07:00"  
    },  
    {  
      "WindowId": "mw-ab12cd34eEXAMPLE",  
      "WindowExecutionId": "9fac7dd9-ff21-42a5-96ad-bbc4bEXAMPLE",  
      "Status": "FAILED",  
      "StatusDetails": "One or more tasks in the orchestration failed.",  
      "StartTime": "2021-08-02T11:00:00.000000-07:00",  
      "EndTime": "2021-08-02T11:22:36.190000-07:00"  
    }  
  ]  
}
```

Example 2: To list all executions for a maintenance window before a specified date

The following `describe-maintenance-window-executions` example lists all of the executions for the specified maintenance window before the specified date.

```
aws ssm describe-maintenance-window-executions \  
  --window-id "mw-ab12cd34eEXAMPLE" \  
  --filters "Key=ExecutedBefore,Values=2021-08-03T00:00:00Z"
```

Output:

```
{  
  "WindowExecutions": [  
    {  
      "WindowId": "mw-ab12cd34eEXAMPLE",  
      "WindowExecutionId": "9fac7dd9-ff21-42a5-96ad-bbc4bEXAMPLE",  
      "Status": "FAILED",  
      "StatusDetails": "One or more tasks in the orchestration failed.",  
      "StartTime": "2021-08-02T11:00:00.000000-07:00",  
      "EndTime": "2021-08-02T11:22:36.190000-07:00"  
    }  
  ]  
}
```

Example 3: To list all executions for a maintenance window after a specified date

The following `describe-maintenance-window-executions` example lists all of the executions for the specified maintenance window after the specified date.

```
aws ssm describe-maintenance-window-executions \  
  --window-id "mw-ab12cd34eEXAMPLE" \  
  --filters "Key=ExecutedAfter,Values=2021-08-04T00:00:00Z"
```

Output:

```
{  
  "WindowExecutions": [  
    {  
      "WindowId": "mw-ab12cd34eEXAMPLE",  
      "WindowExecutionId": "6027b513-64fe-4cf0-be7d-1191aEXAMPLE",  
      "Status": "IN_PROGRESS",  
      "StartTime": "2021-08-04T11:00:00.000000-07:00"  
    }  
  ]  
}
```

For more information, see [View information about tasks and task executions \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeMaintenanceWindowExecutions](#) in *AWS CLI Command Reference*.

describe-maintenance-window-schedule

The following code example shows how to use `describe-maintenance-window-schedule`.

AWS CLI

Example 1: To list upcoming executions for a maintenance window

The following `describe-maintenance-window-schedule` example lists all upcoming executions for the specified maintenance window.

```
aws ssm describe-maintenance-window-schedule \
  --window-id mw-ab12cd34eEXAMPLE
```

Output:

```
{
  "ScheduledWindowExecutions": [
    {
      "WindowId": "mw-ab12cd34eEXAMPLE",
      "Name": "My-First-Maintenance-Window",
      "ExecutionTime": "2020-02-19T16:00Z"
    },
    {
      "WindowId": "mw-ab12cd34eEXAMPLE",
      "Name": "My-First-Maintenance-Window",
      "ExecutionTime": "2020-02-26T16:00Z"
    },
    ...
  ]
}
```

Example 2: To list all upcoming executions for a maintenance window before a specified date

The following `describe-maintenance-window-schedule` example lists all upcoming executions for the specified maintenance window that occur before the specified date.

```
aws ssm describe-maintenance-window-schedule \  
  --window-id mw-0ecb1226dd7b2e9a6 \  
  --filters "Key=ScheduledBefore,Values=2020-02-15T06:00:00Z"
```

Example 3: To list all upcoming executions for a maintenance window after a specified date

The following `describe-maintenance-window-schedule` example lists all upcoming executions for the specified maintenance window that occur after the specified date.

```
aws ssm describe-maintenance-window-schedule \  
  --window-id mw-0ecb1226dd7b2e9a6 \  
  --filters "Key=ScheduledAfter,Values=2020-02-15T06:00:00Z"
```

For more information, see [View Information About Maintenance Windows \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeMaintenanceWindowSchedule](#) in *AWS CLI Command Reference*.

describe-maintenance-window-targets

The following code example shows how to use `describe-maintenance-window-targets`.

AWS CLI

Example 1: To list all targets for a Maintenance Window

The following `describe-maintenance-window-targets` example lists all of the targets for a maintenance window.

```
aws ssm describe-maintenance-window-targets \  
  --window-id "mw-06cf17cbefEXAMPLE"
```

Output:

```
{  
  "Targets": [  
    {  
      "ResourceType": "INSTANCE",  
      "OwnerInformation": "Single instance",  
      "WindowId": "mw-06cf17cbefEXAMPLE",  
      "Targets": [  
        {
```



```

        "Values": [
            "i-0000293ffdEXAMPLE"
        ],
        "Key": "InstanceIds"
    }
],
"WindowTargetId": "350d44e6-28cc-44e2-951f-4b2c9EXAMPLE"
},
{
    "ResourceType": "INSTANCE",
    "OwnerInformation": "Two instances in a list",
    "WindowId": "mw-06cf17cbefEXAMPLE",
    "Targets": [
        {
            "Values": [
                "i-0000293ffdEXAMPLE",
                "i-0cb2b964d3EXAMPLE"
            ],
            "Key": "InstanceIds"
        }
    ],
    "WindowTargetId": "e078a987-2866-47be-bedd-d9cf4EXAMPLE"
}
]
}

```

Example 2: To list all targets for a maintenance window matching a specific owner information value

This `describe-maintenance-window-targets` example lists all of the targets for a maintenance window with a specific value.

```

aws ssm describe-maintenance-window-targets \
  --window-id "mw-0ecb1226ddEXAMPLE" \
  --filters "Key=OwnerInformation,Values=CostCenter1"

```

Output:

```

{
  "Targets": [
    {
      "WindowId": "mw-0ecb1226ddEXAMPLE",
      "WindowTargetId": "da89dcc3-7f9c-481d-ba2b-edcb7d0057f9",

```

```

    "ResourceType": "INSTANCE",
    "Targets": [
      {
        "Key": "tag:Environment",
        "Values": [
          "Prod"
        ]
      }
    ],
    "OwnerInformation": "CostCenter1",
    "Name": "ProdTarget1"
  }
]
}

```

For more information, see [View Information About Maintenance Windows \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeMaintenanceWindowTargets](#) in *AWS CLI Command Reference*.

describe-maintenance-window-tasks

The following code example shows how to use `describe-maintenance-window-tasks`.

AWS CLI

Example 1: To list all tasks for a maintenance window

The following `describe-maintenance-window-tasks` example lists all of the tasks for the specified maintenance window.

```
aws ssm describe-maintenance-window-tasks \
  --window-id "mw-06cf17cbefEXAMPLE"
```

Output:

```

{
  "Tasks": [
    {
      "WindowId": "mw-06cf17cbefEXAMPLE",
      "WindowTaskId": "018b31c3-2d77-4b9e-bd48-c91edEXAMPLE",
      "TaskArn": "AWS-RestartEC2Instance",
      "TaskParameters": {},
    }
  ]
}

```

```

    "Type": "AUTOMATION",
    "Description": "Restarting EC2 Instance for maintenance",
    "MaxConcurrency": "1",
    "MaxErrors": "1",
    "Name": "My-Automation-Example-Task",
    "Priority": 0,
    "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
    "Targets": [
      {
        "Key": "WindowTargetIds",
        "Values": [
          "da89dcc3-7f9c-481d-ba2b-edcb7EXAMPLE"
        ]
      }
    ]
  },
  {
    "WindowId": "mw-06cf17cbefEXAMPLE",
    "WindowTaskId": "1943dee0-0a17-4978-9bf4-3cc2fEXAMPLE",
    "TaskArn": "AWS-DisableS3BucketPublicReadWrite",
    "TaskParameters": {},
    "Type": "AUTOMATION",
    "Description": "Automation task to disable read/write access on public
S3 buckets",
    "MaxConcurrency": "10",
    "MaxErrors": "5",
    "Name": "My-Disable-S3-Public-Read-Write-Access-Automation-Task",
    "Priority": 0,
    "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
    "Targets": [
      {
        "Key": "WindowTargetIds",
        "Values": [
          "da89dcc3-7f9c-481d-ba2b-edcb7EXAMPLE"
        ]
      }
    ]
  }
]
}

```

Example 2: To list all tasks for a maintenance window that invokes the AWS-RunPowerShellScript command document

The following describe-maintenance-window-tasks example lists all of the tasks for the specified maintenance window that invokes the AWS-RunPowerShellScript command document.

```
aws ssm describe-maintenance-window-tasks \  
  --window-id "mw-ab12cd34eEXAMPLE" \  
  --filters "Key=TaskArn,Values=AWS-RunPowerShellScript"
```

Output:

```
{  
  "Tasks": [  
    {  
      "WindowId": "mw-ab12cd34eEXAMPLE",  
      "WindowTaskId": "0d36e6b4-3a4f-411e-adcb-3558eEXAMPLE",  
      "TaskArn": "AWS-RunPowerShellScript",  
      "Type": "RUN_COMMAND",  
      "Targets": [  
        {  
          "Key": "WindowTargetIds",  
          "Values": [  
            "da89dcc3-7f9c-481d-ba2b-edcb7EXAMPLE"  
          ]  
        }  
      ],  
      "TaskParameters": {},  
      "Priority": 1,  
      "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/  
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",  
      "MaxConcurrency": "1",  
      "MaxErrors": "1",  
      "Name": "MyTask"  
    }  
  ]  
}
```

Example 3: To list all tasks for a maintenance window that have a Priority of 3

The following `describe-maintenance-window-tasks` example lists all of the tasks for the specified maintenance window that have a `Priority` of 3.

```
aws ssm describe-maintenance-window-tasks \  
  --window-id "mw-ab12cd34eEXAMPLE" \  
  --filters "Key=Priority,Values=3"
```

Output:

```
{  
  "Tasks": [  
    {  
      "WindowId": "mw-ab12cd34eEXAMPLE",  
      "WindowTaskId": "0d36e6b4-3a4f-411e-adcb-3558eEXAMPLE",  
      "TaskArn": "AWS-RunPowerShellScript",  
      "Type": "RUN_COMMAND",  
      "Targets": [  
        {  
          "Key": "WindowTargetIds",  
          "Values": [  
            "da89dcc3-7f9c-481d-ba2b-edcb7EXAMPLE"  
          ]  
        }  
      ],  
      "TaskParameters": {},  
      "Priority": 3,  
      "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/  
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",  
      "MaxConcurrency": "1",  
      "MaxErrors": "1",  
      "Name": "MyRunCommandTask"  
    },  
    {  
      "WindowId": "mw-ab12cd34eEXAMPLE",  
      "WindowTaskId": "ee45feff-ad65-4a6c-b478-5cab8EXAMPLE",  
      "TaskArn": "AWS-RestartEC2Instance",  
      "Type": "AUTOMATION",  
      "Targets": [  
        {  
          "Key": "WindowTargetIds",  
          "Values": [  
            "da89dcc3-7f9c-481d-ba2b-edcb7EXAMPLE"  
          ]  
        }  
      ]  
    }  
  ]  
}
```

```

        }
    ],
    "TaskParameters": {},
    "Priority": 3,
    "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
    "MaxConcurrency": "10",
    "MaxErrors": "5",
    "Name": "My-Automation-Task",
    "Description": "A description for my Automation task"
    }
]
}

```

Example 4: To list all tasks for a maintenance window that have a Priority of 1 and use Run Command

This describe-maintenance-window-tasks example lists all of the tasks for the specified maintenance window that have a Priority of 1 and use Run Command.

```

aws ssm describe-maintenance-window-tasks \
  --window-id "mw-ab12cd34eEXAMPLE" \
  --filters "Key=Priority,Values=1" "Key=TaskType,Values=RUN_COMMAND"

```

Output:

```

{
  "Tasks": [
    {
      "WindowId": "mw-ab12cd34eEXAMPLE",
      "WindowTaskId": "0d36e6b4-3a4f-411e-adcb-3558eEXAMPLE",
      "TaskArn": "AWS-RunPowerShellScript",
      "Type": "RUN_COMMAND",
      "Targets": [
        {
          "Key": "WindowTargetIds",
          "Values": [
            "da89dcc3-7f9c-481d-ba2b-edcb7EXAMPLE"
          ]
        }
      ]
    },
    {
      "TaskParameters": {},
      "Priority": 1,

```

```
        "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
        ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
        "MaxConcurrency": "1",
        "MaxErrors": "1",
        "Name": "MyRunCommandTask"
    }
]
}
```

For more information, see [View information about maintenance windows \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeMaintenanceWindowTasks](#) in *AWS CLI Command Reference*.

describe-maintenance-windows-for-target

The following code example shows how to use `describe-maintenance-windows-for-target`.

AWS CLI

To list all maintenance windows associated with a specific instance

The following `describe-maintenance-windows-for-target` example lists the maintenance windows that have targets or tasks associated with the specified instance.

```
aws ssm describe-maintenance-windows-for-target \
  --targets Key=InstanceIds,Values=i-1234567890EXAMPLE \
  --resource-type INSTANCE
```

Output:

```
{
  "WindowIdentities": [
    {
      "WindowId": "mw-0c5ed765acEXAMPLE",
      "Name": "My-First-Maintenance-Window"
    }
  ]
}
```

For more information, see [View Information About Maintenance Windows \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeMaintenanceWindowsForTarget](#) in *AWS CLI Command Reference*.

describe-maintenance-windows

The following code example shows how to use `describe-maintenance-windows`.

AWS CLI

Example 1: To list all maintenance windows

The following `describe-maintenance-windows` example lists all maintenance windows in your AWS account in the current Region.

```
aws ssm describe-maintenance-windows
```

Output:

```
{
  "WindowIdentities": [
    {
      "WindowId": "mw-0ecb1226ddEXAMPLE",
      "Name": "MyMaintenanceWindow-1",
      "Enabled": true,
      "Duration": 2,
      "Cutoff": 1,
      "Schedule": "rate(180 minutes)",
      "NextExecutionTime": "2020-02-12T23:19:20.596Z"
    },
    {
      "WindowId": "mw-03eb9db428EXAMPLE",
      "Name": "MyMaintenanceWindow-2",
      "Enabled": true,
      "Duration": 3,
      "Cutoff": 1,
      "Schedule": "rate(7 days)",
      "NextExecutionTime": "2020-02-17T23:22:00.956Z"
    }
  ]
}
```

Example 2: To list all enabled maintenance windows

The following `describe-maintenance-windows` example lists all enabled maintenance windows.

```
aws ssm describe-maintenance-windows \  
  --filters "Key=Enabled,Values=true"
```

Example 3: To list maintenance windows matching a specific name

This `describe-maintenance-windows` example lists all maintenance windows with the specified name.

```
aws ssm describe-maintenance-windows \  
  --filters "Key=Name,Values=MyMaintenanceWindow"
```

For more information, see [View Information About Maintenance Windows \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeMaintenanceWindows](#) in *AWS CLI Command Reference*.

describe-ops-items

The following code example shows how to use `describe-ops-items`.

AWS CLI

To list a set of OpsItems

The following `describe-ops-items` example displays a list of all open OpsItems in your AWS account.

```
aws ssm describe-ops-items \  
  --ops-item-filters "Key=Status,Values=Open,Operator=Equal"
```

Output:

```
{  
  "OpsItemSummaries": [  
    {  
      "CreatedBy": "arn:aws:sts::111222333444:assumed-role/OpsItem-CWE-Role/  
fbf77cbe264a33509569f23e4EXAMPLE",  
      "CreatedTime": "2020-03-14T17:02:46.375000-07:00",
```

```

        "LastModifiedBy": "arn:aws:sts::111222333444:assumed-role/OpsItem-CWE-
Role/fbf77cbe264a33509569f23e4EXAMPLE",
        "LastModifiedTime": "2020-03-14T17:02:46.375000-07:00",
        "Source": "SSM",
        "Status": "Open",
        "OpsItemId": "oi-7cfc5EXAMPLE",
        "Title": "SSM Maintenance Window execution failed",
        "OperationalData": {
            "/aws/dedup": {
                "Value": "{\"dedupString\": \"SSMOpsItems-SSM-maintenance-window-
execution-failed\"}",
                "Type": "SearchableString"
            },
            "/aws/resources": {
                "Value": "[{\"arn\": \"arn:aws:ssm:us-
east-2:111222333444:maintenancewindow/mw-034093d322EXAMPLE\"}]",
                "Type": "SearchableString"
            }
        },
        "Category": "Availability",
        "Severity": "3"
    },
    {
        "CreatedBy": "arn:aws:sts::1112223233444:assumed-role/OpsItem-CWE-Role/
fbf77cbe264a33509569f23e4EXAMPLE",
        "CreatedTime": "2020-02-26T11:43:15.426000-08:00",
        "LastModifiedBy": "arn:aws:sts::111222333444:assumed-role/OpsItem-CWE-
Role/fbf77cbe264a33509569f23e4EXAMPLE",
        "LastModifiedTime": "2020-02-26T11:43:15.426000-08:00",
        "Source": "EC2",
        "Status": "Open",
        "OpsItemId": "oi-6f966EXAMPLE",
        "Title": "EC2 instance stopped",
        "OperationalData": {
            "/aws/automations": {
                "Value": "[ { \"automationType\": \"AWS:SSM:Automation\",
\"automationId\": \"AWS-RestartEC2Instance\" } ]",
                "Type": "SearchableString"
            },
            "/aws/dedup": {
                "Value": "{\"dedupString\": \"SSMOpsItems-EC2-instance-stopped
\"}",
                "Type": "SearchableString"
            }
        },
    }

```

```

        "/aws/resources": {
            "Value": "[{\"arn\":\"arn:aws:ec2:us-
east-2:111222333444:instance/i-0beccfbc02EXAMPLE\"}]",
            "Type": "SearchableString"
        }
    },
    "Category": "Availability",
    "Severity": "3"
}
]
}

```

For more information, see [Working with OpsItems](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeOpsItems](#) in *AWS CLI Command Reference*.

describe-parameters

The following code example shows how to use describe-parameters.

AWS CLI

Example 1: To list all parameters

The following describe-parameters example lists all parameters in the current AWS account and Region.

```
aws ssm describe-parameters
```

Output:

```

{
  "Parameters": [
    {
      "Name": "MySecureStringParameter",
      "Type": "SecureString",
      "KeyId": "alias/aws/ssm",
      "LastModifiedDate": 1582155479.205,
      "LastModifiedUser": "arn:aws:sts::111222333444:assumed-role/Admin/
Richard-Roe-Managed",
      "Description": "This is a SecureString parameter",
      "Version": 2,
      "Tier": "Advanced",
    }
  ]
}

```

```

    "Policies": [
      {
        "PolicyText": "{\"Type\":\"Expiration\",\"Version\":\"1.0\",
\\Attributes\":{\"Timestamp\":\"2020-07-07T22:30:00Z\"}}",
        "PolicyType": "Expiration",
        "PolicyStatus": "Pending"
      },
      {
        "PolicyText": "{\"Type\":\"ExpirationNotification\",\"Version\":
\\\"1.0\\\",\\\"Attributes\":{\"Before\":\"12\\\",\\\"Unit\":\"Hours\"}}",
        "PolicyType": "ExpirationNotification",
        "PolicyStatus": "Pending"
      }
    ]
  },
  {
    "Name": "MyStringListParameter",
    "Type": "StringList",
    "LastModifiedDate": 1582154764.222,
    "LastModifiedUser": "arn:aws:iam::111222333444:user/Mary-Major",
    "Description": "This is a StringList parameter",
    "Version": 1,
    "Tier": "Standard",
    "Policies": []
  },
  {
    "Name": "MyStringParameter",
    "Type": "String",
    "LastModifiedDate": 1582154711.976,
    "LastModifiedUser": "arn:aws:iam::111222333444:user/Alejandro-Rosalez",
    "Description": "This is a String parameter",
    "Version": 1,
    "Tier": "Standard",
    "Policies": []
  },
  {
    "Name": "latestAmi",
    "Type": "String",
    "LastModifiedDate": 1580862415.521,
    "LastModifiedUser": "arn:aws:sts::111222333444:assumed-role/lambda-ssm-
role/Automation-UpdateSSM-Param",
    "Version": 3,
    "Tier": "Standard",
    "Policies": []
  }
}

```

```
    }  
  ]  
}
```

Example 2: To list all parameters matching specific metadata

This `describe-parameters` example lists all parameters matching a filter.

```
aws ssm describe-parameters --filters "Key=Type,Values=StringList"
```

Output:

```
{  
  "Parameters": [  
    {  
      "Name": "MyStringListParameter",  
      "Type": "StringList",  
      "LastModifiedDate": 1582154764.222,  
      "LastModifiedUser": "arn:aws:iam::111222333444:user/Mary-Major",  
      "Description": "This is a StringList parameter",  
      "Version": 1,  
      "Tier": "Standard",  
      "Policies": []  
    }  
  ]  
}
```

For more information, see [Searching for Systems Manager Parameters](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeParameters](#) in *AWS CLI Command Reference*.

describe-patch-baselines

The following code example shows how to use `describe-patch-baselines`.

AWS CLI

Example 1: To list all patch baselines

The following `describe-patch-baselines` example retrieves details for all patch baselines in your account in the current Region.

```
aws ssm describe-patch-baselines
```

Output:

```
{
  "BaselineIdentities": [
    {
      "BaselineName": "AWS-SuseDefaultPatchBaseline",
      "DefaultBaseline": true,
      "BaselineDescription": "Default Patch Baseline for Suse Provided by
AWS.",
      "BaselineId": "arn:aws:ssm:us-east-2:733109147000:patchbaseline/
pb-0123fdb36e334a3b2",
      "OperatingSystem": "SUSE"
    },
    {
      "BaselineName": "AWS-DefaultPatchBaseline",
      "DefaultBaseline": false,
      "BaselineDescription": "Default Patch Baseline Provided by AWS.",
      "BaselineId": "arn:aws:ssm:us-east-2:733109147000:patchbaseline/
pb-020d361a05defe4ed",
      "OperatingSystem": "WINDOWS"
    },
    ...
    {
      "BaselineName": "MyWindowsPatchBaseline",
      "DefaultBaseline": true,
      "BaselineDescription": "My patch baseline for EC2 instances for Windows
Server",
      "BaselineId": "pb-0ad00e0dd7EXAMPLE",
      "OperatingSystem": "WINDOWS"
    }
  ]
}
```

Example 2: To list all patch baselines provided by AWS

The following describe-patch-baselines example lists all patch baselines provided by AWS.

```
aws ssm describe-patch-baselines \
  --filters "Key=OWNER,Values=[AWS]"
```

Example 3: To list all patch baselines that you own

The following `describe-patch-baselines` example lists all custom patch baselines created in your account in the current Region.

```
aws ssm describe-patch-baselines \  
  --filters "Key=OWNER,Values=[Self]"
```

For more information, see [About Predefined and Custom Patch Baselines](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribePatchBaselines](#) in *AWS CLI Command Reference*.

`describe-patch-group-state`

The following code example shows how to use `describe-patch-group-state`.

AWS CLI

To get the state of a patch group

The following `describe-patch-group-state` example retrieves the high-level patch compliance summary for a patch group.

```
aws ssm describe-patch-group-state \  
  --patch-group "Production"
```

Output:

```
{  
  "Instances": 21,  
  "InstancesWithCriticalNonCompliantPatches": 1,  
  "InstancesWithFailedPatches": 2,  
  "InstancesWithInstalledOtherPatches": 3,  
  "InstancesWithInstalledPatches": 21,  
  "InstancesWithInstalledPendingRebootPatches": 2,  
  "InstancesWithInstalledRejectedPatches": 1,  
  "InstancesWithMissingPatches": 3,  
  "InstancesWithNotApplicablePatches": 4,  
  "InstancesWithOtherNonCompliantPatches": 1,  
  "InstancesWithSecurityNonCompliantPatches": 1,  
}
```

```
"InstancesWithUnreportedNotApplicablePatches": 2
}
```

For more information, see About patch groups <<https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-patchgroups.html>>__ and [Understanding patch compliance state values](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribePatchGroupState](#) in *AWS CLI Command Reference*.

describe-patch-groups

The following code example shows how to use describe-patch-groups.

AWS CLI

To display patch group registrations

The following describe-patch-groups example lists the patch group registrations.

```
aws ssm describe-patch-groups
```

Output:

```
{
  "Mappings": [
    {
      "PatchGroup": "Production",
      "BaselineIdentity": {
        "BaselineId": "pb-0123456789abcdef0",
        "BaselineName": "ProdPatching",
        "OperatingSystem": "WINDOWS",
        "BaselineDescription": "Patches for Production",
        "DefaultBaseline": false
      }
    },
    {
      "PatchGroup": "Development",
      "BaselineIdentity": {
        "BaselineId": "pb-0713accee01234567",
        "BaselineName": "DevPatching",
        "OperatingSystem": "WINDOWS",
        "BaselineDescription": "Patches for Development",

```



```
        "DefaultBaseline": true
      }
    },
    ...
  ]
}
```

For more information, see [Create a Patch Group](https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-group-tagging.html) <<https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-group-tagging.html>>__ and [Add a Patch Group to a Patch Baseline](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribePatchGroups](#) in *AWS CLI Command Reference*.

describe-patch-properties

The following code example shows how to use `describe-patch-properties`.

AWS CLI

To list the Amazon Linux patch availability

The following `describe-patch-properties` example displays a list of the Amazon Linux products for which patches are available in your AWS account.

```
aws ssm describe-patch-properties \
  --operating-system AMAZON_LINUX \
  --property PRODUCT
```

Output:

```
{
  "Properties": [
    {
      "Name": "AmazonLinux2012.03"
    },
    {
      "Name": "AmazonLinux2012.09"
    },
    {
      "Name": "AmazonLinux2013.03"
    },
    {
```

```
    "Name": "AmazonLinux2013.09"  
  },  
  {  
    "Name": "AmazonLinux2014.03"  
  },  
  {  
    "Name": "AmazonLinux2014.09"  
  },  
  {  
    "Name": "AmazonLinux2015.03"  
  },  
  {  
    "Name": "AmazonLinux2015.09"  
  },  
  {  
    "Name": "AmazonLinux2016.03"  
  },  
  {  
    "Name": "AmazonLinux2016.09"  
  },  
  {  
    "Name": "AmazonLinux2017.03"  
  },  
  {  
    "Name": "AmazonLinux2017.09"  
  },  
  {  
    "Name": "AmazonLinux2018.03"  
  }  
]  
}
```

For more information, see [About Patch Baselines](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribePatchProperties](#) in *AWS CLI Command Reference*.

describe-sessions

The following code example shows how to use describe-sessions.

AWS CLI

Example 1: To list all active Session Manager sessions

This `describe-sessions` example retrieves a list of the active sessions created most recently (both connected and disconnected sessions) over the past 30 days that were started by the specified user. This command returns only results for connections to targets initiated using Session Manager. It does not list connections made through other means, such as Remote Desktop Connections or SSH.

```
aws ssm describe-sessions \  
  --state "Active" \  
  --filters "key=Owner,value=arn:aws:sts::123456789012:assumed-role/Administrator/  
Shirley-Rodriguez"
```

Output:

```
{  
  "Sessions": [  
    {  
      "SessionId": "John-07a16060613c408b5",  
      "Target": "i-1234567890abcdef0",  
      "Status": "Connected",  
      "StartDate": 1550676938.352,  
      "Owner": "arn:aws:sts::123456789012:assumed-role/Administrator/Shirley-  
Rodriguez",  
      "OutputUrl": {}  
    },  
    {  
      "SessionId": "John-01edf534b8b56e8eb",  
      "Target": "i-9876543210abcdef0",  
      "Status": "Connected",  
      "StartDate": 1550676842.194,  
      "Owner": "arn:aws:sts::123456789012:assumed-role/Administrator/Shirley-  
Rodriguez",  
      "OutputUrl": {}  
    }  
  ]  
}
```

Example 2: To list all terminated Session Manager sessions

This `describe-sessions` example retrieves a list of the most recently terminated sessions from the past 30 days for all users.

```
aws ssm describe-sessions \  
  --state "Terminated"
```

```
--state "History"
```

Output:

```
{
  "Sessions": [
    {
      "SessionId": "Mary-Major-0022b1eb2b0d9e3bd",
      "Target": "i-1234567890abcdef0",
      "Status": "Terminated",
      "StartDate": 1550520701.256,
      "EndDate": 1550521931.563,
      "Owner": "arn:aws:sts::123456789012:assumed-role/Administrator/Mary-
Major"
    },
    {
      "SessionId": "Jane-Roe-0db53f487931ed9d4",
      "Target": "i-9876543210abcdef0",
      "Status": "Terminated",
      "StartDate": 1550161369.149,
      "EndDate": 1550162580.329,
      "Owner": "arn:aws:sts::123456789012:assumed-role/Administrator/Jane-Roe"
    },
    ...
  ],
  "NextToken": "--token string truncated--"
}
```

For more information, see [View Session History](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DescribeSessions](#) in *AWS CLI Command Reference*.

disassociate-ops-item-related-item

The following code example shows how to use `disassociate-ops-item-related-item`.

AWS CLI

To delete a related item association

The following `disassociate-ops-item-related-item` example deletes the association between the `OpsItem` and a related item.

```
aws ssm disassociate-ops-item-related-item \  
  --ops-item-id "oi-f99f2EXAMPLE" \  
  --association-id "e2036148-cccb-490e-ac2a-390e5EXAMPLE"
```

This command produces no output.

For more information, see [Working with Incident Manager incidents in OpsCenter](#) in the *AWS Systems Manager User Guide*.

- For API details, see [DisassociateOpsItemRelatedItem](#) in *AWS CLI Command Reference*.

get-automation-execution

The following code example shows how to use `get-automation-execution`.

AWS CLI

To display details about an automation execution

The following `get-automation-execution` example displays detailed information about an Automation execution.

```
aws ssm get-automation-execution \  
  --automation-execution-id 73c8eef8-f4ee-4a05-820c-e354fEXAMPLE
```

Output:

```
{  
  "AutomationExecution": {  
    "AutomationExecutionId": "73c8eef8-f4ee-4a05-820c-e354fEXAMPLE",  
    "DocumentName": "AWS-StartEC2Instance",  
    "DocumentVersion": "1",  
    "ExecutionStartTime": 1583737233.748,  
    "ExecutionEndTime": 1583737234.719,  
    "AutomationExecutionStatus": "Success",  
    "StepExecutions": [  
      {  
        "StepName": "startInstances",  
        "Action": "aws:changeInstanceState",  
        "ExecutionStartTime": 1583737234.134,  
        "ExecutionEndTime": 1583737234.672,  
      }  
    ]  
  }  
}
```

```

        "StepStatus": "Success",
        "Inputs": {
            "DesiredState": "\"running\"",
            "InstanceIds": "[\"i-0cb99161f6EXAMPLE\"]"
        },
        "Outputs": {
            "InstanceStates": [
                "running"
            ]
        },
        "StepExecutionId": "95e70479-cf20-4d80-8018-7e4e2EXAMPLE",
        "OverriddenParameters": {}
    }
],
"StepExecutionsTruncated": false,
"Parameters": {
    "AutomationAssumeRole": [
        ""
    ],
    "InstanceId": [
        "i-0cb99161f6EXAMPLE"
    ]
},
"Outputs": {},
"Mode": "Auto",
"ExecutedBy": "arn:aws:sts::29884EXAMPLE:assumed-role/mw_service_role/
OrchestrationService",
"Targets": [],
"ResolvedTargets": {
    "ParameterValues": [],
    "Truncated": false
}
}
}

```

For more information, see [Walkthrough: Patch a Linux AMI \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetAutomationExecution](#) in *AWS CLI Command Reference*.

get-calendar-state

The following code example shows how to use `get-calendar-state`.

AWS CLI

Example 1: To get the current state of a change calendar

This `get-calendar-state` example returns the state of a calendar at the current time. Because the example doesn't specify a time, the current state of the calendar is reported.

```
aws ssm get-calendar-state \  
  --calendar-names "MyCalendar"
```

Output:

```
{  
  "State": "OPEN",  
  "AtTime": "2020-02-19T22:28:51Z",  
  "NextTransitionTime": "2020-02-24T21:15:19Z"  
}
```

Example 2: To get the state of a change calendar at a specified time

This `get-calendar-state` example returns the state of a calendar at the specified time.

```
aws ssm get-calendar-state \  
  --calendar-names "MyCalendar" \  
  --at-time "2020-07-19T21:15:19Z"
```

Output:

```
{  
  "State": "CLOSED",  
  "AtTime": "2020-07-19T21:15:19Z"  
}
```

For more information, see [Get the State of the Change Calendar](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetCalendarState](#) in *AWS CLI Command Reference*.

get-command-invocation

The following code example shows how to use `get-command-invocation`.

AWS CLI

To display the details of a command invocation

The following `get-command-invocation` example lists all the invocations of the specified command on the specified instance.

```
aws ssm get-command-invocation \  
  --command-id "ef7fd8-9b57-4151-a15c-db9a12345678" \  
  --instance-id "i-1234567890abcdef0"
```

Output:

```
{  
  "CommandId": "ef7fd8-9b57-4151-a15c-db9a12345678",  
  "InstanceId": "i-1234567890abcdef0",  
  "Comment": "b48291dd-ba76-43e0-b9df-13e11ddaac26:6960febb-2907-4b59-8e1a-d6ce8EXAMPLE",  
  "DocumentName": "AWS-UpdateSSMAgent",  
  "DocumentVersion": "",  
  "PluginName": "aws:updateSsmAgent",  
  "ResponseCode": 0,  
  "ExecutionStartDateTime": "2020-02-19T18:18:03.419Z",  
  "ExecutionElapsedTime": "PT0.091S",  
  "ExecutionEndDateTime": "2020-02-19T18:18:03.419Z",  
  "Status": "Success",  
  "StatusDetails": "Success",  
  "StandardOutputContent": "Updating amazon-ssm-agent from 2.3.842.0 to latest  
\nSuccessfully downloaded https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/  
ssm-agent-manifest.json\namazon-ssm-agent 2.3.842.0 has already been installed,  
update skipped\n",  
  "StandardOutputUrl": "",  
  "StandardErrorContent": "",  
  "StandardErrorUrl": "",  
  "CloudWatchOutputConfig": {  
    "CloudWatchLogGroupName": "",  
    "CloudWatchOutputEnabled": false  
  }  
}
```

For more information, see [Understanding Command Statuses](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetCommandInvocation](#) in *AWS CLI Command Reference*.

get-connection-status

The following code example shows how to use `get-connection-status`.

AWS CLI

To display the connection status of a managed instance

This `get-connection-status` example returns the connection status of the specified managed instance.

```
aws ssm get-connection-status \  
  --target i-1234567890abcdef0
```

Output:

```
{  
  "Target": "i-1234567890abcdef0",  
  "Status": "connected"  
}
```

- For API details, see [GetConnectionStatus](#) in *AWS CLI Command Reference*.

get-default-patch-baseline

The following code example shows how to use `get-default-patch-baseline`.

AWS CLI

Example 1: To display the default Windows patch baseline

The following `get-default-patch-baseline` example retrieves details for the default patch baseline for Windows Server.

```
aws ssm get-default-patch-baseline
```

Output:

```
{
  "BaselineId": "pb-0713accee01612345",
  "OperatingSystem": "WINDOWS"
}
```

Example 2: To display the default patch baseline for Amazon Linux

The following `get-default-patch-baseline` example retrieves details for the default patch baseline for Amazon Linux.

```
aws ssm get-default-patch-baseline \
  --operating-system AMAZON_LINUX
```

Output:

```
{
  "BaselineId": "pb-047c6eb9c8fc12345",
  "OperatingSystem": "AMAZON_LINUX"
}
```

For more information, see [About Predefined and Custom Patch Baselines](https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-baselines.html) <<https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-baselines.html>> and [Set an Existing Patch Baseline as the Default](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetDefaultPatchBaseline](#) in *AWS CLI Command Reference*.

get-deployable-patch-snapshot-for-instance

The following code example shows how to use `get-deployable-patch-snapshot-for-instance`.

AWS CLI

To retrieve the current snapshot for the patch baseline an instance uses

The following `get-deployable-patch-snapshot-for-instance` example retrieves details for the current snapshot for the specified patch baseline used by an instance. This command must be run from the instance using the instance credentials. To ensure it uses the instance credentials, run `aws configure` and specify only the Region of your instance. Leave the `Access Key` and `Secret Key` fields empty.

Tip: Use `uuidgen` to generate a `snapshot-id`.

```
aws ssm get-deployable-patch-snapshot-for-instance \  
  --instance-id "i-1234567890abcdef0" \  
  --snapshot-id "521c3536-930c-4aa9-950e-01234567abcd"
```

Output:

```
{  
  "InstanceId": "i-1234567890abcdef0",  
  "SnapshotId": "521c3536-930c-4aa9-950e-01234567abcd",  
  "Product": "AmazonLinux2018.03",  
  "SnapshotDownloadUrl": "https://patch-baseline-snapshot-us-  
east-1.s3.amazonaws.com/  
ed85194ef27214f5984f28b4d664d14f7313568fea7d4b6ac6c10ad1f729d7e7-773304212436/  
AMAZON_LINUX-521c3536-930c-4aa9-950e-01234567abcd?X-Amz-Algorithm=AWS4-HMAC-  
SHA256&X-Amz-Date=20190215T164031Z&X-Amz-SignedHeaders=host&X-Amz-Expires=86400&X-  
Amz-Credential=AKIAJ5C56P35AEBRX2QQ%2F20190215%2Fus-east-1%2Fs3%2Faws4_request&X-  
Amz-Signature=efaaaf6e3878e77f48a6697e015efdbda9c426b09c5822055075c062f6ad2149"  
}
```

For more information, see [Parameter name: Snapshot ID](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetDeployablePatchSnapshotForInstance](#) in *AWS CLI Command Reference*.

get-document

The following code example shows how to use `get-document`.

AWS CLI

To get document content

The following `get-document` example displays the content of a Systems Manager document.

```
aws ssm get-document \  
  --name "AWS-RunShellScript"
```

Output:

```
{
  "Name": "AWS-RunShellScript",
  "DocumentVersion": "1",
  "Status": "Active",
  "Content": "{\n  \"schemaVersion\": \"1.2\",\n  \"description\": \"Run a shell script or specify the commands to run.\",\n  \"parameters\": {\n    \"commands\": {\n      \"type\": \"StringList\",\n      \"description\": \"(Required) Specify a shell script or a command to run.\",\n      \"minItems\": 1,\n      \"displayType\": \"textarea\",\n      \"workingDirectory\": {\n        \"type\": \"String\",\n        \"default\": \"\",\n        \"description\": \"(Optional) The path to the working directory on your instance.\",\n        \"maxChars\": 4096,\n        \"executionTimeout\": {\n          \"type\": \"String\",\n          \"default\": \"3600\",\n          \"description\": \"(Optional) The time in seconds for a command to complete before it is considered to have failed. Default is 3600 (1 hour). Maximum is 172800 (48 hours).\",\n          \"allowedPattern\": \"([1-9][0-9]{0,4})|(1[0-6][0-9]{4})|(17[0-1][0-9]{3})|(172[0-7][0-9]{2})|(172800)\"\n        }\n      },\n      \"runtimeConfig\": {\n        \"aws:runShellScript\": {\n          \"properties\": [\n            {\n              \"id\": \"0.aws:runShellScript\",\n              \"runCommand\": \"{{ commands }}\",\n              \"workingDirectory\": \"{{ workingDirectory }}\",\n              \"timeoutSeconds\": \"{{ executionTimeout }}\"\n            }\n          ]\n        }\n      }\n    }\n  },\n  \"DocumentType\": \"Command\",\n  \"DocumentFormat\": \"JSON\"\n}"
```

For more information, see [AWS Systems Manager Documents](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetDocument](#) in *AWS CLI Command Reference*.

get-inventory-schema

The following code example shows how to use `get-inventory-schema`.

AWS CLI

To view your inventory schema

This example returns a list of inventory type names for the account.

Command:

```
aws ssm get-inventory-schema
```

Output:

```
{
  "Schemas": [
    {
      "TypeName": "AWS:AWSComponent",
      "Version": "1.0",
      "Attributes": [
        {
          "Name": "Name",
          "DataType": "STRING"
        },
        {
          "Name": "ApplicationType",
          "DataType": "STRING"
        },
        {
          "Name": "Publisher",
          "DataType": "STRING"
        },
        {
          "Name": "Version",
          "DataType": "STRING"
        },
        {
          "Name": "InstalledTime",
          "DataType": "STRING"
        },
        {
          "Name": "Architecture",
          "DataType": "STRING"
        },
        {
          "Name": "URL",
          "DataType": "STRING"
        }
      ]
    },
    ...
  ],
  "NextToken": "--token string truncated--"
}
```

```
}
```

To view the inventory schema for a specific inventory type

This example return the inventory schema for a the AWS:AWSComponent inventory type.

Command:

```
aws ssm get-inventory-schema --type-name "AWS:AWSComponent"
```

- For API details, see [GetInventorySchema](#) in *AWS CLI Command Reference*.

get-inventory

The following code example shows how to use `get-inventory`.

AWS CLI

To view your inventory

This example gets the custom metadata for your inventory.

Command:

```
aws ssm get-inventory
```

Output:

```
{
  "Entities": [
    {
      "Data": {
        "AWS:InstanceInformation": {
          "Content": [
            {
              "ComputerName": "ip-172-31-44-222.us-
west-2.compute.internal",
              "InstanceId": "i-0cb2b964d3e14fd9f",
              "IpAddress": "172.31.44.222",
              "AgentType": "amazon-ssm-agent",
              "ResourceType": "EC2Instance",
              "AgentVersion": "2.0.672.0",
              "PlatformVersion": "2016.09",
```

```

        "PlatformName": "Amazon Linux AMI",
        "PlatformType": "Linux"
      }
    ],
    "TypeName": "AWS:InstanceInformation",
    "SchemaVersion": "1.0",
    "CaptureTime": "2017-02-20T18:03:58Z"
  }
},
  "Id": "i-0cb2b964d3e14fd9f"
}
]
}

```

- For API details, see [GetInventory](#) in *AWS CLI Command Reference*.

get-maintenance-window-execution-task-invocation

The following code example shows how to use `get-maintenance-window-execution-task-invocation`.

AWS CLI

To get information about a maintenance window task invocation

The following `get-maintenance-window-execution-task-invocation` example lists information about the specified task invocation that is part of the specified maintenance window execution.

```

aws ssm get-maintenance-window-execution-task-invocation \
  --window-execution-id "bc494bfa-e63b-49f6-8ad1-aa9f2EXAMPLE" \
  --task-id "96f2ad59-97e3-461d-a63d-40c8aEXAMPLE" \
  --invocation-id "a5273e2c-d2c6-4880-b3e1-5e550EXAMPLE"

```

Output:

```

{
  "Status": "SUCCESS",
  "Parameters": "{\"comment\": \"\", \"documentName\": \"AWS-RunPowerShellScript\", \"instanceIds\": [\"i-1234567890EXAMPLE\"], \"maxConcurrency\": \"1\", \"maxErrors\": \"1\", \"parameters\": {\"executionTimeout\": [\"3600\"], \"workingDirectory\": [\"\"], \"commands\": [\"echo Hello\"]}, \"timeoutSeconds\": 600}"
}

```

```

"ExecutionId": "03b6baa0-5460-4e15-83f2-ea685EXAMPLE",
"InvocationId": "a5273e2c-d2c6-4880-b3e1-5e550EXAMPLE",
"StartTime": 1549998326.421,
"TaskType": "RUN_COMMAND",
"EndTime": 1550001931.784,
"WindowExecutionId": "bc494bfa-e63b-49f6-8ad1-aa9f2EXAMPLE",
"StatusDetails": "Failed",
"TaskExecutionId": "96f2ad59-97e3-461d-a63d-40c8aEXAMPLE"
}

```

For more information, see [View Information About Tasks and Task Executions \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetMaintenanceWindowExecutionTaskInvocation](#) in *AWS CLI Command Reference*.

get-maintenance-window-execution-task

The following code example shows how to use `get-maintenance-window-execution-task`.

AWS CLI

To get information about a maintenance window task execution

The following `get-maintenance-window-execution-task` example lists information about a task that is part of the specified maintenance window execution.

```

aws ssm get-maintenance-window-execution-task \
  --window-execution-id "518d5565-5969-4cca-8f0e-da3b2EXAMPLE" \
  --task-id "ac0c6ae1-daa3-4a89-832e-d3845EXAMPLE"

```

Output:

```

{
  "WindowExecutionId": "518d5565-5969-4cca-8f0e-da3b2EXAMPLE",
  "TaskExecutionId": "ac0c6ae1-daa3-4a89-832e-d3845EXAMPLE",
  "TaskArn": "AWS-RunPatchBaseline",
  "ServiceRole": "arn:aws:iam::111222333444:role/aws-service-role/ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
  "Type": "RUN_COMMAND",
  "TaskParameters": [
    {
      "BaselineOverride": {

```



```
        "Values": [
            ""
        ]
    },
    "InstallOverrideList": {
        "Values": [
            ""
        ]
    },
    "Operation": {
        "Values": [
            "Scan"
        ]
    },
    "RebootOption": {
        "Values": [
            "RebootIfNeeded"
        ]
    },
    "SnapshotId": {
        "Values": [
            "{{ aws:ORCHESTRATION_ID }}"
        ]
    },
    "aws:InstanceId": {
        "Values": [
            "i-02573cafcfEXAMPLE",
            "i-0471e04240EXAMPLE",
            "i-07782c72faEXAMPLE"
        ]
    }
}
],
"Priority": 1,
"MaxConcurrency": "1",
"MaxErrors": "3",
"Status": "SUCCESS",
"StartTime": "2021-08-04T11:45:35.088000-07:00",
"EndTime": "2021-08-04T11:53:09.079000-07:00"
}
```

For more information, see [View information about tasks and task executions \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetMaintenanceWindowExecutionTask](#) in *AWS CLI Command Reference*.

get-maintenance-window-execution

The following code example shows how to use `get-maintenance-window-execution`.

AWS CLI

To get information about a maintenance window task execution

The following `get-maintenance-window-execution` example lists information about a task executed as part of the specified maintenance window execution.

```
aws ssm get-maintenance-window-execution \  
  --window-execution-id "518d5565-5969-4cca-8f0e-da3b2EXAMPLE"
```

Output:

```
{  
  "Status": "SUCCESS",  
  "TaskIds": [  
    "ac0c6ae1-daa3-4a89-832e-d3845EXAMPLE"  
  ],  
  "StartTime": 1487692834.595,  
  "EndTime": 1487692835.051,  
  "WindowExecutionId": "518d5565-5969-4cca-8f0e-da3b2EXAMPLE",  
}
```

For more information, see [View Information About Tasks and Task Executions \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetMaintenanceWindowExecution](#) in *AWS CLI Command Reference*.

get-maintenance-window-task

The following code example shows how to use `get-maintenance-window-task`.

AWS CLI

To get information about a maintenance window task

The following `get-maintenance-window-task` example retrieves details about the specified maintenance window task.

```
aws ssm get-maintenance-window-task \  
  --window-id mw-0c5ed765acEXAMPLE \  
  --window-task-id 0e842a8d-2d44-4886-bb62-af8dcEXAMPLE
```

Output:

```
{  
  "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/  
  ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",  
  "MaxErrors": "1",  
  "TaskArn": "AWS-RunPowerShellScript",  
  "MaxConcurrency": "1",  
  "WindowTaskId": "0e842a8d-2d44-4886-bb62-af8dcEXAMPLE",  
  "TaskParameters": {},  
  "Priority": 1,  
  "TaskInvocationParameters": {  
    "RunCommand": {  
      "Comment": "",  
      "TimeoutSeconds": 600,  
      "Parameters": {  
        "commands": [  
          "echo Hello"  
        ],  
        "executionTimeout": [  
          "3600"  
        ],  
        "workingDirectory": [  
          ""  
        ]  
      }  
    }  
  },  
  "WindowId": "mw-0c5ed765acEXAMPLE",  
  "TaskType": "RUN_COMMAND",  
  "Targets": [  
    {  
      "Values": [  
        "84c818da-b619-4d3d-9651-946f3EXAMPLE"  
      ],  
      "Key": "WindowTargetIds"  
    }  
  ]  
}
```

```
    }
  ],
  "Name": "ExampleTask"
}
```

For more information, see [View Information About Maintenance Windows \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetMaintenanceWindowTask](#) in *AWS CLI Command Reference*.

get-maintenance-window

The following code example shows how to use `get-maintenance-window`.

AWS CLI

To get information about a maintenance window

The following `get-maintenance-window` example retrieves details about the specified maintenance window.

```
aws ssm get-maintenance-window \
  --window-id "mw-03eb9db428EXAMPLE"
```

Output:

```
{
  "AllowUnassociatedTargets": true,
  "CreateDate": 1515006912.957,
  "Cutoff": 1,
  "Duration": 6,
  "Enabled": true,
  "ModifiedDate": 2020-01-01T10:04:04.099Z,
  "Name": "My-Maintenance-Window",
  "Schedule": "rate(3 days)",
  "WindowId": "mw-03eb9db428EXAMPLE",
  "NextExecutionTime": "2020-02-25T00:08:15.099Z"
}
```

For more information, see [View information about maintenance windows \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetMaintenanceWindow](#) in *AWS CLI Command Reference*.

get-ops-item

The following code example shows how to use `get-ops-item`.

AWS CLI

To view information about an OpsItem

The following `get-ops-item` example displays details about the specified OpsItem.

```
aws ssm get-ops-item \
  --ops-item-id oi-0b725EXAMPLE
```

Output:

```
{
  "OpsItem": {
    "CreatedBy": "arn:aws:sts::111222333444:assumed-role/OpsItem-CWE-Role/
fbf77cbe264a33509569f23e4EXAMPLE",
    "CreatedTime": "2019-12-04T15:52:16.793000-08:00",
    "Description": "CloudWatch Event Rule SSMOpsItems-EC2-instance-terminated
was triggered. Your EC2 instance has terminated. See below for more details.",
    "LastModifiedBy": "arn:aws:sts::111222333444:assumed-role/OpsItem-CWE-Role/
fbf77cbe264a33509569f23e4EXAMPLE",
    "LastModifiedTime": "2019-12-04T15:52:16.793000-08:00",
    "Notifications": [],
    "RelatedOpsItems": [],
    "Status": "Open",
    "OpsItemId": "oi-0b725EXAMPLE",
    "Title": "EC2 instance terminated",
    "Source": "EC2",
    "OperationalData": {
      "/aws/automations": {
        "Value": "[ { \"automationType\": \"AWS:SSM:Automation\",
\"automationId\": \"AWS-CreateManagedWindowsInstance\" }, { \"automationType\":
\"AWS:SSM:Automation\", \"automationId\": \"AWS-CreateManagedLinuxInstance\" } ]",
        "Type": "SearchableString"
      },
      "/aws/dedup": {
        "Value": "{\"dedupString\":\"SSMOpsItems-EC2-instance-terminated
\"}",
        "Type": "SearchableString"
      },
      "/aws/resources": {
```

```

        "Value": "[{\"arn\":\"arn:aws:ec2:us-east-2:111222333444:instance/
i-05adec7e97EXAMPLE\"}]",
        "Type": "SearchableString"
    },
    "event-time": {
        "Value": "2019-12-04T23:52:16Z",
        "Type": "String"
    },
    "instance-state": {
        "Value": "terminated",
        "Type": "String"
    }
},
"Category": "Availability",
"Severity": "4"
}
}

```

For more information, see [Working with OpsItems](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetOpsItem](#) in *AWS CLI Command Reference*.

get-ops-summary

The following code example shows how to use `get-ops-summary`.

AWS CLI

To view a summary of all OpsItems

The following `get-ops-summary` example displays a summary of all OpsItems in your AWS account.

```
aws ssm get-ops-summary
```

Output:

```

{
  "Entities": [
    {
      "Id": "oi-4309fEXAMPLE",
      "Data": {
        "AWS:OpsItem": {

```

```

        "CaptureTime": "2020-02-26T18:58:32.918Z",
        "Content": [
            {
                "AccountId": "111222333444",
                "Category": "Availability",
                "CreatedBy": "arn:aws:sts::111222333444:assumed-role/
OpsItem-CWE-Role/fbf77cbe264a33509569f23e4EXAMPLE",
                "CreatedTime": "2020-02-26T19:10:44.149Z",
                "Description": "CloudWatch Event Rule SSM0psItems-EC2-
instance-terminated was triggered. Your EC2 instance has terminated. See below for
more details.",
                "LastModifiedBy": "arn:aws:sts::111222333444:assumed-
role/OpsItem-CWE-Role/fbf77cbe264a33509569f23e4EXAMPLE",
                "LastModifiedTime": "2020-02-26T19:10:44.149Z",
                "Notifications": "",
                "OperationalData": "{\"/aws/automations\":
{\"type\": \"SearchableString\", \"value\": \"[ { \\\"automationType\\\": \\
\"AWS:SSM:Automation\\\", \\\"automationId\\\": \\\"AWS>CreateManagedWindowsInstance
\\\" }, { \\\"automationType\\\": \\\"AWS:SSM:Automation\\\", \\\"automationId
\\\": \\\"AWS>CreateManagedLinuxInstance\\\" } ]\", \"/aws/resources\":
{\"type\": \"SearchableString\", \"value\": \"[{\\\"arn\\\": \\\"arn:aws:ec2:us-
east-2:111222333444:instance/i-0acbd080fEXAMPLE\\\"]\", \"/aws/dedup\": {\"type\":
\"SearchableString\", \"value\": \"{\\\"dedupString\\\": \\\"SSM0psItems-EC2-instance-
terminated\\\"}\"}}",
                "OpsItemId": "oi-4309fEXAMPLE",
                "RelatedItems": "",
                "Severity": "3",
                "Source": "EC2",
                "Status": "Open",
                "Title": "EC2 instance terminated"
            }
        ]
    },
    {
        "Id": "oi-bb2a0e6a4541",
        "Data": {
            "AWS:OpsItem": {
                "CaptureTime": "2019-11-26T19:20:06.161Z",
                "Content": [
                    {
                        "AccountId": "111222333444",
                        "Category": "Availability",

```

```

        "CreatedBy": "arn:aws:sts::111222333444:assumed-role/
OpsItem-CWE-Role/fbf77cbe264a33509569f23e4EXAMPLE",
        "CreatedTime": "2019-11-26T20:00:07.237Z",
        "Description": "CloudWatch Event Rule SSM0psItems-SSM-
maintenance-window-execution-failed was triggered. Your SSM Maintenance Window
execution has failed. See below for more details.",
        "LastModifiedBy": "arn:aws:sts::111222333444:assumed-
role/OpsItem-CWE-Role/fbf77cbe264a33509569f23e4EXAMPLE",
        "LastModifiedTime": "2019-11-26T20:00:07.237Z",
        "Notifications": "",
        "OperationalData": "{\"/aws/resources\":{\"type
\": \"SearchableString\", \"value\": \"[\\\"arn\\\": \\\"arn:aws:ssm:us-
east-2:111222333444:maintenancewindow/mw-0e83ba440dEXAMPLE\\\"]\"}, \"/aws/dedup\":
{ \"type\": \"SearchableString\", \"value\": \"{\\\"dedupString\\\": \\\"SSM0psItems-SSM-
maintenance-window-execution-failed\\\"}\"}}",
        "OpsItemId": "oi-bb2a0EXAMPLE",
        "RelatedItems": "",
        "Severity": "3",
        "Source": "SSM",
        "Status": "Open",
        "Title": "SSM Maintenance Window execution failed"
    }
  ]
}

```

For more information, see [Working with OpsItems](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetOpsSummary](#) in *AWS CLI Command Reference*.

get-parameter-history

The following code example shows how to use `get-parameter-history`.

AWS CLI

To get a value history for a parameter

The following `get-parameter-history` example lists the history of changes for the specified parameter, including its value.


```
aws ssm get-parameter-history \  
  --name "MyStringParameter"
```

Output:

```
{  
  "Parameters": [  
    {  
      "Name": "MyStringParameter",  
      "Type": "String",  
      "LastModifiedDate": 1582154711.976,  
      "LastModifiedUser": "arn:aws:iam::111222333444:user/Mary-Major",  
      "Description": "This is the first version of my String parameter",  
      "Value": "Veni",  
      "Version": 1,  
      "Labels": [],  
      "Tier": "Standard",  
      "Policies": []  
    },  
    {  
      "Name": "MyStringParameter",  
      "Type": "String",  
      "LastModifiedDate": 1582156093.471,  
      "LastModifiedUser": "arn:aws:iam::111222333444:user/Mary-Major",  
      "Description": "This is the second version of my String parameter",  
      "Value": "Vidi",  
      "Version": 2,  
      "Labels": [],  
      "Tier": "Standard",  
      "Policies": []  
    },  
    {  
      "Name": "MyStringParameter",  
      "Type": "String",  
      "LastModifiedDate": 1582156117.545,  
      "LastModifiedUser": "arn:aws:iam::111222333444:user/Mary-Major",  
      "Description": "This is the third version of my String parameter",  
      "Value": "Vici",  
      "Version": 3,  
      "Labels": [],  
      "Tier": "Standard",  
      "Policies": []  
    }  
  ]  
}
```

```
]
}
```

For more information, see [Working with parameter versions](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetParameterHistory](#) in *AWS CLI Command Reference*.

get-parameter

The following code example shows how to use `get-parameter`.

AWS CLI

Example 1: To display the value of a parameter

The following `get-parameter` example lists the value for the specified single parameter.

```
aws ssm get-parameter \
  --name "MyStringParameter"
```

Output:

```
{
  "Parameter": {
    "Name": "MyStringParameter",
    "Type": "String",
    "Value": "Veni",
    "Version": 1,
    "LastModifiedDate": 1530018761.888,
    "ARN": "arn:aws:ssm:us-east-2:111222333444:parameter/MyStringParameter"
    "DataType": "text"
  }
}
```

For more information, see [Working with Parameter Store](#) in the *AWS Systems Manager User Guide*.

Example 2: To decrypt the value of a SecureString parameter

The following `get-parameter` example decrypts the value of the specified `SecureString` parameter.

```
aws ssm get-parameter \  
  --name "MySecureStringParameter" \  
  --with-decryption
```

Output:

```
{  
  "Parameter": {  
    "Name": "MySecureStringParameter",  
    "Type": "SecureString",  
    "Value": "16679b88-310b-4895-a943-e0764EXAMPLE",  
    "Version": 2,  
    "LastModifiedDate": 1582155479.205,  
    "ARN": "arn:aws:ssm:us-east-2:111222333444:parameter/  
MySecureStringParameter"  
    "DataType": "text"  
  }  
}
```

For more information, see [Working with Parameter Store](#) in the *AWS Systems Manager User Guide*.

Example 3: To display the value of a parameter using labels

The following `get-parameter` example lists the value for the specified single parameter with a specified label.

```
aws ssm get-parameter \  
  --name "MyParameter:label"
```

Output:

```
{  
  "Parameter": {  
    "Name": "MyParameter",  
    "Type": "String",  
    "Value": "parameter version 2",  
    "Version": 2,  
    "Selector": ":label",  
    "LastModifiedDate": "2021-07-12T09:49:15.865000-07:00",  
    "ARN": "arn:aws:ssm:us-west-2:786973925828:parameter/MyParameter",
```

```
    "DataType": "text"
  }
}
```

For more information, see [Working with parameter labels](#) in the *AWS Systems Manager User Guide*.

Example 4: To display the value of a parameter using versions

The following `get-parameter` example lists the value for the specified single parameter version.

```
aws ssm get-parameter \
  --name "MyParameter:2"
```

Output:

```
{
  "Parameter": {
    "Name": "MyParameter",
    "Type": "String",
    "Value": "parameter version 2",
    "Version": 2,
    "Selector": ":2",
    "LastModifiedDate": "2021-07-12T09:49:15.865000-07:00",
    "ARN": "arn:aws:ssm:us-west-2:786973925828:parameter/MyParameter",
    "DataType": "text"
  }
}
```

For more information, see [Working with parameter labels](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetParameter](#) in *AWS CLI Command Reference*.

get-parameters-by-path

The following code example shows how to use `get-parameters-by-path`.

AWS CLI

To list parameters in a specific path

The following `get-parameters-by-path` example lists the parameters within the specified hierarchy.

```
aws ssm get-parameters-by-path \  
  --path "/site/newyork/department/"
```

Output:

```
{  
  "Parameters": [  
    {  
      "Name": "/site/newyork/department/marketing",  
      "Type": "String",  
      "Value": "Floor 2",  
      "Version": 1,  
      "LastModifiedDate": 1530018761.888,  
      "ARN": "arn:aws:ssm:us-east-1:111222333444:parameter/site/newyork/  
department/marketing"  
    },  
    {  
      "Name": "/site/newyork/department/infotech",  
      "Type": "String",  
      "Value": "Floor 3",  
      "Version": 1,  
      "LastModifiedDate": 1530018823.429,  
      "ARN": "arn:aws:ssm:us-east-1:111222333444:parameter/site/newyork/  
department/infotech"  
    },  
    ...  
  ]  
}
```

For more information, see [Working with parameter hierarchies](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetParametersByPath](#) in *AWS CLI Command Reference*.

get-parameters

The following code example shows how to use `get-parameters`.

AWS CLI

Example 1: To list the values for a parameter

The following `get-parameters` example lists the values for the three specified parameters.

```
aws ssm get-parameters \  
  --names "MyStringParameter" "MyStringListParameter" "MyInvalidParameterName"
```

Output:

```
{  
  "Parameters": [  
    {  
      "Name": "MyStringListParameter",  
      "Type": "StringList",  
      "Value": "alpha,beta,gamma",  
      "Version": 1,  
      "LastModifiedDate": 1582154764.222,  
      "ARN": "arn:aws:ssm:us-east-2:111222333444:parameter/  
MyStringListParameter"  
      "DataType": "text"  
    },  
    {  
      "Name": "MyStringParameter",  
      "Type": "String",  
      "Value": "Vici",  
      "Version": 3,  
      "LastModifiedDate": 1582156117.545,  
      "ARN": "arn:aws:ssm:us-east-2:111222333444:parameter/MyStringParameter"  
      "DataType": "text"  
    }  
  ],  
  "InvalidParameters": [  
    "MyInvalidParameterName"  
  ]  
}
```

For more information, see [Working with Parameter Store](#) in the *AWS Systems Manager User Guide*.

Example 2: To list names and values of multiple parameters using the `--query` option

The following `get-parameters` example lists the names and values for the specified parameters.

```
aws ssm get-parameters \  
  --names MyStringParameter MyStringListParameter \  
  --query "Parameters[*].{Name:Name,Value:Value}"
```

Output:

```
[  
  {  
    "Name": "MyStringListParameter",  
    "Value": "alpha,beta,gamma"  
  },  
  {  
    "Name": "MyStringParameter",  
    "Value": "Vidi"  
  }  
]
```

For more information, see [Working with Parameter Store](#) in the *AWS Systems Manager User Guide*.

Example 3: To display the value of a parameter using labels

The following `get-parameter` example lists the value for the specified single parameter with a specified label.

```
aws ssm get-parameter \  
  --name "MyParameter:label"
```

Output:

```
{  
  "Parameters": [  
    {  
      "Name": "MyLabelParameter",  
      "Type": "String",  
      "Value": "parameter by label",  
      "Version": 1,  
      "Selector": ":label",  
      "LastModifiedDate": "2021-07-12T09:49:15.865000-07:00",
```

```
    "ARN": "arn:aws:ssm:us-west-2:786973925828:parameter/MyParameter",
    "DataType": "text"
  },
  {
    "Name": "MyVersionParameter",
    "Type": "String",
    "Value": "parameter by version",
    "Version": 2,
    "Selector": ":2",
    "LastModifiedDate": "2021-03-24T16:20:28.236000-07:00",
    "ARN": "arn:aws:ssm:us-west-2:786973925828:parameter/unlabel-param",
    "DataType": "text"
  }
],
"InvalidParameters": []
}
```

For more information, see [Working with parameter labels](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetParameters](#) in *AWS CLI Command Reference*.

get-patch-baseline-for-patch-group

The following code example shows how to use `get-patch-baseline-for-patch-group`.

AWS CLI

To display the patch baseline for a patch group

The following `get-patch-baseline-for-patch-group` example retrieves details about the patch baseline for the specified patch group.

```
aws ssm get-patch-baseline-for-patch-group \
  --patch-group "DEV"
```

Output:

```
{
  "PatchGroup": "DEV",
  "BaselineId": "pb-0123456789abcdef0",
  "OperatingSystem": "WINDOWS"
```



```
}
```

For more information, see [Create a Patch Group <https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-group-tagging.html>](https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-group-tagging.html) and [Add a Patch Group to a Patch Baseline](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetPatchBaselineForPatchGroup](#) in *AWS CLI Command Reference*.

get-patch-baseline

The following code example shows how to use `get-patch-baseline`.

AWS CLI

To display a patch baseline

The following `get-patch-baseline` example retrieves the details for the specified patch baseline.

```
aws ssm get-patch-baseline \  
  --baseline-id "pb-0123456789abcdef0"
```

Output:

```
{  
  "BaselineId": "pb-0123456789abcdef0",  
  "Name": "WindowsPatching",  
  "OperatingSystem": "WINDOWS",  
  "GlobalFilters": {  
    "PatchFilters": []  
  },  
  "ApprovalRules": {  
    "PatchRules": [  
      {  
        "PatchFilterGroup": {  
          "PatchFilters": [  
            {  
              "Key": "PRODUCT",  
              "Values": [  
                "WindowsServer2016"  
              ]  
            }  
          ]  
        }  
      }  
    ]  
  }  
}
```

```

        ]
        },
        "ComplianceLevel": "CRITICAL",
        "ApproveAfterDays": 0,
        "EnableNonSecurity": false
    }
]
},
"ApprovedPatches": [],
"ApprovedPatchesComplianceLevel": "UNSPECIFIED",
"ApprovedPatchesEnableNonSecurity": false,
"RejectedPatches": [],
"RejectedPatchesAction": "ALLOW_AS_DEPENDENCY",
"PatchGroups": [
    "QA",
    "DEV"
],
"CreateDate": 1550244180.465,
"ModifiedDate": 1550244180.465,
"Description": "Patches for Windows Servers",
"Sources": []
}

```

For more information, see [About Patch Baselines](#) in the *AWS Systems Manager User Guide*.

- For API details, see [GetPatchBaseline](#) in *AWS CLI Command Reference*.

get-service-setting

The following code example shows how to use `get-service-setting`.

AWS CLI

To retrieve the service setting for Parameter Store throughput

The following `get-service-setting` example retrieves the current service setting for Parameter Store throughput in the specified region.

```

aws ssm get-service-setting \
  --setting-id arn:aws:ssm:us-east-1:123456789012:servicesetting/ssm/parameter-
  store/high-throughput-enabled

```

Output:

```
{
  "ServiceSetting": {
    "SettingId": "/ssm/parameter-store/high-throughput-enabled",
    "SettingValue": "false",
    "LastModifiedDate": 1555532818.578,
    "LastModifiedUser": "System",
    "ARN": "arn:aws:ssm:us-east-1:123456789012:servicesetting/ssm/parameter-
store/high-throughput-enabled",
    "Status": "Default"
  }
}
```

For more information, see [Increasing Parameter Store Throughput](#) in the *AWS Systems Manager Users Guide*.

- For API details, see [GetServiceSetting](#) in *AWS CLI Command Reference*.

label-parameter-version

The following code example shows how to use `label-parameter-version`.

AWS CLI

Example 1: To add a label to latest version of a parameter

The following `label-parameter-version` example adds a label to the latest version of the specified parameter.

```
aws ssm label-parameter-version \
  --name "MyStringParameter" \
  --labels "ProductionReady"
```

Output:

```
{
  "InvalidLabels": [],
  "ParameterVersion": 3
}
```

For more information, see [Working with parameter labels](#) in the *AWS Systems Manager User Guide*.

Example 2: To add a label to a specific version of a parameter

The following `label-parameter-version` example adds a label to the specified version of a parameter.

```
aws ssm label-parameter-version \  
  --name "MyStringParameter" \  
  --labels "ProductionReady" \  
  --parameter-version "2" --labels "DevelopmentReady"
```

For more information, see [Working with parameter labels](#) in the *AWS Systems Manager User Guide*.

- For API details, see [LabelParameterVersion](#) in *AWS CLI Command Reference*.

list-association-versions

The following code example shows how to use `list-association-versions`.

AWS CLI

To list all versions of an association for a specific association ID

The following `list-association-versions` example lists all versions of the specified associations.

```
aws ssm list-association-versions \  
  --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

Output:

```
{  
  "AssociationVersions": [  
    {  
      "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",  
      "AssociationVersion": "1",  
      "CreateDate": 1550505536.726,  
      "Name": "AWS-UpdateSSMAgent",  
      "Parameters": {  
        "allowDowngrade": [  
          "false"  
        ],  
      },  
    },  
  ],  
}
```

```
        "version": [
            ""
        ],
    },
    "Targets": [
        {
            "Key": "InstanceIds",
            "Values": [
                "i-1234567890abcdef0"
            ]
        }
    ],
    "ScheduleExpression": "cron(0 00 12 ? * SUN *)",
    "AssociationName": "UpdateSSMAgent"
}
]
```

For more information, see [Working with associations in Systems Manager](#) in the *AWS Systems Manager User Guide*.

- For API details, see [ListAssociationVersions](#) in *AWS CLI Command Reference*.

list-associations

The following code example shows how to use `list-associations`.

AWS CLI

Example 1: To list your associations for a specific instance

The following `list-associations` example lists all associations with the `AssociationName`, `UpdateSSMAgent`.

```
aws ssm list-associations /
  --association-filter-list "key=AssociationName,value=UpdateSSMAgent"
```

Output:

```
{
  "Associations": [
    {
      "Name": "AWS-UpdateSSMAgent",
```

```

    "InstanceId": "i-1234567890abcdef0",
    "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
    "AssociationVersion": "1",
    "Targets": [
      {
        "Key": "InstanceIds",
        "Values": [
          "i-016648b75dd622dab"
        ]
      }
    ],
    "Overview": {
      "Status": "Pending",
      "DetailedStatus": "Associated",
      "AssociationStatusAggregatedCount": {
        "Pending": 1
      }
    },
    "ScheduleExpression": "cron(0 00 12 ? * SUN *)",
    "AssociationName": "UpdateSSMAgent"
  }
]
}

```

For more information, see [Working with associations in Systems Manager](#) in the *Systems Manager User Guide*.

Example 2: To list your associations for a specific document

The following list-associations example lists all associations for the specified document.

```

aws ssm list-associations /
  --association-filter-list "key=Name,value=AWS-UpdateSSMAgent"

```

Output:

```

{
  "Associations": [
    {
      "Name": "AWS-UpdateSSMAgent",
      "InstanceId": "i-1234567890abcdef0",
      "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
      "AssociationVersion": "1",

```

```
    "Targets": [
      {
        "Key": "InstanceIds",
        "Values": [
          "i-1234567890abcdef0"
        ]
      }
    ],
    "LastExecutionDate": 1550505828.548,
    "Overview": {
      "Status": "Success",
      "DetailedStatus": "Success",
      "AssociationStatusAggregatedCount": {
        "Success": 1
      }
    },
    "ScheduleExpression": "cron(0 00 12 ? * SUN *)",
    "AssociationName": "UpdateSSMAgent"
  },
  {
    "Name": "AWS-UpdateSSMAgent",
    "InstanceId": "i-9876543210abcdef0",
    "AssociationId": "fbc07ef7-b985-4684-b82b-0123456789ab",
    "AssociationVersion": "1",
    "Targets": [
      {
        "Key": "InstanceIds",
        "Values": [
          "i-9876543210abcdef0"
        ]
      }
    ],
    "LastExecutionDate": 1550507531.0,
    "Overview": {
      "Status": "Success",
      "AssociationStatusAggregatedCount": {
        "Success": 1
      }
    }
  }
]
}
```

For more information, see [Working with associations in Systems Manager](#) in the *Systems Manager User Guide*.

- For API details, see [ListAssociations](#) in *AWS CLI Command Reference*.

list-command-invocations

The following code example shows how to use `list-command-invocations`.

AWS CLI

To list the invocations of a specific command

The following `list-command-invocations` example lists all the invocations of a command.

```
aws ssm list-command-invocations \
  --command-id "ef7fd8-9b57-4151-a15c-db9a12345678" \
  --details
```

Output:

```
{
  "CommandInvocations": [
    {
      "CommandId": "ef7fd8-9b57-4151-a15c-db9a12345678",
      "InstanceId": "i-02573cafcfEXAMPLE",
      "InstanceName": "",
      "Comment": "b48291dd-ba76-43e0-
b9df-13e11ddaac26:6960febb-2907-4b59-8e1a-d6ce8EXAMPLE",
      "DocumentName": "AWS-UpdateSSMAgent",
      "DocumentVersion": "",
      "RequestedDateTime": 1582136283.089,
      "Status": "Success",
      "StatusDetails": "Success",
      "StandardOutputUrl": "",
      "StandardErrorUrl": "",
      "CommandPlugins": [
        {
          "Name": "aws:updateSsmAgent",
          "Status": "Success",
          "StatusDetails": "Success",
          "ResponseCode": 0,
          "ResponseStartDateTime": 1582136283.419,
```



```

        "ResponseFinishDateTime": 1582136283.51,
        "Output": "Updating amazon-ssm-agent from 2.3.842.0 to latest
\nSuccessfully downloaded https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/
ssm-agent-manifest.json\namazon-ssm-agent 2.3.842.0 has already been installed,
update skipped\n",
        "StandardOutputUrl": "",
        "StandardErrorUrl": "",
        "OutputS3Region": "us-east-2",
        "OutputS3BucketName": "",
        "OutputS3KeyPrefix": ""
    }
],
"ServiceRole": "",
"NotificationConfig": {
    "NotificationArn": "",
    "NotificationEvents": [],
    "NotificationType": ""
},
"CloudWatchOutputConfig": {
    "CloudWatchLogGroupName": "",
    "CloudWatchOutputEnabled": false
}
},
{
    "CommandId": "ef7fd8-9b57-4151-a15c-db9a12345678",
    "InstanceId": "i-0471e04240EXAMPLE",
    "InstanceName": "",
    "Comment": "b48291dd-ba76-43e0-
b9df-13e11ddaac26:6960febb-2907-4b59-8e1a-d6ce8EXAMPLE",
    "DocumentName": "AWS-UpdateSSMAgent",
    "DocumentVersion": "",
    "RequestedDateTime": 1582136283.02,
    "Status": "Success",
    "StatusDetails": "Success",
    "StandardOutputUrl": "",
    "StandardErrorUrl": "",
    "CommandPlugins": [
        {
            "Name": "aws:updateSsmAgent",
            "Status": "Success",
            "StatusDetails": "Success",
            "ResponseCode": 0,
            "ResponseStartDateTime": 1582136283.812,
            "ResponseFinishDateTime": 1582136295.031,

```

```

        "Output": "Updating amazon-ssm-agent from 2.3.672.0 to latest
\nSuccessfully downloaded https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/
ssm-agent-manifest.json\nSuccessfully downloaded https://s3.us-east-2.amazonaws.com/
amazon-ssm-us-east-2/amazon-ssm-agent-updater/2.3.842.0/amazon-ssm-agent-updater-
snap-amd64.tar.gz\nSuccessfully downloaded https://s3.us-east-2.amazonaws.com/
amazon-ssm-us-east-2/amazon-ssm-agent/2.3.672.0/amazon-ssm-agent-snap-amd64.tar.gz
\nSuccessfully downloaded https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/
amazon-ssm-agent/2.3.842.0/amazon-ssm-agent-snap-amd64.tar.gz\nInitiating amazon-
ssm-agent update to 2.3.842.0\namazon-ssm-agent updated successfully to 2.3.842.0",
        "StandardOutputUrl": "",
        "StandardErrorUrl": "",
        "OutputS3Region": "us-east-2",
        "OutputS3BucketName": "",
        "OutputS3KeyPrefix": "8bee3135-398c-4d31-99b6-e42d2EXAMPLE/
i-0471e04240EXAMPLE/awsupdateSsmAgent"
    }
  ],
  "ServiceRole": "",
  "NotificationConfig": {
    "NotificationArn": "",
    "NotificationEvents": [],
    "NotificationType": ""
  },
  "CloudWatchOutputConfig": {
    "CloudWatchLogGroupName": "",
    "CloudWatchOutputEnabled": false
  }
}
]
}

```

For more information, see [Understanding Command Statuses](#) in the *AWS Systems Manager User Guide*.

- For API details, see [ListCommandInvocations](#) in *AWS CLI Command Reference*.

list-commands

The following code example shows how to use `list-commands`.

AWS CLI

Example 1: To get the status of a specific command

The following `list-commands` example retrieves and displays the status of the specified command.

```
aws ssm list-commands \  
  --command-id "0831e1a8-a1ac-4257-a1fd-c831bEXAMPLE"
```

Example 2: To get the status of commands requested after a specific date

The following `list-commands` example retrieves the details of commands requested after the specified date.

```
aws ssm list-commands \  
  --filter "key=InvokedAfter,value=2020-02-01T00:00:00Z"
```

Example 3: To list all commands requested in an AWS account

The following `list-commands` example lists all commands requested by users in the current AWS account and Region.

```
aws ssm list-commands
```

Output:

```
{  
  "Commands": [  
    {  
      "CommandId": "8bee3135-398c-4d31-99b6-e42d2EXAMPLE",  
      "DocumentName": "AWS-UpdateSSMAgent",  
      "DocumentVersion": "",  
      "Comment": "b48291dd-ba76-43e0-  
b9df-13e11ddaac26:6960febb-2907-4b59-8e1a-d6ce8EXAMPLE",  
      "ExpiresAfter": "2020-02-19T11:28:02.500000-08:00",  
      "Parameters": {},  
      "InstanceIds": [  
        "i-028ea792daEXAMPLE",  
        "i-02feef8c46EXAMPLE",  
        "i-038613f3f0EXAMPLE",  
        "i-03a530a2d4EXAMPLE",  
        "i-083b678d37EXAMPLE",  
        "i-0dee81debaEXAMPLE"  
      ],  
      "Targets": [],  
    }  
  ]  
}
```

```

    "RequestedDateTime": "2020-02-19T10:18:02.500000-08:00",
    "Status": "Success",
    "StatusDetails": "Success",
    "OutputS3BucketName": "",
    "OutputS3KeyPrefix": "",
    "MaxConcurrency": "50",
    "MaxErrors": "100%",
    "TargetCount": 6,
    "CompletedCount": 6,
    "ErrorCount": 0,
    "DeliveryTimedOutCount": 0,
    "ServiceRole": "",
    "NotificationConfig": {
      "NotificationArn": "",
      "NotificationEvents": [],
      "NotificationType": ""
    },
    "CloudWatchOutputConfig": {
      "CloudWatchLogGroupName": "",
      "CloudWatchOutputEnabled": false
    }
  }
}
{
  "CommandId": "e9ade581-c03d-476b-9b07-26667EXAMPLE",
  "DocumentName": "AWS-FindWindowsUpdates",
  "DocumentVersion": "1",
  "Comment": "",
  "ExpiresAfter": "2020-01-24T12:37:31.874000-08:00",
  "Parameters": {
    "KbArticleIds": [
      ""
    ],
    "UpdateLevel": [
      "All"
    ]
  },
  "InstanceIds": [],
  "Targets": [
    {
      "Key": "InstanceIds",
      "Values": [
        "i-00ec29b21eEXAMPLE",
        "i-09911ddd90EXAMPLE"
      ]
    }
  ]
}

```

```

    }
  ],
  "RequestedDateTime": "2020-01-24T11:27:31.874000-08:00",
  "Status": "Success",
  "StatusDetails": "Success",
  "OutputS3BucketName": "my-us-east-2-bucket",
  "OutputS3KeyPrefix": "my-rc-output",
  "MaxConcurrency": "50",
  "MaxErrors": "0",
  "TargetCount": 2,
  "CompletedCount": 2,
  "ErrorCount": 0,
  "DeliveryTimedOutCount": 0,
  "ServiceRole": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
  "NotificationConfig": {
    "NotificationArn": "arn:aws:sns:us-east-2:111222333444:my-us-east-2-
notification-arn",
    "NotificationEvents": [
      "All"
    ],
    "NotificationType": "Invocation"
  },
  "CloudWatchOutputConfig": {
    "CloudWatchLogGroupName": "",
    "CloudWatchOutputEnabled": false
  }
}
{
  "CommandId": "d539b6c3-70e8-4853-80e5-0ce4fEXAMPLE",
  "DocumentName": "AWS-RunPatchBaseline",
  "DocumentVersion": "1",
  "Comment": "",
  "ExpiresAfter": "2020-01-24T12:21:04.350000-08:00",
  "Parameters": {
    "InstallOverrideList": [
      ""
    ],
    "Operation": [
      "Install"
    ],
    "RebootOption": [
      "RebootIfNeeded"
    ]
  },

```

```
        "SnapshotId": [
            ""
        ]
    },
    "InstanceIds": [],
    "Targets": [
        {
            "Key": "InstanceIds",
            "Values": [
                "i-00ec29b21eEXAMPLE",
                "i-09911ddd90EXAMPLE"
            ]
        }
    ],
    "RequestedDateTime": "2020-01-24T11:11:04.350000-08:00",
    "Status": "Success",
    "StatusDetails": "Success",
    "OutputS3BucketName": "my-us-east-2-bucket",
    "OutputS3KeyPrefix": "my-rc-output",
    "MaxConcurrency": "50",
    "MaxErrors": "0",
    "TargetCount": 2,
    "CompletedCount": 2,
    "ErrorCount": 0,
    "DeliveryTimedOutCount": 0,
    "ServiceRole": "arn:aws:iam::111222333444:role/aws-service-role/
    ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
    "NotificationConfig": {
        "NotificationArn": "arn:aws:sns:us-east-2:111222333444:my-us-east-2-
    notification-arn",
        "NotificationEvents": [
            "All"
        ],
        "NotificationType": "Invocation"
    },
    "CloudWatchOutputConfig": {
        "CloudWatchLogGroupName": "",
        "CloudWatchOutputEnabled": false
    }
}
]
```

For more information, see [Running Commands Using Systems Manager Run Command](#) in the *AWS Systems Manager User Guide*.

- For API details, see [ListCommands](#) in *AWS CLI Command Reference*.

list-compliance-items

The following code example shows how to use `list-compliance-items`.

AWS CLI

To list compliance items for a specific instance

This example lists all compliance items for the specified instance.

Command:

```
aws ssm list-compliance-items --resource-ids "i-1234567890abcdef0" --resource-types "ManagedInstance"
```

Output:

```
{
  "ComplianceItems": [
    {
      "ComplianceType": "Association",
      "ResourceType": "ManagedInstance",
      "ResourceId": "i-1234567890abcdef0",
      "Id": "8dfe3659-4309-493a-8755-0123456789ab",
      "Title": "",
      "Status": "COMPLIANT",
      "Severity": "UNSPECIFIED",
      "ExecutionSummary": {
        "ExecutionTime": 1550408470.0
      },
      "Details": {
        "DocumentName": "AWS-GatherSoftwareInventory",
        "DocumentVersion": "1"
      }
    },
    {
      "ComplianceType": "Association",
      "ResourceType": "ManagedInstance",
```

```

    "ResourceId": "i-1234567890abcdef0",
    "Id": "e4c2ed6d-516f-41aa-aa2a-0123456789ab",
    "Title": "",
    "Status": "COMPLIANT",
    "Severity": "UNSPECIFIED",
    "ExecutionSummary": {
      "ExecutionTime": 1550508475.0
    },
    "Details": {
      "DocumentName": "AWS-UpdateSSMAgent",
      "DocumentVersion": "1"
    }
  },
  ...
],
"NextToken": "--token string truncated--"
}

```

To list compliance items for a specific instance and association ID

This example lists all compliance items for the specified instance and association ID.

Command:

```

aws ssm list-compliance-items --resource-ids "i-1234567890abcdef0" --resource-types
"ManagedInstance" --filters "Key=ComplianceType,Values=Association,Type=EQUAL"
"Key=Id,Values=e4c2ed6d-516f-41aa-aa2a-0123456789ab,Type=EQUAL"

```

To list compliance items for a instance after a specific date and time

This example lists all compliance items for an instance after the specified date and time.

Command:

```

aws ssm list-compliance-items --resource-ids "i-1234567890abcdef0" --resource-types
"ManagedInstance" --filters
"Key=ExecutionTime,Values=2019-02-18T16:00:00Z,Type=GREATER_THAN"

```

- For API details, see [ListComplianceItems](#) in *AWS CLI Command Reference*.

list-compliance-summaries

The following code example shows how to use `list-compliance-summaries`.

AWS CLI

To list compliance summaries for all compliance types

This example lists compliance summaries for all compliance types in your account.

Command:

```
aws ssm list-compliance-summaries
```

Output:

```
{
  "ComplianceSummaryItems": [
    {
      "ComplianceType": "Association",
      "CompliantSummary": {
        "CompliantCount": 2,
        "SeveritySummary": {
          "CriticalCount": 0,
          "HighCount": 0,
          "MediumCount": 0,
          "LowCount": 0,
          "InformationalCount": 0,
          "UnspecifiedCount": 2
        }
      },
      "NonCompliantSummary": {
        "NonCompliantCount": 0,
        "SeveritySummary": {
          "CriticalCount": 0,
          "HighCount": 0,
          "MediumCount": 0,
          "LowCount": 0,
          "InformationalCount": 0,
          "UnspecifiedCount": 0
        }
      }
    },
    {
      "ComplianceType": "Patch",
      "CompliantSummary": {
        "CompliantCount": 1,
        "SeveritySummary": {
```

```

        "CriticalCount": 0,
        "HighCount": 0,
        "MediumCount": 0,
        "LowCount": 0,
        "InformationalCount": 0,
        "UnspecifiedCount": 1
      }
    },
    "NonCompliantSummary": {
      "NonCompliantCount": 1,
      "SeveritySummary": {
        "CriticalCount": 1,
        "HighCount": 0,
        "MediumCount": 0,
        "LowCount": 0,
        "InformationalCount": 0,
        "UnspecifiedCount": 0
      }
    }
  },
  ...
],
"NextToken": "eyJ0ZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAyfQ=="
}

```

To list compliance summaries for a specific compliance type

This example lists the compliance summary for the Patch compliance type.

Command:

```
aws ssm list-compliance-summaries --filters
"Key=ComplianceType,Values=Patch,Type=EQUAL"
```

- For API details, see [ListComplianceSummaries](#) in *AWS CLI Command Reference*.

list-document-metadata-history

The following code example shows how to use `list-document-metadata-history`.

AWS CLI

Example: To view approval history and status for a change template

The following `list-document-metadata-history` example returns the approval history for the specified Change Manager change template.

```
aws ssm list-document-metadata-history \  
  --name MyChangeManageTemplate \  
  --metadata DocumentReviews
```

Output:

```
{  
  "Name": "MyChangeManagerTemplate",  
  "DocumentVersion": "1",  
  "Author": "arn:aws:iam::111222333444:user/JohnDoe",  
  "Metadata": {  
    "ReviewerResponse": [  
      {  
        "CreateTime": "2021-07-30T11:58:28.025000-07:00",  
        "UpdateTime": "2021-07-30T12:01:19.274000-07:00",  
        "ReviewStatus": "APPROVED",  
        "Comment": [  
          {  
            "Type": "COMMENT",  
            "Content": "I approve this template version"  
          }  
        ],  
        "Reviewer": "arn:aws:iam::111222333444:user/ShirleyRodriguez"  
      },  
      {  
        "CreateTime": "2021-07-30T11:58:28.025000-07:00",  
        "UpdateTime": "2021-07-30T11:58:28.025000-07:00",  
        "ReviewStatus": "PENDING"  
      }  
    ]  
  }  
}
```

For more information, see [Reviewing and approving or rejecting change templates](#) in the *AWS Systems Manager User Guide*.

- For API details, see [ListDocumentMetadataHistory](#) in *AWS CLI Command Reference*.

list-document-versions

The following code example shows how to use `list-document-versions`.

AWS CLI

To list document versions

The following `list-document-versions` example lists all versions for a Systems Manager document.

```
aws ssm list-document-versions \  
  --name "Example"
```

Output:

```
{  
  "DocumentVersions": [  
    {  
      "Name": "Example",  
      "DocumentVersion": "1",  
      "CreateDate": 1583257938.266,  
      "IsDefaultVersion": true,  
      "DocumentFormat": "YAML",  
      "Status": "Active"  
    }  
  ]  
}
```

For more information, see [Sending Commands that Use the Document Version Parameter](#) in the *AWS Systems Manager User Guide*.

- For API details, see [ListDocumentVersions](#) in *AWS CLI Command Reference*.

list-documents

The following code example shows how to use `list-documents`.

AWS CLI

Example 1: To list documents

The following `list-documents` example lists documents owned by the requesting account tagged with the custom tag.

```
aws ssm list-documents \  
  --filters Key=Owner,Values=Self Key=tag:DocUse,Values=Testing
```

Output:

```
{  
  "DocumentIdentifiers": [  
    {  
      "Name": "Example",  
      "Owner": "29884EXAMPLE",  
      "PlatformTypes": [  
        "Windows",  
        "Linux"  
      ],  
      "DocumentVersion": "1",  
      "DocumentType": "Automation",  
      "SchemaVersion": "0.3",  
      "DocumentFormat": "YAML",  
      "Tags": [  
        {  
          "Key": "DocUse",  
          "Value": "Testing"  
        }  
      ]  
    }  
  ]  
}
```

For more information, see [AWS Systems Manager Documents](#) in the *AWS Systems Manager User Guide*.

Example 2: To list shared documents

The following `list-documents` example lists shared documents, including private shared documents not owned by AWS.

```
aws ssm list-documents \  
  --filters Key=Name,Values=sharedDocNamePrefix Key=Owner,Values=Private
```

Output:

```
{
  "DocumentIdentifiers": [
    {
      "Name": "Example",
      "Owner": "12345EXAMPLE",
      "PlatformTypes": [
        "Windows",
        "Linux"
      ],
      "DocumentVersion": "1",
      "DocumentType": "Command",
      "SchemaVersion": "0.3",
      "DocumentFormat": "YAML",
      "Tags": []
    }
  ]
}
```

For more information, see [AWS Systems Manager Documents](#) in the *AWS Systems Manager User Guide*.

- For API details, see [ListDocuments](#) in *AWS CLI Command Reference*.

list-inventory-entries

The following code example shows how to use `list-inventory-entries`.

AWS CLI**Example 1: To view specific inventory type entries for an instance**

This following `list-inventory-entries` example lists the inventory entries for the `AWS:Application` inventory type on a specific instance.

```
aws ssm list-inventory-entries \
  --instance-id "i-1234567890abcdef0" \
  --type-name "AWS:Application"
```

Output:

```
{
  "TypeName": "AWS:Application",
  "InstanceId": "i-1234567890abcdef0",
  "SchemaVersion": "1.1",
  "CaptureTime": "2019-02-15T12:17:55Z",
  "Entries": [
    {
      "Architecture": "i386",
      "Name": "Amazon SSM Agent",
      "PackageId": "{88a60be2-89a1-4df8-812a-80863c2a2b68}",
      "Publisher": "Amazon Web Services",
      "Version": "2.3.274.0"
    },
    {
      "Architecture": "x86_64",
      "InstalledTime": "2018-05-03T13:42:34Z",
      "Name": "AmazonCloudWatchAgent",
      "Publisher": "",
      "Version": "1.200442.0"
    }
  ]
}
```

Example 2: To view custom inventory entries assigned to an instance

The following `list-inventory-entries` example lists a custom inventory entry assigned to an instance.

```
aws ssm list-inventory-entries \
  --instance-id "i-1234567890abcdef0" \
  --type-name "Custom:RackInfo"
```

Output:

```
{
  "TypeName": "Custom:RackInfo",
  "InstanceId": "i-1234567890abcdef0",
  "SchemaVersion": "1.0",
  "CaptureTime": "2021-05-22T10:01:01Z",
  "Entries": [
    {
      "RackLocation": "Bay B/Row C/Rack D/Shelf E"
    }
  ]
}
```

```

    }
  ]
}

```

- For API details, see [ListInventoryEntries](#) in *AWS CLI Command Reference*.

list-ops-item-related-items

The following code example shows how to use `list-ops-item-related-items`.

AWS CLI

To list the related-item resources of an OpsItem

The following `list-ops-item-related-items` example lists the related-item resources of an OpsItem.

```

aws ssm list-ops-item-related-items \
  --ops-item-id "oi-f99f2EXAMPLE"

```

Output:

```

{
  "Summaries": [
    {
      "OpsItemId": "oi-f99f2EXAMPLE",
      "AssociationId": "e2036148-cccb-490e-ac2a-390e5EXAMPLE",
      "ResourceType": "AWS::SSMIncidents::IncidentRecord",
      "AssociationType": "IsParentOf",
      "ResourceUri": "arn:aws:ssm-incidents::111122223333:incident-record/example-response/64bd9b45-1d0e-2622-840d-03a87a1451fa",
      "CreatedBy": {
        "Arn": "arn:aws:sts::111122223333:assumed-role/AWSServiceRoleForIncidentManager/IncidentResponse"
      },
      "CreatedTime": "2021-08-11T18:47:14.994000+00:00",
      "LastModifiedBy": {
        "Arn": "arn:aws:sts::111122223333:assumed-role/AWSServiceRoleForIncidentManager/IncidentResponse"
      },
      "LastModifiedTime": "2021-08-11T18:47:14.994000+00:00"
    }
  ]
}

```



```
]
}
```

For more information, see [Working with Incident Manager incidents in OpsCenter](#) in the *AWS Systems Manager User Guide*.

- For API details, see [ListOpsItemRelatedItems](#) in *AWS CLI Command Reference*.

list-resource-compliance-summaries

The following code example shows how to use `list-resource-compliance-summaries`.

AWS CLI

To list resource-level compliance summary counts

This example lists resource-level compliance summary counts.

Command:

```
aws ssm list-resource-compliance-summaries
```

Output:

```
{
  "ResourceComplianceSummaryItems": [
    {
      "ComplianceType": "Association",
      "ResourceType": "ManagedInstance",
      "ResourceId": "i-1234567890abcdef0",
      "Status": "COMPLIANT",
      "OverallSeverity": "UNSPECIFIED",
      "ExecutionSummary": {
        "ExecutionTime": 1550509273.0
      },
      "CompliantSummary": {
        "CompliantCount": 2,
        "SeveritySummary": {
          "CriticalCount": 0,
          "HighCount": 0,
          "MediumCount": 0,
          "LowCount": 0,
          "InformationalCount": 0,

```

```
        "UnspecifiedCount": 2
      }
    },
    "NonCompliantSummary": {
      "NonCompliantCount": 0,
      "SeveritySummary": {
        "CriticalCount": 0,
        "HighCount": 0,
        "MediumCount": 0,
        "LowCount": 0,
        "InformationalCount": 0,
        "UnspecifiedCount": 0
      }
    }
  },
  {
    "ComplianceType": "Patch",
    "ResourceType": "ManagedInstance",
    "ResourceId": "i-9876543210abcdef0",
    "Status": "COMPLIANT",
    "OverallSeverity": "UNSPECIFIED",
    "ExecutionSummary": {
      "ExecutionTime": 1550248550.0,
      "ExecutionId": "7abb6378-a4a5-4f10-8312-0123456789ab",
      "ExecutionType": "Command"
    },
    "CompliantSummary": {
      "CompliantCount": 397,
      "SeveritySummary": {
        "CriticalCount": 0,
        "HighCount": 0,
        "MediumCount": 0,
        "LowCount": 0,
        "InformationalCount": 0,
        "UnspecifiedCount": 397
      }
    },
    "NonCompliantSummary": {
      "NonCompliantCount": 0,
      "SeveritySummary": {
        "CriticalCount": 0,
        "HighCount": 0,
        "MediumCount": 0,
        "LowCount": 0,
```

```

        "InformationalCount": 0,
        "UnspecifiedCount": 0
      }
    }
  ],
  "NextToken": "--token string truncated--"
}

```

To list resource-level compliance summaries for a specific compliance type

This example lists resource-level compliance summaries for the Patch compliance type.

Command:

```
aws ssm list-resource-compliance-summaries --filters
"Key=ComplianceType,Values=Patch,Type=EQUAL"
```

- For API details, see [ListResourceComplianceSummaries](#) in *AWS CLI Command Reference*.

list-resource-data-sync

The following code example shows how to use `list-resource-data-sync`.

AWS CLI

To list your resource data sync configurations

This example retrieves information about your resource data sync configurations.

```
aws ssm list-resource-data-sync
```

Output:

```

{
  "ResourceDataSyncItems": [
    {
      "SyncName": "MyResourceDataSync",
      "S3Destination": {
        "BucketName": "ssm-resource-data-sync",
        "SyncFormat": "JsonSerDe",
        "Region": "us-east-1"
      }
    }
  ]
}

```

```
    },
    "LastSyncTime": 1550261472.003,
    "LastSuccessfulSyncTime": 1550261472.003,
    "LastStatus": "Successful",
    "SyncCreatedTime": 1543235736.72,
    "LastSyncStatusMessage": "The sync was successfully completed"
  }
]
}
```

- For API details, see [ListResourceDataSync](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags applied to a patch baseline

The following `list-tags-for-resource` example lists the tags for a patch baseline.

```
aws ssm list-tags-for-resource \
  --resource-type "PatchBaseline" \
  --resource-id "pb-0123456789abcdef0"
```

Output:

```
{
  "TagList": [
    {
      "Key": "Environment",
      "Value": "Production"
    },
    {
      "Key": "Region",
      "Value": "EMEA"
    }
  ]
}
```

For more information, see [Tagging AWS Resources](#) in the *AWS General Reference*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

modify-document-permission

The following code example shows how to use `modify-document-permission`.

AWS CLI

To modify document permissions

The following `modify-document-permission` example shares a Systems Manager document publicly.

```
aws ssm modify-document-permission \  
  --name "Example" \  
  --permission-type "Share" \  
  --account-ids-to-add "All"
```

This command produces no output.

For more information, see [Share a Systems Manager Document](#) in the *AWS Systems Manager User Guide*.

- For API details, see [ModifyDocumentPermission](#) in *AWS CLI Command Reference*.

put-compliance-items

The following code example shows how to use `put-compliance-items`.

AWS CLI

To register a compliance type and compliance details to a designated instance

This example registers the compliance type `Custom:AVCheck` to the specified managed instance. There is no output if the command succeeds.

Command:

```
aws ssm put-compliance-items --resource-id "i-1234567890abcdef0" --  
resource-type "ManagedInstance" --compliance-type "Custom:AVCheck"  
  --execution-summary "ExecutionTime=2019-02-18T16:00:00Z" --items  
  "Id=Version2.0,Title=ScanHost,Severity=CRITICAL,Status=COMPLIANT"
```

- For API details, see [PutComplianceItems](#) in *AWS CLI Command Reference*.

put-inventory

The following code example shows how to use `put-inventory`.

AWS CLI

To assign customer metadata to an instance

This example assigns rack location information to an instance. There is no output if the command succeeds.

Command (Linux):

```
aws ssm put-inventory --instance-id "i-016648b75dd622dab" --items
' [{"TypeName": "Custom:RackInfo", "SchemaVersion": "1.0", "CaptureTime":
"2019-01-22T10:01:01Z", "Content": [{"RackLocation": "Bay B/Row C/Rack D/Shelf
E"}]} ]'
```

Command (Windows):

```
aws ssm put-inventory --instance-id "i-016648b75dd622dab" --items
"TypeName=Custom:RackInfo,SchemaVersion=1.0,CaptureTime=2019-01-22T10:01:01Z,Content=[{Rack
B/Row C/Rack D/Shelf F'}]"
```

- For API details, see [PutInventory](#) in *AWS CLI Command Reference*.

put-parameter

The following code example shows how to use `put-parameter`.

AWS CLI

Example 1: To change a parameter value

The following `put-parameter` example changes the value of the specified parameter.

```
aws ssm put-parameter \
  --name "MyStringParameter" \
  --type "String" \
  --value "Vici" \
```

```
--overwrite
```

Output:

```
{
  "Version": 2,
  "Tier": "Standard"
}
```

For more information, see [Create a Systems Manager parameter \(AWS CLI\)](https://docs.aws.amazon.com/systems-manager/latest/userguide/parameter-store-advanced-parameters.html), 'Managing parameter tiers <<https://docs.aws.amazon.com/systems-manager/latest/userguide/parameter-store-advanced-parameters.html>>` __, and [Working with parameter policies](#) in the *AWS Systems Manager User Guide*.

Example 2: To create an advanced parameter

The following `put-parameter` example creates an advanced parameter.

```
aws ssm put-parameter \
  --name "MyAdvancedParameter" \
  --description "This is an advanced parameter" \
  --value "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat [truncated]" \
  --type "String" \
  --tier Advanced
```

Output:

```
{
  "Version": 1,
  "Tier": "Advanced"
}
```

For more information, see [Create a Systems Manager parameter \(AWS CLI\)](https://docs.aws.amazon.com/systems-manager/latest/userguide/parameter-store-advanced-parameters.html), 'Managing parameter tiers <<https://docs.aws.amazon.com/systems-manager/latest/userguide/parameter-store-advanced-parameters.html>>` __, and [Working with parameter policies](#) in the *AWS Systems Manager User Guide*.

Example 3: To convert a standard parameter to an advanced parameter

The following `put-parameter` example converts a existing standard parameter into an advanced parameter.

```
aws ssm put-parameter \  
  --name "MyConvertedParameter" \  
  --value "abc123" \  
  --type "String" \  
  --tier Advanced \  
  --overwrite
```

Output:

```
{  
  "Version": 2,  
  "Tier": "Advanced"  
}
```

For more information, see [Create a Systems Manager parameter \(AWS CLI\)](#), 'Managing parameter tiers <<https://docs.aws.amazon.com/systems-manager/latest/userguide/parameter-store-advanced-parameters.html>>` __, and [Working with parameter policies](#) in the *AWS Systems Manager User Guide*.

Example 4: To create a parameter with a policy attached

The following `put-parameter` example creates an advanced parameter with a parameter policy attached.

```
aws ssm put-parameter \  
  --name "/Finance/Payroll/q2accesskey" \  
  --value "P@sSwW)rd" \  
  --type "SecureString" \  
  --tier Advanced \  
  --policies "[{\"Type\":\"Expiration\",\"Version\":\"1.0\",\"Attributes\":{\"Timestamp\":\"2020-06-30T00:00:00.000Z\"}},{\"Type\":\"ExpirationNotification\",\"Version\":\"1.0\",\"Attributes\":{\"Before\":\"5\",\"Unit\":\"Days\"}},{\"Type\":\"NoChangeNotification\",\"Version\":\"1.0\",\"Attributes\":{\"After\":\"60\",\"Unit\":\"Days\"}}]"
```

Output:

```
{
```



```

    "Version": 1,
    "Tier": "Advanced"
  }

```

For more information, see [Create a Systems Manager parameter \(AWS CLI\)](https://docs.aws.amazon.com/systems-manager/latest/userguide/parameter-store-advanced-parameters.html), 'Managing parameter tiers <<https://docs.aws.amazon.com/systems-manager/latest/userguide/parameter-store-advanced-parameters.html>> `___, and [Working with parameter policies](#) in the *AWS Systems Manager User Guide*.

Example 5: To add a policy to an existing parameter

The following `put-parameter` example attaches a policy to an existing advanced parameter.

```

aws ssm put-parameter \
  --name "/Finance/Payroll/q2accesskey" \
  --value "N3wP@sSw)rd" \
  --type "SecureString" \
  --tier Advanced \
  --policies "[{\"Type\":\"Expiration\",\"Version\":\"1.0\",\"Attributes\":{\"Timestamp\":\"2020-06-30T00:00:00.000Z\"}},{\"Type\":\"ExpirationNotification\",\"Version\":\"1.0\",\"Attributes\":{\"Before\":\"5\",\"Unit\":\"Days\"}},{\"Type\":\"NoChangeNotification\",\"Version\":\"1.0\",\"Attributes\":{\"After\":\"60\",\"Unit\":\"Days\"}}]"
  --overwrite

```

Output:

```

{
  "Version": 2,
  "Tier": "Advanced"
}

```

For more information, see [Create a Systems Manager parameter \(AWS CLI\)](https://docs.aws.amazon.com/systems-manager/latest/userguide/parameter-store-advanced-parameters.html), 'Managing parameter tiers <<https://docs.aws.amazon.com/systems-manager/latest/userguide/parameter-store-advanced-parameters.html>> `___, and [Working with parameter policies](#) in the *AWS Systems Manager User Guide*.

- For API details, see [PutParameter](#) in *AWS CLI Command Reference*.

register-default-patch-baseline

The following code example shows how to use `register-default-patch-baseline`.

AWS CLI

To set the default patch baseline

The following `register-default-patch-baseline` example registers the specified custom patch baseline as the default patch baseline for the operating system type that it supports.

```
aws ssm register-default-patch-baseline \  
  --baseline-id "pb-abc123cf9bEXAMPLE"
```

Output:

```
{  
  "BaselineId": "pb-abc123cf9bEXAMPLE"  
}
```

The following `register-default-patch-baseline` example registers the default patch baseline provided by AWS for CentOS as the default patch baseline.

```
aws ssm register-default-patch-baseline \  
  --baseline-id "arn:aws:ssm:us-east-2:733109147000:patchbaseline/  
pb-0574b43a65ea646ed"
```

Output:

```
{  
  "BaselineId": "pb-abc123cf9bEXAMPLE"  
}
```

For more information, see [About Predefined and Custom Patch Baselines](#) in the *AWS Systems Manager User Guide*.

- For API details, see [RegisterDefaultPatchBaseline](#) in *AWS CLI Command Reference*.

register-patch-baseline-for-patch-group

The following code example shows how to use `register-patch-baseline-for-patch-group`.

AWS CLI

To register a patch baseline for a patch group

The following `register-patch-baseline-for-patch-group` example registers a patch baseline for a patch group.

```
aws ssm register-patch-baseline-for-patch-group \  
  --baseline-id "pb-045f10b4f382baeda" \  
  --patch-group "Production"
```

Output:

```
{  
  "BaselineId": "pb-045f10b4f382baeda",  
  "PatchGroup": "Production"  
}
```

For more information, see [Create a Patch Group <https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-group-tagging.html>](https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-group-tagging.html) and [Add a Patch Group to a Patch Baseline](#) in the *AWS Systems Manager User Guide*.

- For API details, see [RegisterPatchBaselineForPatchGroup](#) in *AWS CLI Command Reference*.

register-target-with-maintenance-window

The following code example shows how to use `register-target-with-maintenance-window`.

AWS CLI

Example 1: To register a single target with a maintenance window

The following `register-target-with-maintenance-window` example registers an instance with a maintenance window.

```
aws ssm register-target-with-maintenance-window \  
  --window-id "mw-ab12cd34ef56gh78" \  
  --target "Key=InstanceIds,Values=i-0000293ffd8c57862" \  
  --owner-information "Single instance" \  
  --resource-type "INSTANCE"
```

Output:

```
{  
  "WindowTargetId": "1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
```

```
}
```

Example 2: To register multiple targets with a maintenance window using instance IDs

The following `register-target-with-maintenance-window` example registers two instances with a maintenance window by specifying their instance IDs.

```
aws ssm register-target-with-maintenance-window \  
  --window-id "mw-ab12cd34ef56gh78" \  
  --target "Key=InstanceIds,Values=i-0000293ffd8c57862,i-0cb2b964d3e14fd9f" \  
  --owner-information "Two instances in a list" \  
  --resource-type "INSTANCE"
```

Output:

```
{  
  "WindowTargetId": "1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"  
}
```

Example 3: To register targets with a maintenance window using resource tags

The following `register-target-with-maintenance-window` example registers instances with a maintenance window by specifying resource tags that have been applied to the instances.

```
aws ssm register-target-with-maintenance-window \  
  --window-id "mw-06cf17cbefcb4bf4f" \  
  --targets "Key=tag:Environment,Values=Prod" "Key=Role,Values=Web" \  
  --owner-information "Production Web Servers" \  
  --resource-type "INSTANCE"
```

Output:

```
{  
  "WindowTargetId": "1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"  
}
```

Example 4: To register targets using a group of tag keys

The following `register-target-with-maintenance-window` example registers instances that all have one or more tag keys assigned to them, regardless of their key values.

```
aws ssm register-target-with-maintenance-window \  
  --window-id "mw-0c50858d01EXAMPLE" \  
  --resource-type "INSTANCE" \  
  --target "Key=tag-key,Values=Name,Instance-Type,CostCenter"
```

Output:

```
{  
  "WindowTargetId": "1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"  
}
```

Example 5: To register targets using a resource group name

The following `register-target-with-maintenance-window` example registers a specified resource group, regardless of the type of resources it contains.

```
aws ssm register-target-with-maintenance-window \  
  --window-id "mw-0c50858d01EXAMPLE" \  
  --resource-type "RESOURCE_GROUP" \  
  --target "Key=resource-groups:Name,Values=MyResourceGroup"
```

Output:

```
{  
  "WindowTargetId": "1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"  
}
```

For more information, see [Register a Target Instance with the Maintenance Window \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [RegisterTargetWithMaintenanceWindow](#) in *AWS CLI Command Reference*.

register-task-with-maintenance-window

The following code example shows how to use `register-task-with-maintenance-window`.

AWS CLI

Example 1: To register an Automation task with a maintenance window

The following `register-task-with-maintenance-window` example registers an Automation task with a maintenance window that is targeted at an instance.

```
aws ssm register-task-with-maintenance-window \
  --window-id "mw-082dcd7649EXAMPLE" \
  --targets Key=InstanceIds,Values=i-1234520122EXAMPLE \
  --task-arn AWS-RestartEC2Instance \
  --service-role-arn arn:aws:iam::111222333444:role/SSM --task-type AUTOMATION \
  --task-invocation-parameters "{\"Automation\":{\"DocumentVersion\":{\"\"$LATEST\"},\
  \"Parameters\":{\"\"InstanceId\":[\"{{RESOURCE_ID}}\"]}}}" \
  --priority 0 \
  --max-concurrency 1 \
  --max-errors 1 \
  --name "AutomationExample" \
  --description "Restarting EC2 Instance for maintenance"
```

Output:

```
{
  "WindowTaskId": "11144444-5555-6666-7777-88888888"
}
```

For more information, see [Register a Task with the Maintenance Window \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

Example 2: To register a Lambda task with a Maintenance Window

The following `register-task-with-maintenance-window` example registers a Lambda task with a Maintenance Window that is targeted at an instance.

```
aws ssm register-task-with-maintenance-window \
  --window-id "mw-082dcd7649dee04e4" \
  --targets Key=InstanceIds,Values=i-12344d305eEXAMPLE \
  --task-arn arn:aws:lambda:us-east-1:111222333444:function:SSMTestLAMBDA \
  --service-role-arn arn:aws:iam::111222333444:role/SSM \
  --task-type LAMBDA \
  --task-invocation-parameters '{"Lambda":{"Payload":{"\"InstanceId\":"\
  \"{{RESOURCE_ID}}\"\",\"targetType\":{\"\"{{TARGET_TYPE}}\""},\"Qualifier\":\"$LATEST\"}}' \
  --priority 0 \
  --max-concurrency 10 \
  --max-errors 5 \
  --name "Lambda_Example" \
```

```
--description "My Lambda Example"
```

Output:

```
{
  "WindowTaskId": "22244444-5555-6666-7777-88888888"
}
```

For more information, see [Register a Task with the Maintenance Window \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

Example 3: To register a Run Command task with a maintenance window

The following `register-task-with-maintenance-window` example registers a Run Command task with a maintenance window that is targeted at an instance.

```
aws ssm register-task-with-maintenance-window \
  --window-id "mw-082dcd7649dee04e4" \
  --targets "Key=InstanceIds,Values=i-12344d305eEXAMPLE" \
  --service-role-arn "arn:aws:iam::111222333444:role/SSM" \
  --task-type "RUN_COMMAND" \
  --name "SSMInstallPowerShellModule" \
  --task-arn "AWS-InstallPowerShellModule" \
  --task-invocation-parameters "{\"RunCommand\":{\"Comment\":"\"\",
  \"OutputS3BucketName\":"\"runcommandlogs\"\", \"Parameters\":"{\"commands\":[\"Get-
  Module -ListAvailable\"], \"executionTimeout\":[\"3600\"], \"source\":[\"https://
  gallery.technet.microsoft.com/EZOut-33ae0fb7/file/110351/1/EZOut.zip\"],
  \"workingDirectory\":[\"\\\\\\\\\\\\\\\\\"]}, \"TimeoutSeconds\":600}}" \
  --max-concurrency 1 \
  --max-errors 1 \
  --priority 10
```

Output:

```
{
  "WindowTaskId": "33344444-5555-6666-7777-88888888"
}
```

For more information, see [Register a Task with the Maintenance Window \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

Example 4: To register a Step Functions task with a maintenance window

The following `register-task-with-maintenance-window` example registers a Step Functions task with a maintenance window that is targeted at an instance.

```
aws ssm register-task-with-maintenance-window \
  --window-id "mw-1234d787d6EXAMPLE" \
  --targets Key=WindowTargetIds,Values=12347414-69c3-49f8-95b8-ed2dcEXAMPLE \
  --task-arn arn:aws:states:us-
east-1:111222333444:stateMachine:SSMTestStateMachine \
  --service-role-arn arn:aws:iam::111222333444:role/MaintenanceWindows \
  --task-type STEP_FUNCTIONS \
  --task-invocation-parameters '{"StepFunctions":{"Input":{"\"InstanceId\":
\"{{RESOURCE_ID}}\"}}}' \
  --priority 0 \
  --max-concurrency 10 \
  --max-errors 5 \
  --name "Step_Functions_Example" \
  --description "My Step Functions Example"
```

Output:

```
{
  "WindowTaskId": "444444444-5555-6666-7777-88888888"
}
```

For more information, see [Register a Task with the Maintenance Window \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

Example 5: To register a task using a maintenance windows target ID

The following `register-task-with-maintenance-window` example registers a task using a maintenance window target ID. The maintenance window target ID was in the output of the `aws ssm register-target-with-maintenance-window` command. You can also retrieve it from the output of the `aws ssm describe-maintenance-window-targets` command.

```
aws ssm register-task-with-maintenance-window \
  --targets "Key=WindowTargetIds,Values=350d44e6-28cc-44e2-951f-4b2c9EXAMPLE" \
  --task-arn "AWS-RunShellScript" \
  --service-role-arn "arn:aws:iam::111222333444:role/MaintenanceWindowsRole" \
  --window-id "mw-ab12cd34eEXAMPLE" \
  --task-type "RUN_COMMAND" \
  --task-parameters "{\"commands\":{\"Values\":[\"df\"]}}" \
  --max-concurrency 1 \
```



```
--max-errors 1 \  
--priority 10
```

Output:

```
{  
  "WindowTaskId": "33344444-5555-6666-7777-88888888"  
}
```

For more information, see [Register a Task with the Maintenance Window \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [RegisterTaskWithMaintenanceWindow](#) in *AWS CLI Command Reference*.

remove-tags-from-resource

The following code example shows how to use `remove-tags-from-resource`.

AWS CLI**To remove a tag from a patch baseline**

The following `remove-tags-from-resource` example removes tags from a patch baseline.

```
aws ssm remove-tags-from-resource \  
  --resource-type "PatchBaseline" \  
  --resource-id "pb-0123456789abcdef0" \  
  --tag-keys "Region"
```

This command produces no output.

For more information, see [Tagging AWS Resources](#) in the *AWS General Reference*.

- For API details, see [RemoveTagsFromResource](#) in *AWS CLI Command Reference*.

reset-service-setting

The following code example shows how to use `reset-service-setting`.

AWS CLI**To reset the service setting for Parameter Store throughput**

The following `reset-service-setting` example resets the service setting for Parameter Store throughput in the specified region to no longer use increased throughput.

```
aws ssm reset-service-setting \  
  --setting-id arn:aws:ssm:us-east-1:123456789012:servicesetting/ssm/parameter-  
store/high-throughput-enabled
```

Output:

```
{  
  "ServiceSetting": {  
    "SettingId": "/ssm/parameter-store/high-throughput-enabled",  
    "SettingValue": "false",  
    "LastModifiedDate": 1555532818.578,  
    "LastModifiedUser": "System",  
    "ARN": "arn:aws:ssm:us-east-1:123456789012:servicesetting/ssm/parameter-  
store/high-throughput-enabled",  
    "Status": "Default"  
  }  
}
```

For more information, see [Increasing Parameter Store Throughput](#) in the *AWS Systems Manager User Guide*.

- For API details, see [ResetServiceSetting](#) in *AWS CLI Command Reference*.

resume-session

The following code example shows how to use `resume-session`.

AWS CLI

To resume a Session Manager session

This `resume-session` example resumes a Session Manager session with an instance after it has been disconnected. Note that this interactive command requires the Session Manager plugin to be installed on the client machine making the call.

```
aws ssm resume-session \  
  --session-id Mary-Major-07a16060613c408b5
```

Output:

```
{
  "SessionId": "Mary-Major-07a16060613c408b5",
  "TokenValue":
    "AAEAAVbTGsa0nyvcUoNGqifbv5r/8l9xuQljCuY8qVcv0noBAAAAAFxtd3jIXAFUUXGTJ7zF/
    AWJpWdvi0lF5p3dlAgrqVIV06IEXhkHLz0/1gXKRKEME71E6TL0p1LDJAMZ
    +kREejkZu4c5AxMkrQjMF+gtHP1bYJKTwtHQd1wjulPLex08SH17g5R/
    wekrj6WsDUpnEegFBfGftpAIz2GXQVfTJXKfkc5qepQ11C11D0IT2doz0qXgHwfQHfAKLErM5dWDZqKwyT1Z3iw7unQd
    +ihfGa6MEJJ97Jmat/a2TspEn0jNn9Mvu5iwXIW2yCvWZrGUj+/
    QI5Xr7s1XJBEskR54o4fN0GV9RWl0RZsZm1m1ki0JJtiwwgZ",
  "StreamUrl": "wss://ssmmessages.us-east-2.amazonaws.com/v1/data-channel/Mary-
    Major-07a16060613c408b5?role=publish_subscribe"
}
```

For more information, see [Install the Session Manager Plugin for the AWS CLI](#) in the *AWS Systems Manager User Guide*.

- For API details, see [ResumeSession](#) in *AWS CLI Command Reference*.

send-automation-signal

The following code example shows how to use `send-automation-signal`.

AWS CLI

To send a signal to an automation execution

The following `send-automation-signal` example sends an Approve signal to an Automation execution.

```
aws ssm send-automation-signal \
  --automation-execution-id 73c8eef8-f4ee-4a05-820c-e354fEXAMPLE \
  --signal-type "Approve"
```

This command produces no output.

For more information, see [Running an Automation Workflow with Approvers](#) in the *AWS Systems Manager User Guide*.

- For API details, see [SendAutomationSignal](#) in *AWS CLI Command Reference*.

send-command

The following code example shows how to use send-command.

AWS CLI

Example 1: To run a command on one or more remote instances

The following send-command example runs an echo command on a target instance.

```
aws ssm send-command \  
  --document-name "AWS-RunShellScript" \  
  --parameters 'commands=["echo HelloWorld"]' \  
  --targets "Key=instanceids,Values=i-1234567890abcdef0" \  
  --comment "echo HelloWorld"
```

Output:

```
{  
  "Command": {  
    "CommandId": "92853adf-ba41-4cd6-9a88-142d1EXAMPLE",  
    "DocumentName": "AWS-RunShellScript",  
    "DocumentVersion": "",  
    "Comment": "echo HelloWorld",  
    "ExpiresAfter": 1550181014.717,  
    "Parameters": {  
      "commands": [  
        "echo HelloWorld"  
      ]  
    },  
    "InstanceIds": [  
      "i-0f00f008a2dcbefe2"  
    ],  
    "Targets": [],  
    "RequestedDateTime": 1550173814.717,  
    "Status": "Pending",  
    "StatusDetails": "Pending",  
    "OutputS3BucketName": "",  
    "OutputS3KeyPrefix": "",  
    "MaxConcurrency": "50",  
    "MaxErrors": "0",  
    "TargetCount": 1,  
    "CompletedCount": 0,  
    "ErrorCount": 0,  
  },  
}
```

```
"DeliveryTimedOutCount": 0,
"ServiceRole": "",
"NotificationConfig": {
  "NotificationArn": "",
  "NotificationEvents": [],
  "NotificationType": ""
},
"CloudWatchOutputConfig": {
  "CloudWatchLogGroupName": "",
  "CloudWatchOutputEnabled": false
}
}
```

For more information, see [Running Commands Using Systems Manager Run Command](#) in the *AWS Systems Manager User Guide*.

Example 2: To get IP information about an instance

The following send-command example retrieves the IP information about an instance.

```
aws ssm send-command \
  --instance-ids "i-1234567890abcdef0" \
  --document-name "AWS-RunShellScript" \
  --comment "IP config" \
  --parameters "commands=ifconfig"
```

See example 1 for sample output.

For more information, see [Running Commands Using Systems Manager Run Command](#) in the *AWS Systems Manager User Guide*.

Example 3: To run a command on instances with specific tags

The following send-command example runs a command on instances that have the tag key "ENV" and the value "Dev".

```
aws ssm send-command \
  --targets "Key=tag:ENV,Values=Dev" \
  --document-name "AWS-RunShellScript" \
  --parameters "commands=ifconfig"
```

See example 1 for sample output.

For more information, see [Running Commands Using Systems Manager Run Command](#) in the *AWS Systems Manager User Guide*.

Example 4: To run a command that sends SNS notifications

The following `send-command` example runs a command that sends SNS notifications for all notification events and the Command notification type.

```
aws ssm send-command \  
  --instance-ids "i-1234567890abcdef0" \  
  --document-name "AWS-RunShellScript" \  
  --comment "IP config" \  
  --parameters "commands=ifconfig" \  
  --service-role-arn "arn:aws:iam::123456789012:role/SNS_Role" \  
  --notification-config "NotificationArn=arn:aws:sns:us-  
east-1:123456789012:SNSTopicName,NotificationEvents=All,NotificationType=Command"
```

See example 1 for sample output.

For more information, see [Running Commands Using Systems Manager Run Command](#) in the *AWS Systems Manager User Guide*.

Example 5: To run a command that outputs to S3 and CloudWatch

The following `send-command` example runs a command that outputs command details to an S3 bucket and to a CloudWatch Logs log group.

```
aws ssm send-command \  
  --instance-ids "i-1234567890abcdef0" \  
  --document-name "AWS-RunShellScript" \  
  --comment "IP config" \  
  --parameters "commands=ifconfig" \  
  --output-s3-bucket-name "s3-bucket-name" \  
  --output-s3-key-prefix "runcommand" \  
  --cloud-watch-output-config  
  "CloudWatchOutputEnabled=true,CloudWatchLogGroupName=CWLGroupName"
```

See example 1 for sample output.

For more information, see [Running Commands Using Systems Manager Run Command](#) in the *AWS Systems Manager User Guide*.

Example 6: To run commands on multiple instances with different tags

The following `send-command` example runs a command on instances with two different tag keys and values.

```
aws ssm send-command \  
  --document-name "AWS-RunPowerShellScript" \  
  --parameters commands=["echo helloWorld"] \  
  --targets Key=tag:Env,Values=Dev Key=tag:Role,Values=WebServers
```

See example 1 for sample output.

For more information, see [Running Commands Using Systems Manager Run Command](#) in the *AWS Systems Manager User Guide*.

Example 7: To target multiple instances with the same tag key

The following `send-command` example runs a command on instances that have the same tag key but with different values.

```
aws ssm send-command \  
  --document-name "AWS-RunPowerShellScript" \  
  --parameters commands=["echo helloWorld"] \  
  --targets Key=tag:Env,Values=Dev,Test
```

See example 1 for sample output.

For more information, see [Running Commands Using Systems Manager Run Command](#) in the *AWS Systems Manager User Guide*.

Example 8: To run a command that uses a shared document

The following `send-command` example runs a shared document on a target instance.

```
aws ssm send-command \  
  --document-name "arn:aws:ssm:us-east-1:123456789012:document/ExampleDocument" \  
  --targets "Key=instanceids,Values=i-1234567890abcdef0"
```

See example 1 for sample output.

For more information, see [Using shared SSM documents](#) in the *AWS Systems Manager User Guide*.

- For API details, see [SendCommand](#) in *AWS CLI Command Reference*.

start-associations-once

The following code example shows how to use `start-associations-once`.

AWS CLI

To run an association immediately and only one time

The following `start-associations-once` example run the specified association immediately and only once. There is no output if the command succeeds.

```
aws ssm start-associations-once \  
  --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

This command produces no output.

For more information, see [Viewing association histories](#) in the *AWS Systems Manager User Guide*.

- For API details, see [StartAssociationsOnce](#) in *AWS CLI Command Reference*.

start-automation-execution

The following code example shows how to use `start-automation-execution`.

AWS CLI

Example 1: To execute an automation document

The following `start-automation-execution` example runs an Automation document.

```
aws ssm start-automation-execution \  
  --document-name "AWS-UpdateLinuxAmi" \  
  --parameters "AutomationAssumeRole=arn:aws:iam::123456789012:role/  
SSMAutomationRole,SourceAmiId=ami-EXAMPLE,IamInstanceProfileName=EC2InstanceRole"
```

Output:

```
{  
  "AutomationExecutionId": "4105a4fc-f944-11e6-9d32-0a1b2EXAMPLE"  
}
```

For more information, see [Running an Automation Workflow Manually](#) in the *AWS Systems Manager User Guide*.

Example 2: To run a shared automation document

The following `start-automation-execution` example runs a shared Automation document.

```
aws ssm start-automation-execution \  
  --document-name "arn:aws:ssm:us-east-1:123456789012:document/ExampleDocument"
```

Output:

```
{  
  "AutomationExecutionId": "4105a4fc-f944-11e6-9d32-0a1b2EXAMPLE"  
}
```

For more information, see [Using shared SSM documents](#) in the *AWS Systems Manager User Guide*.

- For API details, see [StartAutomationExecution](#) in *AWS CLI Command Reference*.

start-change-request-execution

The following code example shows how to use `start-change-request-execution`.

AWS CLI

Example 1: To start a change request

The following `start-change-request-execution` example starts a change request with minimal options specified.

```
aws ssm start-change-request-execution \  
  --change-request-name MyChangeRequest \  
  --document-name AWS-HelloWorldChangeTemplate \  
  --runbooks '[{"DocumentName": "AWS-HelloWorld", "Parameters":  
  {"AutomationAssumeRole": ["arn:aws:iam:us-east-2:1112223233444:role/  
MyChangeManagerAssumeRole"]}]}' \  
  --parameters  
  Approver="JohnDoe", ApproverType="IamUser", ApproverSnsTopicArn="arn:aws:sns:us-  
east-2:1112223233444:MyNotificationTopic"
```

Output:

```
{
```

```
"AutomationExecutionId": "9d32a4fc-f944-11e6-4105-0a1b2EXAMPLE"  
}
```

Example 2: To start a change request using an external JSON file

The following `start-automation-execution` example starts a change request with multiple options specified in a JSON file.

```
aws ssm start-change-request-execution \  
  --cli-input-json file://MyChangeRequest.json
```

Contents of `MyChangeRequest.json`:

```
{  
  "ChangeRequestName": "MyChangeRequest",  
  "DocumentName": "AWS-HelloWorldChangeTemplate",  
  "DocumentVersion": "$DEFAULT",  
  "ScheduledTime": "2021-12-30T03:00:00",  
  "ScheduledEndTime": "2021-12-30T03:05:00",  
  "Tags": [  
    {  
      "Key": "Purpose",  
      "Value": "Testing"  
    }  
  ],  
  "Parameters": {  
    "Approver": [  
      "JohnDoe"  
    ],  
    "ApproverType": [  
      "IamUser"  
    ],  
    "ApproverSnsTopicArn": [  
      "arn:aws:sns:us-east-2:111222333444:MyNotificationTopic"  
    ]  
  },  
  "Runbooks": [  
    {  
      "DocumentName": "AWS-HelloWorld",  
      "DocumentVersion": "1",  
      "MaxConcurrency": "1",  
      "MaxErrors": "1",  
      "Parameters": {
```

```

        "AutomationAssumeRole": [
            "arn:aws:iam::111222333444:role/MyChangeManagerAssumeRole"
        ]
    },
    ],
    "ChangeDetails": "### Document Name: HelloWorldChangeTemplate\n\n## What does
this document do?\nThis change template demonstrates the feature set available
for creating change templates for Change Manager. This template starts a Runbook
workflow for the Automation document called AWS-HelloWorld.\n\n## Input Parameters
\n* ApproverSnsTopicArn: (Required) Amazon Simple Notification Service ARN for
approvers.\n* Approver: (Required) The name of the approver to send this request
to.\n* ApproverType: (Required) The type of reviewer.\n * Allowed Values: IamUser,
IamGroup, IamRole, SSOGroup, SSOUser\n\n## Output Parameters\nThis document has no
outputs \n"
}

```

Output:

```

{
  "AutomationExecutionId": "9d32a4fc-f944-11e6-4105-0a1b2EXAMPLE"
}

```

For more information, see [Creating change requests](#) in the *AWS Systems Manager User Guide*.

- For API details, see [StartChangeRequestExecution](#) in *AWS CLI Command Reference*.

start-session

The following code example shows how to use `start-session`.

AWS CLI

Example 1: To start a Session Manager session

This `start-session` example establishes a connection with an instance for a Session Manager session. Note that this interactive command requires the Session Manager plugin to be installed on the client machine making the call.

```

aws ssm start-session \
  --target "i-1234567890abcdef0"

```

Output:

```
Starting session with SessionId: Jane-Roe-07a16060613c408b5
```

Example 2: To start a Session Manager session using SSH

This `start-session` example establishes a connection with an instance for a Session Manager session using SSH. Note that this interactive command requires the Session Manager plugin to be installed on the client machine making the call, and that the command uses the default user on the instance, such as `ec2-user` for EC2 instances for Linux.

```
ssh -i /path/my-key-pair.pem ec2-user@i-02573cafcfEXAMPLE
```

Output:

```
Starting session with SessionId: ec2-user-07a16060613c408b5
```

For more information, see [Start a Session](#) and [Install the Session Manager Plugin for the AWS CLI](#) in the *AWS Systems Manager User Guide*.

- For API details, see [StartSession](#) in *AWS CLI Command Reference*.

stop-automation-execution

The following code example shows how to use `stop-automation-execution`.

AWS CLI**To stop an automation execution**

The following `stop-automation-execution` example stops an Automation document.

```
aws ssm stop-automation-execution  
  --automation-execution-id "4105a4fc-f944-11e6-9d32-0a1b2EXAMPLE"
```

This command produces no output.

For more information, see [Running an Automation Workflow Manually](#) in the *AWS Systems Manager User Guide*.

- For API details, see [StopAutomationExecution](#) in *AWS CLI Command Reference*.

terminate-session

The following code example shows how to use `terminate-session`.

AWS CLI

To end a Session Manager session

This `terminate-session` example permanently ends a session that was created by the user "Shirley-Rodriguez" and closes the data connection between the Session Manager client and SSM Agent on the instance.

```
aws ssm terminate-session \  
  --session-id "Shirley-Rodriguez-07a16060613c408b5"
```

Output:

```
{  
  "SessionId": "Shirley-Rodriguez-07a16060613c408b5"  
}
```

For more information, see [Terminate a Session](#) in the *AWS Systems Manager User Guide*.

- For API details, see [TerminateSession](#) in *AWS CLI Command Reference*.

unlabel-parameter-version

The following code example shows how to use `unlabel-parameter-version`.

AWS CLI

To delete parameter labels

The following `unlabel-parameter-version` example deletes the specified labels from the given parameter version.

```
aws ssm unlabel-parameter-version \  
  --name "parameterName" \  
  --parameter-version "version" \  
  --labels "label_1" "label_2" "label_3"
```

Output:

```
{
  "RemovedLabels": [
    "label_1"
    "label_2"
    "label_3"
  ],
  "InvalidLabels": []
}
```

For more information, see [Delete parameter labels \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [UnlabelParameterVersion](#) in *AWS CLI Command Reference*.

update-association-status

The following code example shows how to use `update-association-status`.

AWS CLI**To update the association status**

The following `update-association-status` example updates the association status of the association between an instance and a document.

```
aws ssm update-association-status \
  --name "AWS-UpdateSSMAgent" \
  --instance-id "i-1234567890abcdef0" \
  --association-status
  "Date=1424421071.939,Name=Pending,Message=temp_status_change,AdditionalInfo=Additional-
  Config-Needed"
```

Output:

```
{
  "AssociationDescription": {
    "Name": "AWS-UpdateSSMAgent",
    "InstanceId": "i-1234567890abcdef0",
    "AssociationVersion": "1",
    "Date": 1550507529.604,
```

```

    "LastUpdateAssociationDate": 1550507806.974,
    "Status": {
      "Date": 1424421071.0,
      "Name": "Pending",
      "Message": "temp_status_change",
      "AdditionalInfo": "Additional-Config-Needed"
    },
    "Overview": {
      "Status": "Success",
      "AssociationStatusAggregatedCount": {
        "Success": 1
      }
    },
    "DocumentVersion": "$DEFAULT",
    "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
    "Targets": [
      {
        "Key": "InstanceIds",
        "Values": [
          "i-1234567890abcdef0"
        ]
      }
    ],
    "LastExecutionDate": 1550507808.0,
    "LastSuccessfulExecutionDate": 1550507808.0
  }
}

```

For more information, see [Working with associations in Systems Manager](#) in the *AWS Systems Manager User Guide*.

- For API details, see [UpdateAssociationStatus](#) in *AWS CLI Command Reference*.

update-association

The following code example shows how to use update-association.

AWS CLI

Example 1: To update a document association

The following update-association example updates an association with a new document version.

```
aws ssm update-association \  
  --association-id "8dfe3659-4309-493a-8755-0123456789ab" \  
  --document-version "$LATEST"
```

Output:

```
{  
  "AssociationDescription": {  
    "Name": "AWS-UpdateSSMAgent",  
    "AssociationVersion": "2",  
    "Date": 1550508093.293,  
    "LastUpdateAssociationDate": 1550508106.596,  
    "Overview": {  
      "Status": "Pending",  
      "DetailedStatus": "Creating"  
    },  
    "DocumentVersion": "$LATEST",  
    "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",  
    "Targets": [  
      {  
        "Key": "tag:Name",  
        "Values": [  
          "Linux"  
        ]  
      }  
    ],  
    "LastExecutionDate": 1550508094.879,  
    "LastSuccessfulExecutionDate": 1550508094.879  
  }  
}
```

For more information, see [Editing and creating a new version of an association](#) in the *AWS Systems Manager User Guide*.

Example 2: To update the schedule expression of an association

The following `update-association` example updates the schedule expression for the specified association.

```
aws ssm update-association \  
  --association-id "8dfe3659-4309-493a-8755-0123456789ab" \  
  --document-version "$LATEST"
```



```
--schedule-expression "cron(0 0 0/4 1/1 * ? *)"
```

Output:

```
{
  "AssociationDescription": {
    "Name": "AWS-HelloWorld",
    "AssociationVersion": "2",
    "Date": "2021-02-08T13:54:19.203000-08:00",
    "LastUpdateAssociationDate": "2021-06-29T11:51:07.933000-07:00",
    "Overview": {
      "Status": "Pending",
      "DetailedStatus": "Creating"
    },
    "DocumentVersion": "$DEFAULT",
    "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
    "Targets": [
      {
        "Key": "aws:NoOpAutomationTag",
        "Values": [
          "AWS-NoOpAutomationTarget-Value"
        ]
      }
    ],
    "ScheduleExpression": "cron(0 0 0/4 1/1 * ? *)",
    "LastExecutionDate": "2021-06-26T19:00:48.110000-07:00",
    "ApplyOnlyAtCronInterval": false
  }
}
```

For more information, see [Editing and creating a new version of an association](#) in the *AWS Systems Manager User Guide*.

- For API details, see [UpdateAssociation](#) in *AWS CLI Command Reference*.

update-document-default-version

The following code example shows how to use `update-document-default-version`.

AWS CLI**To update the default version of a document**

The following `update-document-default-version` example updates the default version of a Systems Manager document.

```
aws ssm update-document-default-version \  
  --name "Example" \  
  --document-version "2"
```

Output:

```
{  
  "Description": {  
    "Name": "Example",  
    "DefaultVersion": "2"  
  }  
}
```

For more information, see [Writing SSM Document Content](#) in the *AWS Systems Manager User Guide*.

- For API details, see [UpdateDocumentDefaultVersion](#) in *AWS CLI Command Reference*.

update-document-metadata

The following code example shows how to use `update-document-metadata`.

AWS CLI

Example: To approve the latest version of a change template

The following `update-document-metadata` provides an approval for the latest version of a change template that has been submitted for review.

```
aws ssm update-document-metadata \  
  --name MyChangeManagerTemplate \  
  --document-reviews 'Action=Approve,Comment=[{Type=Comment,Content=Approved!}]'
```

This command produces no output.

For more information, see [Reviewing and approving or rejecting change templates](#) in the *AWS Systems Manager User Guide*.

- For API details, see [UpdateDocumentMetadata](#) in *AWS CLI Command Reference*.

update-document

The following code example shows how to use update-document.

AWS CLI

To create a new version of a document

The following update-document example creates a new version of a document when run on a Windows computer. The document specified by --document must be in JSON format. Note that file:// must be referenced followed by the path of the content file. Because of the \$ at the beginning of the --document-version parameter, On Windows you must surround the value with double quotes. On Linux, MacOS, or at a PowerShell prompt, you must surround the value with single quotes.

Windows version:

```
aws ssm update-document \  
  --name "RunShellScript" \  
  --content "file://RunShellScript.json" \  
  --document-version "$LATEST"
```

Linux/Mac version:

```
aws ssm update-document \  
  --name "RunShellScript" \  
  --content "file://RunShellScript.json" \  
  --document-version '$LATEST'
```

Output:

```
{  
  "DocumentDescription": {  
    "Status": "Updating",  
    "Hash": "f775e5df4904c6fa46686c4722fae9de1950dace25cd9608ff8d622046b68d9b",  
    "Name": "RunShellScript",  
    "Parameters": [  
      {  
        "Type": "StringList",  
        "Name": "commands",  
        "Description": "(Required) Specify a shell script or a command to  
run."      }  
    ]  
  }  
}
```

```

    }
  ],
  "DocumentType": "Command",
  "PlatformTypes": [
    "Linux"
  ],
  "DocumentVersion": "2",
  "HashType": "Sha256",
  "CreateDate": 1487899655.152,
  "Owner": "809632081692",
  "SchemaVersion": "2.0",
  "DefaultVersion": "1",
  "LatestVersion": "2",
  "Description": "Run an updated script"
}
}

```

- For API details, see [UpdateDocument](#) in *AWS CLI Command Reference*.

update-maintenance-window-target

The following code example shows how to use `update-maintenance-window-target`.

AWS CLI

To update a maintenance window target

The following `update-maintenance-window-target` example updates only the name of a maintenance window target.

```

aws ssm update-maintenance-window-target \
  --window-id "mw-0c5ed765acEXAMPLE" \
  --window-target-id "57e8344e-fe64-4023-8191-6bf05EXAMPLE" \
  --name "NewName" \
  --no-replace

```

Output:

```

{
  "Description": "",
  "OwnerInformation": "",
  "WindowTargetId": "57e8344e-fe64-4023-8191-6bf05EXAMPLE",

```

```

    "WindowId": "mw-0c5ed765acEXAMPLE",
    "Targets": [
      {
        "Values": [
          "i-1234567890EXAMPLE"
        ],
        "Key": "InstanceIds"
      }
    ],
    "Name": "NewName"
  }

```

For more information, see [Update a Maintenance Window \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [UpdateMaintenanceWindowTarget](#) in *AWS CLI Command Reference*.

update-maintenance-window-task

The following code example shows how to use `update-maintenance-window-task`.

AWS CLI

To update a maintenance window task

The following `update-maintenance-window-task` example updates the service role for a maintenance window task.

```

aws ssm update-maintenance-window-task \
  --window-id "mw-0c5ed765acEXAMPLE" \
  --window-task-id "23d3809e-9fbe-4ddf-b41a-b49d7EXAMPLE" \
  --service-role-arn "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM"

```

Output:

```

{
  "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
  "MaxErrors": "1",
  "TaskArn": "AWS-UpdateEC2Config",
  "MaxConcurrency": "1",
  "WindowTaskId": "23d3809e-9fbe-4ddf-b41a-b49d7EXAMPLE",

```

```

    "TaskParameters": {},
    "Priority": 1,
    "TaskInvocationParameters": {
      "RunCommand": {
        "TimeoutSeconds": 600,
        "Parameters": {
          "allowDowngrade": [
            "false"
          ]
        }
      }
    },
    "WindowId": "mw-0c5ed765acEXAMPLE",
    "Description": "UpdateEC2Config",
    "Targets": [
      {
        "Values": [
          "57e8344e-fe64-4023-8191-6bf05EXAMPLE"
        ],
        "Key": "WindowTargetIds"
      }
    ],
    "Name": "UpdateEC2Config"
  }
}

```

For more information, see [Update a Maintenance Window \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [UpdateMaintenanceWindowTask](#) in *AWS CLI Command Reference*.

update-maintenance-window

The following code example shows how to use `update-maintenance-window`.

AWS CLI

Example 1: To update a maintenance window

The following `update-maintenance-window` example updates the name of a maintenance window.

```

aws ssm update-maintenance-window \
  --window-id "mw-1a2b3c4d5e6f7g8h9" \

```

```
--name "My-Renamed-MW"
```

Output:

```
{
  "Cutoff": 1,
  "Name": "My-Renamed-MW",
  "Schedule": "cron(0 16 ? * TUE *)",
  "Enabled": true,
  "AllowUnassociatedTargets": true,
  "WindowId": "mw-1a2b3c4d5e6f7g8h9",
  "Duration": 4
}
```

Example 2: To disable a maintenance window

The following `update-maintenance-window` example disables a maintenance window.

```
aws ssm update-maintenance-window \
  --window-id "mw-1a2b3c4d5e6f7g8h9" \
  --no-enabled
```

Example 3: To enable a maintenance window

The following `update-maintenance-window` example enables a maintenance window.

```
aws ssm update-maintenance-window \
  --window-id "mw-1a2b3c4d5e6f7g8h9" \
  --enabled
```

For more information, see [Update a Maintenance Window \(AWS CLI\)](#) in the *AWS Systems Manager User Guide*.

- For API details, see [UpdateMaintenanceWindow](#) in *AWS CLI Command Reference*.

update-managed-instance-role

The following code example shows how to use `update-managed-instance-role`.

AWS CLI

To update the IAM role of a managed instance

The following `update-managed-instance-role` example updates the IAM instance profile of a managed instance.

```
aws ssm update-managed-instance-role \  
  --instance-id "mi-08ab247cdfEXAMPLE" \  
  --iam-role "ExampleRole"
```

This command produces no output.

For more information, see [Step 4: Create an IAM Instance Profile for Systems Manager](#) in the *AWS Systems Manager User Guide*.

- For API details, see [UpdateManagedInstanceRole](#) in *AWS CLI Command Reference*.

update-ops-item

The following code example shows how to use `update-ops-item`.

AWS CLI

To update an OpsItem

The following `update-ops-item` example updates the description, priority, and category for an OpsItem. In addition, the command specifies an SNS topic where the notifications are sent when this OpsItem is edited or changed.

```
aws ssm update-ops-item \  
  --ops-item-id "oi-287b5EXAMPLE" \  
  --description "Primary OpsItem for failover event 2020-01-01-fh398yf" \  
  --priority 2 \  
  --category "Security" \  
  --notifications "Arn=arn:aws:sns:us-east-2:111222333444:my-us-east-2-topic"
```

Output:

```
This command produces no output.
```

For more information, see [Working with OpsItems](#) in the *AWS Systems Manager User Guide*.

- For API details, see [UpdateOpsItem](#) in *AWS CLI Command Reference*.

update-patch-baseline

The following code example shows how to use update-patch-baseline.

AWS CLI

Example 1: To update a patch baseline

The following update-patch-baseline example adds the specified two patches as rejected and one patch as approved to the specified patch baseline.

```
aws ssm update-patch-baseline \  
  --baseline-id "pb-0123456789abcdef0" \  
  --rejected-patches "KB2032276" "MS10-048" \  
  --approved-patches "KB2124261"
```

Output:

```
{  
  "BaselineId": "pb-0123456789abcdef0",  
  "Name": "WindowsPatching",  
  "OperatingSystem": "WINDOWS",  
  "GlobalFilters": {  
    "PatchFilters": []  
  },  
  "ApprovalRules": {  
    "PatchRules": [  
      {  
        "PatchFilterGroup": {  
          "PatchFilters": [  
            {  
              "Key": "PRODUCT",  
              "Values": [  
                "WindowsServer2016"  
              ]  
            }  
          ]  
        }  
      }  
    ],  
    "ComplianceLevel": "CRITICAL",  
    "ApproveAfterDays": 0,  
    "EnableNonSecurity": false  
  }  
}
```

```

    },
    "ApprovedPatches": [
      "KB2124261"
    ],
    "ApprovedPatchesComplianceLevel": "UNSPECIFIED",
    "ApprovedPatchesEnableNonSecurity": false,
    "RejectedPatches": [
      "KB2032276",
      "MS10-048"
    ],
    "RejectedPatchesAction": "ALLOW_AS_DEPENDENCY",
    "CreateDate": 1550244180.465,
    "ModifiedDate": 1550244180.465,
    "Description": "Patches for Windows Servers",
    "Sources": []
  }
}

```

Example 2: To rename a patch baseline

The following `update-patch-baseline` example renames the specified patch baseline.

```

aws ssm update-patch-baseline \
  --baseline-id "pb-0713accee01234567" \
  --name "Windows-Server-2012-R2-Important-and-Critical-Security-Updates"

```

For more information, see [Update or Delete a Patch Baseline](https://docs.aws.amazon.com/systems-manager/latest/userguide/patch-baseline-update-or-delete.html) <<https://docs.aws.amazon.com/systems-manager/latest/userguide/patch-baseline-update-or-delete.html>>` __ in the *AWS Systems Manager User Guide*.

- For API details, see [UpdatePatchBaseline](#) in *AWS CLI Command Reference*.

update-resource-data-sync

The following code example shows how to use `update-resource-data-sync`.

AWS CLI

To update a resource data sync

The following `update-resource-data-sync` example updates a `SyncFromSource` resource data sync.

```
aws ssm update-resource-data-sync \  
  --sync-name exampleSync \  
  --sync-type SyncFromSource \  
  --sync-source '{"SourceType":"SingleAccountMultiRegions", "SourceRegions":["us-  
east-1", "us-west-2"]}'
```

This command produces no output.

For more information, see [Setting Up Systems Manager Explorer to Display Data from Multiple Accounts and Regions](#) in the *AWS Systems Manager User Guide*.

- For API details, see [UpdateResourceDataSync](#) in *AWS CLI Command Reference*.

update-service-setting

The following code example shows how to use `update-service-setting`.

AWS CLI

To update the service setting for Parameter Store throughput

The following `update-service-setting` example updates the current service setting for Parameter Store throughput in the specified region to use increased throughput.

```
aws ssm update-service-setting \  
  --setting-id arn:aws:ssm:us-east-1:123456789012:servicesetting/ssm/parameter-  
store/high-throughput-enabled \  
  --setting-value true
```

This command produces no output.

For more information, see [Increasing Parameter Store Throughput](#) in the *AWS Systems Manager User Guide*.

- For API details, see [UpdateServiceSetting](#) in *AWS CLI Command Reference*.

Amazon Textract examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon Textract.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

analyze-document

The following code example shows how to use `analyze-document`.

AWS CLI

To analyze text in a document

The following `analyze-document` example shows how to analyze text in a document.

Linux/macOS:

```
aws textract analyze-document \
  --document '{"S3Object":{"Bucket":"bucket","Name":"document"}}' \
  --feature-types ['TABLES','FORMS']
```

Windows:

```
aws textract analyze-document \
  --document "{\"S3Object\":{\"Bucket\":\"bucket\",\"Name\":\"document\"}}\" \
  --feature-types [\"TABLES\",\"FORMS\"] \
  --region region-name
```

Output:

```
{
  "Blocks": [
```

```
{
  "Geometry": {
    "BoundingBox": {
      "Width": 1.0,
      "Top": 0.0,
      "Left": 0.0,
      "Height": 1.0
    },
    "Polygon": [
      {
        "Y": 0.0,
        "X": 0.0
      },
      {
        "Y": 0.0,
        "X": 1.0
      },
      {
        "Y": 1.0,
        "X": 1.0
      },
      {
        "Y": 1.0,
        "X": 0.0
      }
    ]
  },
  "Relationships": [
    {
      "Type": "CHILD",
      "Ids": [
        "87586964-d50d-43e2-ace5-8a890657b9a0",
        "a1e72126-21d9-44f4-a8d6-5c385f9002ba",
        "e889d012-8a6b-4d2e-b7cd-7a8b327d876a"
      ]
    }
  ],
  "BlockType": "PAGE",
  "Id": "c2227f12-b25d-4e1f-baea-1ee180d926b2"
},
"DocumentMetadata": {
  "Pages": 1
}
```

```
}
```

For more information, see Analyzing Document Text with Amazon Textract in the *Amazon Textract Developers Guide*

- For API details, see [AnalyzeDocument](#) in *AWS CLI Command Reference*.

detect-document-text

The following code example shows how to use detect-document-text.

AWS CLI

To detect text in a document

The following detect-document-text The following example shows how to detect text in a document.

Linux/macOS:

```
aws textract detect-document-text \  
  --document '{"S3Object":{"Bucket":"bucket","Name":"document"}}'
```

Windows:

```
aws textract detect-document-text \  
  --document "{\"S3Object\":{\"Bucket\":\"bucket\",\"Name\":\"document\"}}\" \  
  --region region-name
```

Output:

```
{  
  "Blocks": [  
    {  
      "Geometry": {  
        "BoundingBox": {  
          "Width": 1.0,  
          "Top": 0.0,  
          "Left": 0.0,  
          "Height": 1.0  
        },  
        "Polygon": [  

```

```

        {
            "Y": 0.0,
            "X": 0.0
        },
        {
            "Y": 0.0,
            "X": 1.0
        },
        {
            "Y": 1.0,
            "X": 1.0
        },
        {
            "Y": 1.0,
            "X": 0.0
        }
    ]
},
"Relationships": [
    {
        "Type": "CHILD",
        "Ids": [
            "896a9f10-9e70-4412-81ce-49ead73ed881",
            "0da18623-dc4c-463d-a3d1-9ac050e9e720",
            "167338d7-d38c-4760-91f1-79a8ec457bb2"
        ]
    }
],
"BlockType": "PAGE",
"Id": "21f0535e-60d5-4bc7-adf2-c05dd851fa25"
},
{
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "62490c26-37ea-49fa-8034-7a9ff9369c9c",
                "1e4f3f21-05bd-4da9-ba10-15d01e66604c"
            ]
        }
    ],
    "Confidence": 89.11581420898438,
    "Geometry": {
        "BoundingBox": {

```

```
        "Width": 0.33642634749412537,
        "Top": 0.17169663310050964,
        "Left": 0.13885067403316498,
        "Height": 0.49159330129623413
    },
    "Polygon": [
        {
            "Y": 0.17169663310050964,
            "X": 0.13885067403316498
        },
        {
            "Y": 0.17169663310050964,
            "X": 0.47527703642845154
        },
        {
            "Y": 0.6632899641990662,
            "X": 0.47527703642845154
        },
        {
            "Y": 0.6632899641990662,
            "X": 0.13885067403316498
        }
    ]
},
"Text": "He llo,",
"BlockType": "LINE",
"Id": "896a9f10-9e70-4412-81ce-49ead73ed881"
},
{
    "Relationships": [
        {
            "Type": "CHILD",
            "Ids": [
                "19b28058-9516-4352-b929-64d7cef29daf"
            ]
        }
    ]
},
"Confidence": 85.5694351196289,
"Geometry": {
    "BoundingBox": {
        "Width": 0.33182239532470703,
        "Top": 0.23131252825260162,
        "Left": 0.5091826915740967,
        "Height": 0.3766750991344452
    }
}
```



```
    },
    "Polygon": [
      {
        "Y": 0.23131252825260162,
        "X": 0.5091826915740967
      },
      {
        "Y": 0.23131252825260162,
        "X": 0.8410050868988037
      },
      {
        "Y": 0.607987642288208,
        "X": 0.8410050868988037
      },
      {
        "Y": 0.607987642288208,
        "X": 0.5091826915740967
      }
    ]
  },
  "Text": "worlc",
  "BlockType": "LINE",
  "Id": "0da18623-dc4c-463d-a3d1-9ac050e9e720"
}
],
"DocumentMetadata": {
  "Pages": 1
}
}
```

For more information, see Detecting Document Text with Amazon Textract in the *Amazon Textract Developers Guide*

- For API details, see [DetectDocumentText](#) in *AWS CLI Command Reference*.

get-document-analysis

The following code example shows how to use get-document-analysis.

AWS CLI

To get the results of asynchronous text analysis of a multi-page document

The following `get-document-analysis` example shows how to get the results of asynchronous text analysis of a multi-page document.

```
aws textract get-document-analysis \
  --job-id df7cf32ebbd2a5de113535fcf4d921926a701b09b4e7d089f3aebadb41e0712b \
  --max-results 1000
```

Output:

```
{
  "Blocks": [
    {
      "Geometry": {
        "BoundingBox": {
          "Width": 1.0,
          "Top": 0.0,
          "Left": 0.0,
          "Height": 1.0
        },
        "Polygon": [
          {
            "Y": 0.0,
            "X": 0.0
          },
          {
            "Y": 0.0,
            "X": 1.0
          },
          {
            "Y": 1.0,
            "X": 1.0
          },
          {
            "Y": 1.0,
            "X": 0.0
          }
        ]
      },
      "Relationships": [
        {
          "Type": "CHILD",
          "Ids": [
            "75966e64-81c2-4540-9649-d66ec341cd8f",

```

```

        "bb099c24-8282-464c-a179-8a9fa0a057f0",
        "5ebf522d-f9e4-4dc7-bfae-a288dc094595"
    ]
}
],
"BlockType": "PAGE",
"Id": "247c28ee-b63d-4aeb-9af0-5f7ea8ba109e",
"Page": 1
}
],
"NextToken": "cY1W3eTFvoB0cH7YrKVudI4Gb0H8J0xAYLo8xI/JunCIPWCthaKQ+07n/
ElyutsSy0+1V0ImoTRmP1zw4P0RFtaeV9Bzhnfedpx1YqwB4xaGDA==",
"DocumentMetadata": {
    "Pages": 1
},
"JobStatus": "SUCCEEDED"
}

```

For more information, see *Detecting and Analyzing Text in Multi-Page Documents in the Amazon Textract Developers Guide*

- For API details, see [GetDocumentAnalysis](#) in *AWS CLI Command Reference*.

get-document-text-detection

The following code example shows how to use `get-document-text-detection`.

AWS CLI

To get the results of asynchronous text detection in a multi-page document

The following `get-document-text-detection` example shows how to get the results of asynchronous text detection in a multi-page document.

```

aws textract get-document-text-detection \
  --job-id 57849a3dc627d4df74123dca269d69f7b89329c870c65bb16c9fd63409d200b9 \
  --max-results 1000

```

Output

```

{
  "Blocks": [
    {

```

```
    "Geometry": {
      "BoundingBox": {
        "Width": 1.0,
        "Top": 0.0,
        "Left": 0.0,
        "Height": 1.0
      },
      "Polygon": [
        {
          "Y": 0.0,
          "X": 0.0
        },
        {
          "Y": 0.0,
          "X": 1.0
        },
        {
          "Y": 1.0,
          "X": 1.0
        },
        {
          "Y": 1.0,
          "X": 0.0
        }
      ]
    },
    "Relationships": [
      {
        "Type": "CHILD",
        "Ids": [
          "1b926a34-0357-407b-ac8f-ec473160c6a9",
          "0c35dc17-3605-4c9d-af1a-d9451059df51",
          "dea3db8a-52c2-41c0-b50c-81f66f4aa758"
        ]
      }
    ],
    "BlockType": "PAGE",
    "Id": "84671a5e-8c99-43be-a9d1-6838965da33e",
    "Page": 1
  }
],
  "NextToken": "GcqyoAJuZwuj0T35EN4LCI3EUzMtiLq3nKyFFHvU5q1SaIdEBcSty+njNgoWwuMP/
muqc96S4o5NzDqehhXvhkodMyV050JGyms51srCxibWJw==",
  "DocumentMetadata": {
```

```

    "Pages": 1
  },
  "JobStatus": "SUCCEEDED"
}

```

For more information, see *Detecting and Analyzing Text in Multi-Page Documents* in the *Amazon Textract Developers Guide*

- For API details, see [GetDocumentTextDetection](#) in *AWS CLI Command Reference*.

start-document-analysis

The following code example shows how to use `start-document-analysis`.

AWS CLI

To start analyzing text in a multi-page document

The following `start-document-analysis` example shows how to start asynchronous analysis of text in a multi-page document.

Linux/macOS:

```

aws textract start-document-analysis \
  --document-location '{"S3Object":{"Bucket":"bucket","Name":"document"}}' \
  --feature-types ["TABLES","FORMS"] \
  --notification-channel "SNSTopicArn=arn:snsTopic,RoleArn=roleArn"

```

Windows:

```

aws textract start-document-analysis \
  --document-location '{"S3Object":{"Bucket":"bucket","Name":"document"}}' \
  --feature-types ["TABLES","FORMS"] \
  --region region-name \
  --notification-channel "SNSTopicArn=arn:snsTopic,RoleArn=roleArn"

```

Output:

```

{
  "JobId": "df7cf32ebbd2a5de113535fcf4d921926a701b09b4e7d089f3aebadb41e0712b"
}

```

```
}
```

For more information, see *Detecting and Analyzing Text in Multi-Page Documents* in the *Amazon Textract Developers Guide*

- For API details, see [StartDocumentAnalysis](#) in *AWS CLI Command Reference*.

start-document-text-detection

The following code example shows how to use `start-document-text-detection`.

AWS CLI

To start detecting text in a multi-page document

The following `start-document-text-detection` example shows how to start asynchronous detection of text in a multi-page document.

Linux/macOS:

```
aws textract start-document-text-detection \  
  --document-location '{"S3Object":{"Bucket":"bucket","Name":"document"}}' \  
  --notification-channel "SNSTopicArn=arn:snsTopic,RoleArn=roleARN"
```

Windows:

```
aws textract start-document-text-detection \  
  --document-location "{\"S3Object\":{\"Bucket\":\"bucket\",\"Name\":\"document\"]]\" \  
  --region region-name \  
  --notification-channel "SNSTopicArn=arn:snsTopic,RoleArn=roleArn"
```

Output:

```
{  
  "JobId": "57849a3dc627d4df74123dca269d69f7b89329c870c65bb16c9fd63409d200b9"  
}
```

For more information, see *Detecting and Analyzing Text in Multi-Page Documents* in the *Amazon Textract Developers Guide*

- For API details, see [StartDocumentTextDetection](#) in *AWS CLI Command Reference*.

Amazon Transcribe examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon Transcribe.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-language-model

The following code example shows how to use `create-language-model`.

AWS CLI

Example 1: To create a custom language model using both training and tuning data.

The following `create-language-model` example creates a custom language model. You can use a custom language model to improve transcription performance for domains such as legal, hospitality, finance, and insurance. For `language-code`, enter a valid language code. For `base-model-name`, specify a base model that is best suited for the sample rate of the audio that you want to transcribe with your custom language model. For `model-name`, specify the name that you want to call the custom language model.

```
aws transcribe create-language-model \
  --language-code language-code \
  --base-model-name base-model-name \
  --model-name cli-clm-example \
  --input-data-config S3Uri="s3://DOC-EXAMPLE-BUCKET/Amazon-S3-Prefix-for-
training-data",TuningDataS3Uri="s3://DOC-EXAMPLE-BUCKET/Amazon-S3-Prefix-for-
```

```
tuning-data",DataAccessRoleArn="arn:aws:iam::AWS-account-number:role/IAM-role-with-
permissions-to-create-a-custom-language-model"
```

Output:

```
{
  "LanguageCode": "language-code",
  "BaseModelName": "base-model-name",
  "ModelName": "cli-clm-example",
  "InputDataConfig": {
    "S3Uri": "s3://DOC-EXAMPLE-BUCKET/Amazon-S3-Prefix/",
    "TuningDataS3Uri": "s3://DOC-EXAMPLE-BUCKET/Amazon-S3-Prefix/",
    "DataAccessRoleArn": "arn:aws:iam::AWS-account-number:role/IAM-role-with-
permissions-create-a-custom-language-model"
  },
  "ModelStatus": "IN_PROGRESS"
}
```

For more information, see [Improving Domain-Specific Transcription Accuracy with Custom Language Models](#) in the *Amazon Transcribe Developer Guide*.

Example 2: To create a custom language model using only training data.

The following `create-language-model` example transcribes your audio file. You can use a custom language model to improve transcription performance for domains such as legal, hospitality, finance, and insurance. For `language-code`, enter a valid language code. For `base-model-name`, specify a base model that is best suited for the sample rate of the audio that you want to transcribe with your custom language model. For `model-name`, specify the name that you want to call the custom language model.

```
aws transcribe create-language-model \
  --language-code en-US \
  --base-model-name base-model-name \
  --model-name cli-clm-example \
  --input-data-config S3Uri="s3://DOC-EXAMPLE-BUCKET/Amazon-S3-Prefix-For-
Training-Data",DataAccessRoleArn="arn:aws:iam::AWS-account-number:role/IAM-role-
with-permissions-to-create-a-custom-language-model"
```

Output:

```
{
```



```
"LanguageCode": "en-US",
"BaseModelName": "base-model-name",
"ModelName": "cli-clm-example",
"InputDataConfig": {
  "S3Uri": "s3://DOC-EXAMPLE-BUCKET/Amazon-S3-Prefix-For-Training-Data/",
  "DataAccessRoleArn": "arn:aws:iam::your-AWS-account-number:role/IAM-role-
with-permissions-to-create-a-custom-language-model"
},
"ModelStatus": "IN_PROGRESS"
}
```

For more information, see [Improving Domain-Specific Transcription Accuracy with Custom Language Models](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [CreateLanguageModel](#) in *AWS CLI Command Reference*.

create-medical-vocabulary

The following code example shows how to use `create-medical-vocabulary`.

AWS CLI

To create a medical custom vocabulary

The following `create-medical-vocabulary` example creates a custom vocabulary. To create a custom vocabulary, you must have created a text file with all the terms that you want to transcribe more accurately. For `vocabulary-file-uri`, specify the Amazon Simple Storage Service (Amazon S3) URI of that text file. For `language-code`, specify a language code corresponding to the language of your custom vocabulary. For `vocabulary-name`, specify what you want to call your custom vocabulary.

```
aws transcribe create-medical-vocabulary \
  --vocabulary-name cli-medical-vocab-example \
  --language-code language-code \
  --vocabulary-file-uri https://DOC-EXAMPLE-BUCKET.AWS-Region.amazonaws.com/the-
text-file-for-the-medical-custom-vocabulary.txt
```

Output:

```
{
  "VocabularyName": "cli-medical-vocab-example",
  "LanguageCode": "language-code",
```

```
"VocabularyState": "PENDING"
}
```

For more information, see [Medical Custom Vocabularies](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [CreateMedicalVocabulary](#) in *AWS CLI Command Reference*.

create-vocabulary-filter

The following code example shows how to use `create-vocabulary-filter`.

AWS CLI

To create a vocabulary filter

The following `create-vocabulary-filter` example creates a vocabulary filter that uses a text file that contains a list of words that you wouldn't want to appear in a transcription. For `language-code`, specify the language code corresponding to the language of your vocabulary filter. For `vocabulary-filter-file-uri`, specify the Amazon Simple Storage Service (Amazon S3) URI of the text file. For `vocabulary-filter-name`, specify the name of your vocabulary filter.

```
aws transcribe create-vocabulary-filter \
  --language-code language-code \
  --vocabulary-filter-file-uri s3://DOC-EXAMPLE-BUCKET/vocabulary-filter.txt \
  --vocabulary-filter-name cli-vocabulary-filter-example
```

Output:

```
{
  "VocabularyFilterName": "cli-vocabulary-filter-example",
  "LanguageCode": "language-code"
}
```

For more information, see [Filtering Unwanted Words](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [CreateVocabularyFilter](#) in *AWS CLI Command Reference*.

create-vocabulary

The following code example shows how to use `create-vocabulary`.

AWS CLI

To create a custom vocabulary

The following `create-vocabulary` example creates a custom vocabulary. To create a custom vocabulary, you must have created a text file with all the terms that you want to transcribe more accurately. For `vocabulary-file-uri`, specify the Amazon Simple Storage Service (Amazon S3) URI of that text file. For `language-code`, specify a language code corresponding to the language of your custom vocabulary. For `vocabulary-name`, specify what you want to call your custom vocabulary.

```
aws transcribe create-vocabulary \  
  --language-code language-code \  
  --vocabulary-name cli-vocab-example \  
  --vocabulary-file-uri s3://DOC-EXAMPLE-BUCKET/Amazon-S3-prefix/the-text-file-  
for-the-custom-vocabulary.txt
```

Output:

```
{  
  "VocabularyName": "cli-vocab-example",  
  "LanguageCode": "language-code",  
  "VocabularyState": "PENDING"  
}
```

For more information, see [Custom Vocabularies](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [CreateVocabulary](#) in *AWS CLI Command Reference*.

delete-language-model

The following code example shows how to use `delete-language-model`.

AWS CLI

To delete a custom language model

The following `delete-language-model` example deletes a custom language model.

```
aws transcribe delete-language-model \  
  --model-name model-name
```

This command produces no output.

For more information, see [Improving Domain-Specific Transcription Accuracy with Custom Language Models](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [DeleteLanguageModel](#) in *AWS CLI Command Reference*.

delete-medical-transcription-job

The following code example shows how to use `delete-medical-transcription-job`.

AWS CLI

To delete a medical transcription job

The following `delete-medical-transcription-job` example deletes a medical transcription job.

```
aws transcribe delete-medical-transcription-job \  
  --medical-transcription-job-name medical-transcription-job-name
```

This command produces no output.

For more information, see [DeleteMedicalTranscriptionJob](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [DeleteMedicalTranscriptionJob](#) in *AWS CLI Command Reference*.

delete-medical-vocabulary

The following code example shows how to use `delete-medical-vocabulary`.

AWS CLI

To delete a medical custom vocabulary

The following `delete-medical-vocabulary` example deletes a medical custom vocabulary. For `vocabulary-name`, specify the name of the medical custom vocabulary.

```
aws transcribe delete-vocabulary \  
  --vocabulary-name medical-custom-vocabulary-name
```

This command produces no output.

For more information, see [Medical Custom Vocabularies](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [DeleteMedicalVocabulary](#) in *AWS CLI Command Reference*.

delete-transcription-job

The following code example shows how to use `delete-transcription-job`.

AWS CLI

To delete one of your transcription jobs

The following `delete-transcription-job` example deletes one of your transcription jobs.

```
aws transcribe delete-transcription-job \  
  --transcription-job-name your-transcription-job
```

This command produces no output.

For more information, see [DeleteTranscriptionJob](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [DeleteTranscriptionJob](#) in *AWS CLI Command Reference*.

delete-vocabulary-filter

The following code example shows how to use `delete-vocabulary-filter`.

AWS CLI

To delete a vocabulary filter

The following `delete-vocabulary-filter` example deletes a vocabulary filter.

```
aws transcribe delete-vocabulary-filter \  
  --vocabulary-filter-name vocabulary-filter-name
```

This command produces no output.

For more information, see [Filtering Unwanted Words](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [DeleteVocabularyFilter](#) in *AWS CLI Command Reference*.

delete-vocabulary

The following code example shows how to use delete-vocabulary.

AWS CLI

To delete a custom vocabulary

The following delete-vocabulary example deletes a custom vocabulary.

```
aws transcribe delete-vocabulary \  
  --vocabulary-name vocabulary-name
```

This command produces no output.

For more information, see [Custom Vocabularies](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [DeleteVocabulary](#) in *AWS CLI Command Reference*.

describe-language-model

The following code example shows how to use describe-language-model.

AWS CLI

To get information about a specific custom language model

The following describe-language-model example gets information about a specific custom language model. For example, under BaseModelName you can see whether your model is trained using a NarrowBand or WideBand model. Custom language models with a NarrowBand base model can transcribe audio with a sample rate less than 16 kHz. Language models using a WideBand base model can transcribe audio with a sample rate greater than 16 kHz. The S3Uri parameter indicates the Amazon S3 prefix you've used to access the training data to create the custom language model.

```
aws transcribe describe-language-model \  
  --model-name cli-clm-example
```

Output:

```
{
```

```
"LanguageModel": {
  "ModelName": "cli-clm-example",
  "CreateTime": "2020-09-25T17:57:38.504000+00:00",
  "LastModifiedTime": "2020-09-25T17:57:48.585000+00:00",
  "LanguageCode": "language-code",
  "BaseModelName": "base-model-name",
  "ModelStatus": "IN_PROGRESS",
  "UpgradeAvailability": false,
  "InputDataConfig": {
    "S3Uri": "s3://DOC-EXAMPLE-BUCKET/Amazon-S3-Prefix/",
    "TuningDataS3Uri": "s3://DOC-EXAMPLE-BUCKET/Amazon-S3-Prefix/",
    "DataAccessRoleArn": "arn:aws:iam::AWS-account-number:role/IAM-role-
with-permissions-to-create-a-custom-language-model"
  }
}
```

For more information, see [Improving Domain-Specific Transcription Accuracy with Custom Language Models](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [DescribeLanguageModel](#) in *AWS CLI Command Reference*.

get-medical-transcription-job

The following code example shows how to use `get-medical-transcription-job`.

AWS CLI

To get information about a specific medical transcription job

The following `get-medical-transcription-job` example gets information about a specific medical transcription job. To access the transcription results, use the `TranscriptFileUri` parameter. If you've enabled additional features for the transcription job, you can see them in the `Settings` object. The `Specialty` parameter shows the medical specialty of the provider. The `Type` parameter indicates whether the speech in the transcription job is of a medical conversation, or a medical dictation.

```
aws transcribe get-medical-transcription-job \
  --medical-transcription-job-name vocabulary-dictation-medical-transcription-job
```

Output:

```
{
  "MedicalTranscriptionJob": {
    "MedicalTranscriptionJobName": "vocabulary-dictation-medical-transcription-
job",
    "TranscriptionJobStatus": "COMPLETED",
    "LanguageCode": "en-US",
    "MediaSampleRateHertz": 48000,
    "MediaFormat": "mp4",
    "Media": {
      "MediaFileUri": "s3://Amazon-S3-Prefix/your-audio-file.file-extension"
    },
    "Transcript": {
      "TranscriptFileUri": "https://s3.Region.amazonaws.com/Amazon-S3-Prefix/
vocabulary-dictation-medical-transcription-job.json"
    },
    "StartTime": "2020-09-21T21:17:27.045000+00:00",
    "CreationTime": "2020-09-21T21:17:27.016000+00:00",
    "CompletionTime": "2020-09-21T21:17:59.561000+00:00",
    "Settings": {
      "ChannelIdentification": false,
      "ShowAlternatives": false,
      "VocabularyName": "cli-medical-vocab-example"
    },
    "Specialty": "PRIMARYCARE",
    "Type": "DICTATION"
  }
}
```

For more information, see [Batch Transcription](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [GetMedicalTranscriptionJob](#) in *AWS CLI Command Reference*.

get-medical-vocabulary

The following code example shows how to use `get-medical-vocabulary`.

AWS CLI

To get information about a medical custom vocabulary

The following `get-medical-vocabulary` example gets information on a medical custom vocabulary. You can use the `VocabularyState` parameter to see the processing state of the vocabulary. If it's `READY`, you can use it in the `StartMedicalTranscriptionJob` operation.:


```
aws transcribe get-medical-vocabulary \  
  --vocabulary-name medical-vocab-example
```

Output:

```
{  
  "VocabularyName": "medical-vocab-example",  
  "LanguageCode": "en-US",  
  "VocabularyState": "READY",  
  "LastModifiedTime": "2020-09-19T23:59:04.349000+00:00",  
  "DownloadUri": "https://link-to-download-the-text-file-used-to-create-your-  
  medical-custom-vocabulary"  
}
```

For more information, see [Medical Custom Vocabularies](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [GetMedicalVocabulary](#) in *AWS CLI Command Reference*.

get-transcription-job

The following code example shows how to use `get-transcription-job`.

AWS CLI

To get information about a specific transcription job

The following `get-transcription-job` example gets information about a specific transcription job. To access the transcription results, use the `TranscriptFileUri` parameter. Use the `MediaFileUri` parameter to see which audio file you transcribed with this job. You can use the `Settings` object to see the optional features you've enabled in the transcription job.

```
aws transcribe get-transcription-job \  
  --transcription-job-name your-transcription-job
```

Output:

```
{  
  "TranscriptionJob": {  
    "TranscriptionJobName": "your-transcription-job",
```

```
"TranscriptionJobStatus": "COMPLETED",
"LanguageCode": "language-code",
"MediaSampleRateHertz": 48000,
"MediaFormat": "mp4",
"Media": {
  "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.file-extension"
},
"Transcript": {
  "TranscriptFileUri": "https://Amazon-S3-file-location-of-transcription-
output"
},
"StartTime": "2020-09-18T22:27:23.970000+00:00",
"CreationTime": "2020-09-18T22:27:23.948000+00:00",
"CompletionTime": "2020-09-18T22:28:21.197000+00:00",
"Settings": {
  "ChannelIdentification": false,
  "ShowAlternatives": false
},
"IdentifyLanguage": true,
"IdentifiedLanguageScore": 0.8672199249267578
}
}
```

For more information, see [Getting Started \(AWS Command Line Interface\)](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [GetTranscriptionJob](#) in *AWS CLI Command Reference*.

get-vocabulary-filter

The following code example shows how to use `get-vocabulary-filter`.

AWS CLI

To get information about a vocabulary filter

The following `get-vocabulary-filter` example gets information about a vocabulary filter. You can use the `DownloadUri` parameter to get the list of words you used to create the vocabulary filter.

```
aws transcribe get-vocabulary-filter \
  --vocabulary-filter-name testFilter
```

Output:

```
{
  "VocabularyFilterName": "testFilter",
  "LanguageCode": "language-code",
  "LastModifiedTime": "2020-05-07T22:39:32.147000+00:00",
  "DownloadUri": "https://Amazon-S3-location-to-download-your-vocabulary-filter"
}
```

For more information, see [Filter Unwanted Words](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [GetVocabularyFilter](#) in *AWS CLI Command Reference*.

get-vocabulary

The following code example shows how to use `get-vocabulary`.

AWS CLI**To get information about a custom vocabulary**

The following `get-vocabulary` example gets information on a previously created custom vocabulary.

```
aws transcribe get-vocabulary \
  --vocabulary-name cli-vocab-1
```

Output:

```
{
  "VocabularyName": "cli-vocab-1",
  "LanguageCode": "language-code",
  "VocabularyState": "READY",
  "LastModifiedTime": "2020-09-19T23:22:32.836000+00:00",
  "DownloadUri": "https://link-to-download-the-text-file-used-to-create-your-
custom-vocabulary"
}
```

For more information, see [Custom Vocabularies](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [GetVocabulary](#) in *AWS CLI Command Reference*.

list-language-models

The following code example shows how to use `list-language-models`.

AWS CLI

To list your custom language models

The following `list-language-models` example lists the custom language models associated with your AWS account and Region. You can use the `S3Uri` and `TuningDataS3Uri` parameters to find the Amazon S3 prefixes you've used as your training data, or your tuning data. The `BaseModelName` tells you whether you've used a `NarrowBand`, or `WideBand` model to create a custom language model. You can transcribe audio with a sample rate of less than 16 kHz with a custom language model using a `NarrowBand` base model. You can transcribe audio 16 kHz or greater with a custom language model using a `WideBand` base model. The `ModelStatus` parameter shows whether you can use the custom language model in a transcription job. If the value is `COMPLETED`, you can use it in a transcription job.

```
aws transcribe list-language-models
```

Output:

```
{
  "Models": [
    {
      "ModelName": "cli-clm-2",
      "CreateTime": "2020-09-25T17:57:38.504000+00:00",
      "LastModifiedTime": "2020-09-25T17:57:48.585000+00:00",
      "LanguageCode": "language-code",
      "BaseModelName": "WideBand",
      "ModelStatus": "IN_PROGRESS",
      "UpgradeAvailability": false,
      "InputDataConfig": {
        "S3Uri": "s3://DOC-EXAMPLE-BUCKET/clm-training-data/",
        "TuningDataS3Uri": "s3://DOC-EXAMPLE-BUCKET/clm-tuning-data/",
        "DataAccessRoleArn": "arn:aws:iam::AWS-account-number:role/IAM-role-used-to-create-the-custom-language-model"
      }
    },
    {
      "ModelName": "cli-clm-1",
      "CreateTime": "2020-09-25T17:16:01.835000+00:00",
```

```

    "LastModifiedTime": "2020-09-25T17:16:15.555000+00:00",
    "LanguageCode": "language-code",
    "BaseModelName": "WideBand",
    "ModelStatus": "IN_PROGRESS",
    "UpgradeAvailability": false,
    "InputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/clm-training-data/",
      "DataAccessRoleArn": "arn:aws:iam::AWS-account-number:role/IAM-role-used-to-create-the-custom-language-model"
    }
  },
  {
    "ModelName": "clm-console-1",
    "CreateTime": "2020-09-24T19:26:28.076000+00:00",
    "LastModifiedTime": "2020-09-25T04:25:22.271000+00:00",
    "LanguageCode": "language-code",
    "BaseModelName": "NarrowBand",
    "ModelStatus": "COMPLETED",
    "UpgradeAvailability": false,
    "InputDataConfig": {
      "S3Uri": "s3://DOC-EXAMPLE-BUCKET/clm-training-data/",
      "DataAccessRoleArn": "arn:aws:iam::AWS-account-number:role/IAM-role-used-to-create-the-custom-language-model"
    }
  }
]
}

```

For more information, see [Improving Domain-Specific Transcription Accuracy with Custom Language Models](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [ListLanguageModels](#) in *AWS CLI Command Reference*.

list-medical-transcription-jobs

The following code example shows how to use `list-medical-transcription-jobs`.

AWS CLI

To list your medical transcription jobs

The following `list-medical-transcription-jobs` example lists the medical transcription jobs associated with your AWS account and Region. To get more information about a particular

transcription job, copy the value of a `MedicalTranscriptionJobName` parameter in the transcription output, and specify that value for the `MedicalTranscriptionJobName` option of the `get-medical-transcription-job` command. To see more of your transcription jobs, copy the value of the `NextToken` parameter, run the `list-medical-transcription-jobs` command again, and specify that value in the `--next-token` option.

```
aws transcribe list-medical-transcription-jobs
```

Output:

```
{
  "NextToken": "3/PblzkiGhzjER3KHuQt2fmbPLF7cDYafjFMEoGn440N/
gsuUSTIkGyanvRE6WMXfd/ZTEc2EZj+P9eii/
z102FDYli6RLI0WoRX4RwMisVrh9G0Kie0Y8ikBCdtq1ZB10Wa9McC+eb0l
+LaDtZPC4u6ttoHLRl1EfzqstHXSgapXg3tEBtm9piIaPB6MOM5BB6t86+qtmocTR/
qrteHZBBudhTfbCwhsxaqujHiiUvFdm3BQbKKWIW06yV9b+4f38oD2lVIan
+vfUs3gBYA15VTDmXXzQPBQ0HPjtwmFI+IWX15nSUjWuN3TUylHgPWzDaYT8qBtu0Z+3UG4V6b
+K2CC0XszXg5rBq9hYgNzy4XoFh/6s5DoSznq49Q9xHgHdT2yBADFmvFK7myZBsJ75+2vQZ0SVpWUPy3WT/32zFAcoEL
+mFYfUjtTZ8n/jq7aQEjQ42A
+X/7K6Jg0cdVPtEg8P1Dr5kgYYG3q30mYXX37U3FZuJmnTI63VtIXsNn0U5eGoY0btpk00Nq9UkzgSJxqj84ZD5n
+S0EGy9ZUYBJRRcGeYUM3Q4DbSJfUwSAqcFdLIWZdp8qIREMQIBWy7BLwSdyqsQo2vRrd53hm5aWM7SVf6pPq6X/
IXR5+1eU00D8/coaTT4ES2DerbV6RkV4o0VT1d0SdVX/
MmtkNG8nYj8PqU07w7988quh1ZP6D80veJS1q73tUUR9MjnGernW2tAnvnLNhdefBcD
+sZVfyq3iBMFY7wTy1P1G6NqW9GrYDY0X3tTPWlD7phpbVSyKrh/
PdYrps5UxnsGoA1b7L/FfAXDfUoGrGUB4N3JsPYXX9D++g+6gV1qBBs/
Wff934aKqfD6UTggm/zV3GA0WiBpfvAZRvEb924i6yGHyMC7y5401ZAwSBupmI
+FFd13CaP04kN1vJlth6aM5vUPXg4BpyUhtbRhwD/KxCvf9K0tLJGyL1A==",
  "MedicalTranscriptionJobSummaries": [
    {
      "MedicalTranscriptionJobName": "vocabulary-dictation-medical-
transcription-job",
      "CreationTime": "2020-09-21T21:17:27.016000+00:00",
      "StartTime": "2020-09-21T21:17:27.045000+00:00",
      "CompletionTime": "2020-09-21T21:17:59.561000+00:00",
      "LanguageCode": "en-US",
      "TranscriptionJobStatus": "COMPLETED",
      "OutputLocationType": "CUSTOMER_BUCKET",
      "Specialty": "PRIMARYCARE",
      "Type": "DICTATION"
    },
    {

```

```
    "MedicalTranscriptionJobName": "alternatives-dictation-medical-
transcription-job",
    "CreationTime": "2020-09-21T21:01:14.569000+00:00",
    "StartTime": "2020-09-21T21:01:14.592000+00:00",
    "CompletionTime": "2020-09-21T21:01:43.606000+00:00",
    "LanguageCode": "en-US",
    "TranscriptionJobStatus": "COMPLETED",
    "OutputLocationType": "CUSTOMER_BUCKET",
    "Specialty": "PRIMARYCARE",
    "Type": "DICTATION"
  },
  {
    "MedicalTranscriptionJobName": "alternatives-conversation-medical-
transcription-job",
    "CreationTime": "2020-09-21T19:09:18.171000+00:00",
    "StartTime": "2020-09-21T19:09:18.199000+00:00",
    "CompletionTime": "2020-09-21T19:10:22.516000+00:00",
    "LanguageCode": "en-US",
    "TranscriptionJobStatus": "COMPLETED",
    "OutputLocationType": "CUSTOMER_BUCKET",
    "Specialty": "PRIMARYCARE",
    "Type": "CONVERSATION"
  },
  {
    "MedicalTranscriptionJobName": "speaker-id-conversation-medical-
transcription-job",
    "CreationTime": "2020-09-21T18:43:37.157000+00:00",
    "StartTime": "2020-09-21T18:43:37.265000+00:00",
    "CompletionTime": "2020-09-21T18:44:21.192000+00:00",
    "LanguageCode": "en-US",
    "TranscriptionJobStatus": "COMPLETED",
    "OutputLocationType": "CUSTOMER_BUCKET",
    "Specialty": "PRIMARYCARE",
    "Type": "CONVERSATION"
  },
  {
    "MedicalTranscriptionJobName": "multichannel-conversation-medical-
transcription-job",
    "CreationTime": "2020-09-20T23:46:44.053000+00:00",
    "StartTime": "2020-09-20T23:46:44.081000+00:00",
    "CompletionTime": "2020-09-20T23:47:35.851000+00:00",
    "LanguageCode": "en-US",
    "TranscriptionJobStatus": "COMPLETED",
    "OutputLocationType": "CUSTOMER_BUCKET",
```

```
        "Specialty": "PRIMARYCARE",
        "Type": "CONVERSATION"
    }
]
}
```

For more information, see <https://docs.aws.amazon.com/transcribe/latest/dg/batch-med-transcription.html> in the *Amazon Transcribe Developer Guide*.

- For API details, see [ListMedicalTranscriptionJobs](#) in *AWS CLI Command Reference*.

list-medical-vocabularies

The following code example shows how to use `list-medical-vocabularies`.

AWS CLI

To list your medical custom vocabularies

The following `list-medical-vocabularies` example lists the medical custom vocabularies associated with your AWS account and Region. To get more information about a particular transcription job, copy the value of a `MedicalTranscriptionJobName` parameter in the transcription output, and specify that value for the `MedicalTranscriptionJobName` option of the `get-medical-transcription-job` command. To see more of your transcription jobs, copy the value of the `NextToken` parameter, run the `list-medical-transcription-jobs` command again, and specify that value in the `--next-token` option.

```
aws transcribe list-medical-vocabularies
```

Output:

```
{
  "Vocabularies": [
    {
      "VocabularyName": "cli-medical-vocab-2",
      "LanguageCode": "en-US",
      "LastModifiedTime": "2020-09-21T21:44:59.521000+00:00",
      "VocabularyState": "READY"
    },
    {
      "VocabularyName": "cli-medical-vocab-1",
```



```
        "LanguageCode": "en-US",
        "LastModifiedTime": "2020-09-19T23:59:04.349000+00:00",
        "VocabularyState": "READY"
    }
]
}
```

For more information, see [Medical Custom Vocabularies](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [ListMedicalVocabularies](#) in *AWS CLI Command Reference*.

list-transcription-jobs

The following code example shows how to use `list-transcription-jobs`.

AWS CLI

To list your transcription jobs

The following `list-transcription-jobs` example lists the transcription jobs associated with your AWS account and Region.

```
aws transcribe list-transcription-jobs
```

Output:

```
{
  "NextToken": "NextToken",
  "TranscriptionJobSummaries": [
    {
      "TranscriptionJobName": "speak-id-job-1",
      "CreationTime": "2020-08-17T21:06:15.391000+00:00",
      "StartTime": "2020-08-17T21:06:15.416000+00:00",
      "CompletionTime": "2020-08-17T21:07:05.098000+00:00",
      "LanguageCode": "language-code",
      "TranscriptionJobStatus": "COMPLETED",
      "OutputLocationType": "SERVICE_BUCKET"
    },
    {
      "TranscriptionJobName": "job-1",
      "CreationTime": "2020-08-17T20:50:24.207000+00:00",
      "StartTime": "2020-08-17T20:50:24.230000+00:00",

```

```

    "CompletionTime": "2020-08-17T20:52:18.737000+00:00",
    "LanguageCode": "language-code",
    "TranscriptionJobStatus": "COMPLETED",
    "OutputLocationType": "SERVICE_BUCKET"
  },
  {
    "TranscriptionJobName": "sdk-test-job-4",
    "CreationTime": "2020-08-17T20:32:27.917000+00:00",
    "StartTime": "2020-08-17T20:32:27.956000+00:00",
    "CompletionTime": "2020-08-17T20:33:15.126000+00:00",
    "LanguageCode": "language-code",
    "TranscriptionJobStatus": "COMPLETED",
    "OutputLocationType": "SERVICE_BUCKET"
  },
  {
    "TranscriptionJobName": "Diarization-speak-id",
    "CreationTime": "2020-08-10T22:10:09.066000+00:00",
    "StartTime": "2020-08-10T22:10:09.116000+00:00",
    "CompletionTime": "2020-08-10T22:26:48.172000+00:00",
    "LanguageCode": "language-code",
    "TranscriptionJobStatus": "COMPLETED",
    "OutputLocationType": "SERVICE_BUCKET"
  },
  {
    "TranscriptionJobName": "your-transcription-job-name",
    "CreationTime": "2020-07-29T17:45:09.791000+00:00",
    "StartTime": "2020-07-29T17:45:09.826000+00:00",
    "CompletionTime": "2020-07-29T17:46:20.831000+00:00",
    "LanguageCode": "language-code",
    "TranscriptionJobStatus": "COMPLETED",
    "OutputLocationType": "SERVICE_BUCKET"
  }
]
}

```

For more information, see [Getting Started \(AWS Command Line Interface\)](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [ListTranscriptionJobs](#) in *AWS CLI Command Reference*.

list-vocabularies

The following code example shows how to use `list-vocabularies`.

AWS CLI

To list your custom vocabularies

The following `list-vocabularies` example lists the custom vocabularies associated with your AWS account and Region.

```
aws transcribe list-vocabularies
```

Output:

```
{
  "NextToken": "NextToken",
  "Vocabularies": [
    {
      "VocabularyName": "ards-test-1",
      "LanguageCode": "language-code",
      "LastModifiedTime": "2020-04-27T22:00:27.330000+00:00",
      "VocabularyState": "READY"
    },
    {
      "VocabularyName": "sample-test",
      "LanguageCode": "language-code",
      "LastModifiedTime": "2020-04-24T23:04:11.044000+00:00",
      "VocabularyState": "READY"
    },
    {
      "VocabularyName": "CRLF-to-LF-test-3-1",
      "LanguageCode": "language-code",
      "LastModifiedTime": "2020-04-24T22:12:22.277000+00:00",
      "VocabularyState": "READY"
    },
    {
      "VocabularyName": "CRLF-to-LF-test-2",
      "LanguageCode": "language-code",
      "LastModifiedTime": "2020-04-24T21:53:50.455000+00:00",
      "VocabularyState": "READY"
    },
    {
      "VocabularyName": "CRLF-to-LF-1-1",
      "LanguageCode": "language-code",
      "LastModifiedTime": "2020-04-24T21:39:33.356000+00:00",
      "VocabularyState": "READY"
    }
  ]
}
```

```
    }  
  ]  
}
```

For more information, see [Custom Vocabularies](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [ListVocabularies](#) in *AWS CLI Command Reference*.

list-vocabulary-filters

The following code example shows how to use `list-vocabulary-filters`.

AWS CLI

To list your vocabulary filters

The following `list-vocabulary-filters` example lists the vocabulary filters associated with your AWS account and Region.

```
aws transcribe list-vocabulary-filters
```

Output:

```
{  
  "NextToken": "NextToken": [  
    {  
      "VocabularyFilterName": "testFilter",  
      "LanguageCode": "language-code",  
      "LastModifiedTime": "2020-05-07T22:39:32.147000+00:00"  
    },  
    {  
      "VocabularyFilterName": "testFilter2",  
      "LanguageCode": "language-code",  
      "LastModifiedTime": "2020-05-21T23:29:35.174000+00:00"  
    },  
    {  
      "VocabularyFilterName": "filter2",  
      "LanguageCode": "language-code",  
      "LastModifiedTime": "2020-05-08T20:18:26.426000+00:00"  
    },  
    {  
      "VocabularyFilterName": "filter-review",
```

```
    "LanguageCode": "language-code",
    "LastModifiedTime": "2020-06-03T18:52:30.448000+00:00"
  },
  {
    "VocabularyFilterName": "crlf-filt",
    "LanguageCode": "language-code",
    "LastModifiedTime": "2020-05-22T19:42:42.737000+00:00"
  }
]
}
```

For more information, see [Filtering Unwanted Words](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [ListVocabularyFilters](#) in *AWS CLI Command Reference*.

start-medical-transcription-job

The following code example shows how to use `start-medical-transcription-job`.

AWS CLI

Example 1: To transcribe a medical dictation stored as an audio file

The following `start-medical-transcription-job` example transcribes an audio file. You specify the location of the transcription output in the `OutputBucketName` parameter.

```
aws transcribe start-medical-transcription-job \
  --cli-input-json file://myfile.json
```

Contents of `myfile.json`:

```
{
  "MedicalTranscriptionJobName": "simple-dictation-medical-transcription-job",
  "LanguageCode": "language-code",
  "Specialty": "PRIMARYCARE",
  "Type": "DICTATION",
  "OutputBucketName": "DOC-EXAMPLE-BUCKET",
  "Media": {
    "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.extension"
  }
}
```

Output:

```
{
  "MedicalTranscriptionJob": {
    "MedicalTranscriptionJobName": "simple-dictation-medical-transcription-job",
    "TranscriptionJobStatus": "IN_PROGRESS",
    "LanguageCode": "language-code",
    "Media": {
      "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.extension"
    },
    "StartTime": "2020-09-20T00:35:22.256000+00:00",
    "CreationTime": "2020-09-20T00:35:22.218000+00:00",
    "Specialty": "PRIMARYCARE",
    "Type": "DICTATION"
  }
}
```

For more information, see [Batch Transcription Overview](#) in the *Amazon Transcribe Developer Guide*.

Example 2: To transcribe a clinician-patient dialogue stored as an audio file

The following `start-medical-transcription-job` example transcribes an audio file containing a clinician-patient dialogue. You specify the location of the transcription output in the `OutputBucketName` parameter.

```
aws transcribe start-medical-transcription-job \
  --cli-input-json file://mysecondfile.json
```

Contents of `mysecondfile.json`:

```
{
  "MedicalTranscriptionJobName": "simple-dictation-medical-transcription-job",
  "LanguageCode": "language-code",
  "Specialty": "PRIMARYCARE",
  "Type": "CONVERSATION",
  "OutputBucketName": "DOC-EXAMPLE-BUCKET",
  "Media": {
    "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.extension"
  }
}
```

Output:

```
{
  "MedicalTranscriptionJob": {
    "MedicalTranscriptionJobName": "simple-conversation-medical-transcription-
job",
    "TranscriptionJobStatus": "IN_PROGRESS",
    "LanguageCode": "language-code",
    "Media": {
      "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.extension"
    },
    "StartTime": "2020-09-20T23:19:49.965000+00:00",
    "CreationTime": "2020-09-20T23:19:49.941000+00:00",
    "Specialty": "PRIMARYCARE",
    "Type": "CONVERSATION"
  }
}
```

For more information, see [Batch Transcription Overview](#) in the *Amazon Transcribe Developer Guide*.

Example 3: To transcribe a multichannel audio file of a clinician-patient dialogue

The following `start-medical-transcription-job` example transcribes the audio from each channel in the audio file and merges the separate transcriptions from each channel into a single transcription output. You specify the location of the transcription output in the `OutputBucketName` parameter.

```
aws transcribe start-medical-transcription-job \
  --cli-input-json file://mythirdfile.json
```

Contents of `mythirdfile.json`:

```
{
  "MedicalTranscriptionJobName": "multichannel-conversation-medical-transcription-
job",
  "LanguageCode": "language-code",
  "Specialty": "PRIMARYCARE",
  "Type": "CONVERSATION",
  "OutputBucketName": "DOC-EXAMPLE-BUCKET",
  "Media": {
```

```

    "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.extension"
  },
  "Settings":{
    "ChannelIdentification": true
  }
}

```

Output:

```

{
  "MedicalTranscriptionJob": {
    "MedicalTranscriptionJobName": "multichannel-conversation-medical-
transcription-job",
    "TranscriptionJobStatus": "IN_PROGRESS",
    "LanguageCode": "language-code",
    "Media": {
      "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.extension"
    },
    "StartTime": "2020-09-20T23:46:44.081000+00:00",
    "CreationTime": "2020-09-20T23:46:44.053000+00:00",
    "Settings": {
      "ChannelIdentification": true
    },
    "Specialty": "PRIMARYCARE",
    "Type": "CONVERSATION"
  }
}

```

For more information, see [Channel Identification](#) in the *Amazon Transcribe Developer Guide*.

Example 4: To transcribe an audio file of a clinician-patient dialogue and identify the speakers in the transcription output

The following `start-medical-transcription-job` example transcribes an audio file and labels the speech of each speaker in the transcription output. You specify the location of the transcription output in the `OutputBucketName` parameter.

```

aws transcribe start-medical-transcription-job \
  --cli-input-json file://myfourthfile.json

```

Contents of `myfourthfile.json`:


```
{
  "MedicalTranscriptionJobName": "speaker-id-conversation-medical-transcription-
job",
  "LanguageCode": "language-code",
  "Specialty": "PRIMARYCARE",
  "Type": "CONVERSATION",
  "OutputBucketName": "DOC-EXAMPLE-BUCKET",
  "Media": {
    "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.extension"
  },
  "Settings": {
    "ShowSpeakerLabels": true,
    "MaxSpeakerLabels": 2
  }
}
```

Output:

```
{
  "MedicalTranscriptionJob": {
    "MedicalTranscriptionJobName": "speaker-id-conversation-medical-
transcription-job",
    "TranscriptionJobStatus": "IN_PROGRESS",
    "LanguageCode": "language-code",
    "Media": {
      "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.extension"
    },
    "StartTime": "2020-09-21T18:43:37.265000+00:00",
    "CreationTime": "2020-09-21T18:43:37.157000+00:00",
    "Settings": {
      "ShowSpeakerLabels": true,
      "MaxSpeakerLabels": 2
    },
    "Specialty": "PRIMARYCARE",
    "Type": "CONVERSATION"
  }
}
```

For more information, see [Identifying Speakers](#) in the *Amazon Transcribe Developer Guide*.

Example 5: To transcribe a medical conversation stored as an audio file with up to two transcription alternatives

The following `start-medical-transcription-job` example creates up to two alternative transcriptions from a single audio file. Every transcription has a level of confidence associated with it. By default, Amazon Transcribe returns the transcription with the highest confidence level. You can specify that Amazon Transcribe return additional transcriptions with lower confidence levels. You specify the location of the transcription output in the `OutputBucketName` parameter.

```
aws transcribe start-medical-transcription-job \  
  --cli-input-json file://myfifthfile.json
```

Contents of `myfifthfile.json`:

```
{  
  "MedicalTranscriptionJobName": "alternatives-conversation-medical-transcription-  
job",  
  "LanguageCode": "language-code",  
  "Specialty": "PRIMARYCARE",  
  "Type": "CONVERSATION",  
  "OutputBucketName": "DOC-EXAMPLE-BUCKET",  
  "Media": {  
    "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.extension"  
  },  
  "Settings": {  
    "ShowAlternatives": true,  
    "MaxAlternatives": 2  
  }  
}
```

Output:

```
{  
  "MedicalTranscriptionJob": {  
    "MedicalTranscriptionJobName": "alternatives-conversation-medical-  
transcription-job",  
    "TranscriptionJobStatus": "IN_PROGRESS",  
    "LanguageCode": "language-code",  
    "Media": {  
      "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.extension"  
    },  
    "StartTime": "2020-09-21T19:09:18.199000+00:00",  
    "CreationTime": "2020-09-21T19:09:18.171000+00:00",  
  }  
}
```

```
    "Settings": {
      "ShowAlternatives": true,
      "MaxAlternatives": 2
    },
    "Specialty": "PRIMARYCARE",
    "Type": "CONVERSATION"
  }
}
```

For more information, see [Alternative Transcriptions](#) in the *Amazon Transcribe Developer Guide*.

Example 6: To transcribe an audio file of a medical dictation with up to two alternative transcriptions

The following `start-medical-transcription-job` example transcribes an audio file and uses a vocabulary filter to mask any unwanted words. You specify the location of the transcription output in the `OutputBucketName` parameter.

```
aws transcribe start-medical-transcription-job \
  --cli-input-json file://mysixthfile.json
```

Contents of `mysixthfile.json`:

```
{
  "MedicalTranscriptionJobName": "alternatives-conversation-medical-transcription-
job",
  "LanguageCode": "language-code",
  "Specialty": "PRIMARYCARE",
  "Type": "DICTATION",
  "OutputBucketName": "DOC-EXAMPLE-BUCKET",
  "Media": {
    "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.extension"
  },
  "Settings": {
    "ShowAlternatives": true,
    "MaxAlternatives": 2
  }
}
```

Output:

```
{
```

```

    "MedicalTranscriptionJob": {
      "MedicalTranscriptionJobName": "alternatives-dictation-medical-
transcription-job",
      "TranscriptionJobStatus": "IN_PROGRESS",
      "LanguageCode": "language-code",
      "Media": {
        "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.extension"
      },
      "StartTime": "2020-09-21T21:01:14.592000+00:00",
      "CreationTime": "2020-09-21T21:01:14.569000+00:00",
      "Settings": {
        "ShowAlternatives": true,
        "MaxAlternatives": 2
      },
      "Specialty": "PRIMARYCARE",
      "Type": "DICTATION"
    }
  }
}

```

For more information, see [Alternative Transcriptions](#) in the *Amazon Transcribe Developer Guide*.

Example 7: To transcribe an audio file of a medical dictation with increased accuracy by using a custom vocabulary

The following `start-medical-transcription-job` example transcribes an audio file and uses a medical custom vocabulary you've previously created to increase the transcription accuracy. You specify the location of the transcription output in the `OutputBucketName` parameter.

```

aws transcribe start-transcription-job \
  --cli-input-json file://myseventhfile.json

```

Contents of `mysixthfile.json`:

```

{
  "MedicalTranscriptionJobName": "vocabulary-dictation-medical-transcription-job",
  "LanguageCode": "language-code",
  "Specialty": "PRIMARYCARE",
  "Type": "DICTATION",
  "OutputBucketName": "DOC-EXAMPLE-BUCKET",
  "Media": {
    "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.extension"
  }
}

```

```
  },
  "Settings":{
    "VocabularyName": "cli-medical-vocab-1"
  }
}
```

Output:

```
{
  "MedicalTranscriptionJob": {
    "MedicalTranscriptionJobName": "vocabulary-dictation-medical-transcription-
job",
    "TranscriptionJobStatus": "IN_PROGRESS",
    "LanguageCode": "language-code",
    "Media": {
      "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.extension"
    },
    "StartTime": "2020-09-21T21:17:27.045000+00:00",
    "CreationTime": "2020-09-21T21:17:27.016000+00:00",
    "Settings": {
      "VocabularyName": "cli-medical-vocab-1"
    },
    "Specialty": "PRIMARYCARE",
    "Type": "DICTATION"
  }
}
```

For more information, see [Medical Custom Vocabularies](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [StartMedicalTranscriptionJob](#) in *AWS CLI Command Reference*.

start-transcription-job

The following code example shows how to use `start-transcription-job`.

AWS CLI

Example 1: To transcribe an audio file

The following `start-transcription-job` example transcribes your audio file.

```
aws transcribe start-transcription-job \
```

```
--cli-input-json file://myfile.json
```

Contents of `myfile.json`:

```
{
  "TranscriptionJobName": "cli-simple-transcription-job",
  "LanguageCode": "the-language-of-your-transcription-job",
  "Media": {
    "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/Amazon-S3-prefix/your-media-file-
name.file-extension"
  }
}
```

For more information, see [Getting Started \(AWS Command Line Interface\)](#) in the *Amazon Transcribe Developer Guide*.

Example 2: To transcribe a multi-channel audio file

The following `start-transcription-job` example transcribes your multi-channel audio file.

```
aws transcribe start-transcription-job \
  --cli-input-json file://mysecondfile.json
```

Contents of `mysecondfile.json`:

```
{
  "TranscriptionJobName": "cli-channelid-job",
  "LanguageCode": "the-language-of-your-transcription-job",
  "Media": {
    "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/Amazon-S3-prefix/your-media-file-
name.file-extension"
  },
  "Settings":{
    "ChannelIdentification":true
  }
}
```

Output:

```
{
```

```
"TranscriptionJob": {
  "TranscriptionJobName": "cli-channelid-job",
  "TranscriptionJobStatus": "IN_PROGRESS",
  "LanguageCode": "the-language-of-your-transcription-job",
  "Media": {
    "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/Amazon-S3-prefix/your-media-
file-name.file-extension"
  },
  "StartTime": "2020-09-17T16:07:56.817000+00:00",
  "CreationTime": "2020-09-17T16:07:56.784000+00:00",
  "Settings": {
    "ChannelIdentification": true
  }
}
```

For more information, see [Transcribing Multi-Channel Audio](#) in the *Amazon Transcribe Developer Guide*.

Example 3: To transcribe an audio file and identify the different speakers

The following `start-transcription-job` example transcribes your audio file and identifies the speakers in the transcription output.

```
aws transcribe start-transcription-job \
  --cli-input-json file://mythirdfile.json
```

Contents of `mythirdfile.json`:

```
{
  "TranscriptionJobName": "cli-speakerid-job",
  "LanguageCode": "the-language-of-your-transcription-job",
  "Media": {
    "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/Amazon-S3-prefix/your-media-file-
name.file-extension"
  },
  "Settings":{
    "ShowSpeakerLabels": true,
    "MaxSpeakerLabels": 2
  }
}
```

Output:

```
{
  "TranscriptionJob": {
    "TranscriptionJobName": "cli-speakerid-job",
    "TranscriptionJobStatus": "IN_PROGRESS",
    "LanguageCode": "the-language-of-your-transcription-job",
    "Media": {
      "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/Amazon-S3-prefix/your-media-
file-name.file-extension"
    },
    "StartTime": "2020-09-17T16:22:59.696000+00:00",
    "CreationTime": "2020-09-17T16:22:59.676000+00:00",
    "Settings": {
      "ShowSpeakerLabels": true,
      "MaxSpeakerLabels": 2
    }
  }
}
```

For more information, see [Identifying Speakers](#) in the *Amazon Transcribe Developer Guide*.

Example 4: To transcribe an audio file and mask any unwanted words in the transcription output

The following `start-transcription-job` example transcribes your audio file and uses a vocabulary filter you've previously created to mask any unwanted words.

```
aws transcribe start-transcription-job \
  --cli-input-json file://myfourthfile.json
```

Contents of `myfourthfile.json`:

```
{
  "TranscriptionJobName": "cli-filter-mask-job",
  "LanguageCode": "the-language-of-your-transcription-job",
  "Media": {
    "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/Amazon-S3-prefix/your-media-file-
name.file-extension"
  },
  "Settings":{
```



```

    "VocabularyFilterName": "your-vocabulary-filter",
    "VocabularyFilterMethod": "mask"
  }
}

```

Output:

```

{
  "TranscriptionJob": {
    "TranscriptionJobName": "cli-filter-mask-job",
    "TranscriptionJobStatus": "IN_PROGRESS",
    "LanguageCode": "the-language-of-your-transcription-job",
    "Media": {
      "MediaFileUri": "s3://Amazon-S3-Prefix/your-media-file.file-extension"
    },
    "StartTime": "2020-09-18T16:36:18.568000+00:00",
    "CreationTime": "2020-09-18T16:36:18.547000+00:00",
    "Settings": {
      "VocabularyFilterName": "your-vocabulary-filter",
      "VocabularyFilterMethod": "mask"
    }
  }
}

```

For more information, see [Filtering Transcriptions](#) in the *Amazon Transcribe Developer Guide*.

Example 5: To transcribe an audio file and remove any unwanted words in the transcription output

The following `start-transcription-job` example transcribes your audio file and uses a vocabulary filter you've previously created to mask any unwanted words.

```

aws transcribe start-transcription-job \
  --cli-input-json file://myfifthfile.json

```

Contents of `myfifthfile.json`:

```

{
  "TranscriptionJobName": "cli-filter-remove-job",
  "LanguageCode": "the-language-of-your-transcription-job",
  "Media": {

```

```

    "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/Amazon-S3-prefix/your-media-file-
name.file-extension"
  },
  "Settings":{
    "VocabularyFilterName": "your-vocabulary-filter",
    "VocabularyFilterMethod": "remove"
  }
}

```

Output:

```

{
  "TranscriptionJob": {
    "TranscriptionJobName": "cli-filter-remove-job",
    "TranscriptionJobStatus": "IN_PROGRESS",
    "LanguageCode": "the-language-of-your-transcription-job",
    "Media": {
      "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/Amazon-S3-prefix/your-media-
file-name.file-extension"
    },
    "StartTime": "2020-09-18T16:36:18.568000+00:00",
    "CreationTime": "2020-09-18T16:36:18.547000+00:00",
    "Settings": {
      "VocabularyFilterName": "your-vocabulary-filter",
      "VocabularyFilterMethod": "remove"
    }
  }
}

```

For more information, see [Filtering Transcriptions](#) in the *Amazon Transcribe Developer Guide*.

Example 6: To transcribe an audio file with increased accuracy using a custom vocabulary

The following `start-transcription-job` example transcribes your audio file and uses a vocabulary filter you've previously created to mask any unwanted words.

```

aws transcribe start-transcription-job \
  --cli-input-json file://mysixthfile.json

```

Contents of `mysixthfile.json`:

```

{

```

```

    "TranscriptionJobName": "cli-vocab-job",
    "LanguageCode": "the-language-of-your-transcription-job",
    "Media": {
      "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/Amazon-S3-prefix/your-media-file-
name.file-extension"
    },
    "Settings":{
      "VocabularyName": "your-vocabulary"
    }
  }
}

```

Output:

```

{
  "TranscriptionJob": {
    "TranscriptionJobName": "cli-vocab-job",
    "TranscriptionJobStatus": "IN_PROGRESS",
    "LanguageCode": "the-language-of-your-transcription-job",
    "Media": {
      "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/Amazon-S3-prefix/your-media-
file-name.file-extension"
    },
    "StartTime": "2020-09-18T16:36:18.568000+00:00",
    "CreationTime": "2020-09-18T16:36:18.547000+00:00",
    "Settings": {
      "VocabularyName": "your-vocabulary"
    }
  }
}

```

For more information, see [Filtering Transcriptions](#) in the *Amazon Transcribe Developer Guide*.

Example 7: To identify the language of an audio file and transcribe it

The following `start-transcription-job` example transcribes your audio file and uses a vocabulary filter you've previously created to mask any unwanted words.

```

aws transcribe start-transcription-job \
  --cli-input-json file://myseventhfile.json

```

Contents of `myseventhfile.json`:

```
{
  "TranscriptionJobName": "cli-identify-language-transcription-job",
  "IdentifyLanguage": true,
  "Media": {
    "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/Amazon-S3-prefix/your-media-file-
name.file-extension"
  }
}
```

Output:

```
{
  "TranscriptionJob": {
    "TranscriptionJobName": "cli-identify-language-transcription-job",
    "TranscriptionJobStatus": "IN_PROGRESS",
    "Media": {
      "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/Amazon-S3-prefix/your-media-
file-name.file-extension"
    },
    "StartTime": "2020-09-18T22:27:23.970000+00:00",
    "CreationTime": "2020-09-18T22:27:23.948000+00:00",
    "IdentifyLanguage": true
  }
}
```

For more information, see [Identifying the Language](#) in the *Amazon Transcribe Developer Guide*.

Example 8: To transcribe an audio file with personally identifiable information redacted

The following `start-transcription-job` example transcribes your audio file and redacts any personally identifiable information in the transcription output.

```
aws transcribe start-transcription-job \
  --cli-input-json file://myeighthfile.json
```

Contents of `myeighthfile.json`:

```
{
  "TranscriptionJobName": "cli-redaction-job",
  "LanguageCode": "language-code",
  "Media": {
```

```

    "MediaFileUri": "s3://Amazon-S3-Prefix/your-media-file.file-extension"
  },
  "ContentRedaction": {
    "RedactionOutput": "redacted",
    "RedactionType": "PII"
  }
}

```

Output:

```

{
  "TranscriptionJob": {
    "TranscriptionJobName": "cli-redaction-job",
    "TranscriptionJobStatus": "IN_PROGRESS",
    "LanguageCode": "language-code",
    "Media": {
      "MediaFileUri": "s3://Amazon-S3-Prefix/your-media-file.file-extension"
    },
    "StartTime": "2020-09-25T23:49:13.195000+00:00",
    "CreationTime": "2020-09-25T23:49:13.176000+00:00",
    "ContentRedaction": {
      "RedactionType": "PII",
      "RedactionOutput": "redacted"
    }
  }
}

```

For more information, see [Automatic Content Redaction](#) in the *Amazon Transcribe Developer Guide*.

Example 9: To generate a transcript with personally identifiable information (PII) redacted and an unredacted transcript

The following `start-transcription-job` example generates two transcriptions of your audio file, one with the personally identifiable information redacted, and the other without any redactions.

```

aws transcribe start-transcription-job \
  --cli-input-json file://myninthfile.json

```

Contents of `myninthfile.json`:

```
{
  "TranscriptionJobName": "cli-redaction-job-with-unredacted-transcript",
  "LanguageCode": "language-code",
  "Media": {
    "MediaFileUri": "s3://Amazon-S3-Prefix/your-media-file.file-extension"
  },
  "ContentRedaction": {
    "RedactionOutput": "redacted_and_unredacted",
    "RedactionType": "PII"
  }
}
```

Output:

```
{
  "TranscriptionJob": {
    "TranscriptionJobName": "cli-redaction-job-with-unredacted-transcript",
    "TranscriptionJobStatus": "IN_PROGRESS",
    "LanguageCode": "language-code",
    "Media": {
      "MediaFileUri": "s3://Amazon-S3-Prefix/your-media-file.file-extension"
    },
    "StartTime": "2020-09-25T23:59:47.677000+00:00",
    "CreationTime": "2020-09-25T23:59:47.653000+00:00",
    "ContentRedaction": {
      "RedactionType": "PII",
      "RedactionOutput": "redacted_and_unredacted"
    }
  }
}
```

For more information, see [Automatic Content Redaction](#) in the *Amazon Transcribe Developer Guide*.

Example 10: To use a custom language model you've previously created to transcribe an audio file.

The following `start-transcription-job` example transcribes your audio file with a custom language model you've previously created.

```
aws transcribe start-transcription-job \
```

```
--cli-input-json file://mytenthfile.json
```

Contents of mytenthfile.json:

```
{
  "TranscriptionJobName": "cli-clm-2-job-1",
  "LanguageCode": "language-code",
  "Media": {
    "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.file-extension"
  },
  "ModelSettings": {
    "LanguageModelName": "cli-clm-2"
  }
}
```

Output:

```
{
  "TranscriptionJob": {
    "TranscriptionJobName": "cli-clm-2-job-1",
    "TranscriptionJobStatus": "IN_PROGRESS",
    "LanguageCode": "language-code",
    "Media": {
      "MediaFileUri": "s3://DOC-EXAMPLE-BUCKET/your-audio-file.file-extension"
    },
    "StartTime": "2020-09-28T17:56:01.835000+00:00",
    "CreationTime": "2020-09-28T17:56:01.801000+00:00",
    "ModelSettings": {
      "LanguageModelName": "cli-clm-2"
    }
  }
}
```

For more information, see [Improving Domain-Specific Transcription Accuracy with Custom Language Models](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [StartTranscriptionJob](#) in *AWS CLI Command Reference*.

update-medical-vocabulary

The following code example shows how to use update-medical-vocabulary.

AWS CLI

To update a medical custom vocabulary with new terms.

The following `update-medical-vocabulary` example replaces the terms used in a medical custom vocabulary with the new ones. Prerequisite: to replace the terms in a medical custom vocabulary, you need a file with new terms.

```
aws transcribe update-medical-vocabulary \  
  --vocabulary-file-uri s3://DOC-EXAMPLE-BUCKET/Amazon-S3-Prefix/medical-custom-  
vocabulary.txt \  
  --vocabulary-name medical-custom-vocabulary \  
  --language-code language
```

Output:

```
{  
  "VocabularyName": "medical-custom-vocabulary",  
  "LanguageCode": "en-US",  
  "VocabularyState": "PENDING"  
}
```

For more information, see [Medical Custom Vocabularies](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [UpdateMedicalVocabulary](#) in *AWS CLI Command Reference*.

update-vocabulary-filter

The following code example shows how to use `update-vocabulary-filter`.

AWS CLI

To replace the words in a vocabulary filter

The following `update-vocabulary-filter` example replaces the words in a vocabulary filter with new ones. Prerequisite: To update a vocabulary filter with the new words, you must have those words saved as a text file.

```
aws transcribe update-vocabulary-filter \  
  --vocabulary-filter-name medical-custom-vocabulary-filter \  
  --words-file-uri s3://DOC-EXAMPLE-BUCKET/Amazon-S3-Prefix/medical-custom-  
vocabulary-filter-words.txt
```



```
--vocabulary-filter-file-uri s3://DOC-EXAMPLE-BUCKET/Amazon-S3-Prefix/your-text-  
file-to-update-your-vocabulary-filter.txt \  
--vocabulary-filter-name vocabulary-filter-name
```

Output:

```
{  
  "VocabularyFilterName": "vocabulary-filter-name",  
  "LanguageCode": "language-code",  
  "LastModifiedTime": "2020-09-23T18:40:35.139000+00:00"  
}
```

For more information, see [Filtering Unwanted Words](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [UpdateVocabularyFilter](#) in *AWS CLI Command Reference*.

update-vocabulary

The following code example shows how to use update-vocabulary.

AWS CLI

To update a custom vocabulary with new terms.

The following update-vocabulary example overwrites the terms used to create a custom vocabulary with the new ones that you provide. Prerequisite: to replace the terms in a custom vocabulary, you need a file with new terms.

```
aws transcribe update-vocabulary \  
  --vocabulary-file-uri s3://DOC-EXAMPLE-BUCKET/Amazon-S3-Prefix/custom-  
vocabulary.txt \  
  --vocabulary-name custom-vocabulary \  
  --language-code language-code
```

Output:

```
{  
  "VocabularyName": "custom-vocabulary",  
  "LanguageCode": "language",  
  "VocabularyState": "PENDING"
```

```
}
```

For more information, see [Custom Vocabularies](#) in the *Amazon Transcribe Developer Guide*.

- For API details, see [UpdateVocabulary](#) in *AWS CLI Command Reference*.

Amazon Translate examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon Translate.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

import-terminology

The following code example shows how to use `import-terminology`.

AWS CLI

To import a custom terminology from a file

The following `import-terminology` example creates a terminology called `MyTestTerminology` from the `test-terminology.csv` file:

```
aws translate import-terminology \  
  --name MyTestTerminology \  
  --file test-terminology.csv
```

```
--description "Creating a test terminology in AWS Translate" \  
--merge-strategy OVERWRITE \  
--data-file fileb://test-terminology.csv \  
--terminology-data Format=CSV
```

Contents of test-terminology.csv:

```
en,fr,es,zh Hello world!,Bonjour tout le monde!,Hola Mundo!,????  
Amazon,Amazon,Amazon,Amazon
```

Output:

```
{  
  "TerminologyProperties": {  
    "SourceLanguageCode": "en",  
    "Name": "MyTestTerminology",  
    "TargetLanguageCodes": [  
      "fr",  
      "es",  
      "zh"  
    ],  
    "SizeBytes": 97,  
    "LastUpdatedAt": 1571089500.851,  
    "CreatedAt": 1571089500.851,  
    "TermCount": 6,  
    "Arn": "arn:aws:translate:us-west-2:123456789012:terminology/  
MyTestTerminology/LATEST",  
    "Description": "Creating a test terminology in AWS Translate"  
  }  
}
```

- For API details, see [ImportTerminology](#) in *AWS CLI Command Reference*.

Trusted Advisor examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Trusted Advisor.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

get-organization-recommendation

The following code example shows how to use `get-organization-recommendation`.

AWS CLI

To get an organization recommendation

The following `get-organization-recommendation` example gets an organization recommendation by its identifier.

```
aws trustedadvisor get-organization-recommendation \  
  --organization-recommendation-identifier arn:aws:trustedadvisor::organization-  
  recommendation/9534ec9b-bf3a-44e8-8213-2ed68b39d9d5
```

Output:

```
{  
  "organizationRecommendation": {  
    "arn": "arn:aws:trustedadvisor::organization-recommendation/9534ec9b-  
    bf3a-44e8-8213-2ed68b39d9d5",  
    "name": "Lambda Runtime Deprecation Warning",  
    "description": "One or more lambdas are using a deprecated runtime",  
    "awsServices": [  
      "lambda"  
    ],  
    "checkArn": "arn:aws:trustedadvisor::check/L4dfs2Q4C5",  
    "id": "9534ec9b-bf3a-44e8-8213-2ed68b39d9d5",  
    "lifecycleStage": "resolved",
```

```

    "pillars": [
      "security"
    ],
    "resourcesAggregates": {
      "errorCount": 0,
      "okCount": 0,
      "warningCount": 0
    },
    "source": "ta_check",
    "status": "warning",
    "type": "priority"
  }
}

```

For more information, see [Get started with the Trusted Advisor API](#) in the *AWS Trusted Advisor User Guide*.

- For API details, see [GetOrganizationRecommendation](#) in *AWS CLI Command Reference*.

get-recommendation

The following code example shows how to use `get-recommendation`.

AWS CLI

To get a recommendation

The following `get-recommendation` example gets a recommendation by its identifier.

```

aws trustedadvisor get-recommendation \
  --recommendation-identifier
  arn:aws:trustedadvisor::000000000000:recommendation/55fa4d2e-
  bbb7-491a-833b-5773e9589578

```

Output:

```

{
  "recommendation": {
    "arn": "arn:aws:trustedadvisor::000000000000:recommendation/55fa4d2e-
    bbb7-491a-833b-5773e9589578",
    "name": "MFA Recommendation",
    "description": "Enable multi-factor authentication",

```

```
    "awsServices": [
      "iam"
    ],
    "checkArn": "arn:aws:trustedadvisor:::check/7DAFEemoDos",
    "id": "55fa4d2e-bbb7-491a-833b-5773e9589578",
    "lastUpdatedAt": "2023-11-01T15:57:58.673Z",
    "pillarSpecificAggregates": {
      "costOptimizing": {
        "estimatedMonthlySavings": 0.0,
        "estimatedPercentMonthlySavings": 0.0
      }
    },
    "pillars": [
      "security"
    ],
    "resourcesAggregates": {
      "errorCount": 1,
      "okCount": 0,
      "warningCount": 0
    },
    "source": "ta_check",
    "status": "error",
    "type": "standard"
  }
}
```

For more information, see [Get started with the Trusted Advisor API](#) in the *AWS Trusted Advisor User Guide*.

- For API details, see [GetRecommendation](#) in *AWS CLI Command Reference*.

list-checks

The following code example shows how to use `list-checks`.

AWS CLI

To list Trusted Advisor checks

The following `list-checks` example lists all Trusted Advisor checks.

```
aws trustedadvisor list-checks
```

Output:

```
{
  "checkSummaries": [
    {
      "arn": "arn:aws:trustedadvisor:::check/1iG5NDGVre",
      "awsServices": [
        "EC2"
      ],
      "description": "Checks security groups for rules that allow unrestricted
access to a resource. Unrestricted access increases opportunities for malicious
activity (hacking, denial-of-service attacks, loss of data)",
      "id": "1iG5NDGVre",
      "metadata": {
        "0": "Region",
        "1": "Security Group Name",
        "2": "Security Group ID",
        "3": "Protocol",
        "4": "Port",
        "5": "Status",
        "6": "IP Range"
      },
      "name": "Security Groups - Unrestricted Access",
      "pillars": [
        "security"
      ],
      "source": "ta_check"
    },
    {
      "arn": "arn:aws:trustedadvisor:::check/1qazXsw23e",
      "awsServices": [
        "RDS"
      ],
      "description": "Checks your usage of RDS and provides recommendations
on purchase of Reserved Instances to help reduce costs incurred from using RDS
On-Demand. AWS generates these recommendations by analyzing your On-Demand usage
for the past 30 days. We then simulate every combination of reservations in the
generated category of usage in order to identify the best number of each type
of Reserved Instance to purchase to maximize your savings. This check covers
recommendations based on partial upfront payment option with 1-year or 3-year
commitment. This check is not available to accounts linked in Consolidated Billing.
Recommendations are only available for the Paying Account.",
      "id": "1qazXsw23e",
      "metadata": {
```

```

        "0": "Region",
        "1": "Family",
        "2": "Instance Type",
        "3": "License Model",
        "4": "Database Edition",
        "5": "Database Engine",
        "6": "Deployment Option",
        "7": "Recommended number of Reserved Instances to purchase",
        "8": "Expected Average Reserved Instance Utilization",
        "9": "Estimated Savings with Recommendation (monthly)"
        "10": "Upfront Cost of Reserved Instances",
        "11": "Estimated cost of Reserved Instances (monthly)",
        "12": "Estimated On-Demand Cost Post Recommended Reserved Instance
Purchase (monthly)",
        "13": "Estimated Break Even (months)",
        "14": "Lookback Period (days)",
        "15": "Term (years)"
    },
    "name": "Amazon Relational Database Service (RDS) Reserved Instance
Optimization",
    "pillars": [
        "cost_optimizing"
    ],
    "source": "ta_check"
},
{
    "arn": "arn:aws:trustedadvisor:::check/1qw23er45t",
    "awsServices": [
        "Redshift"
    ],
    "description": "Checks your usage of Redshift and provides
recommendations on purchase of Reserved Nodes to help reduce costs incurred from
using Redshift On-Demand. AWS generates these recommendations by analyzing your
On-Demand usage for the past 30 days. We then simulate every combination of
reservations in the generated category of usage in order to identify the best
number of each type of Reserved Nodes to purchase to maximize your savings. This
check covers recommendations based on partial upfront payment option with 1-year or
3-year commitment. This check is not available to accounts linked in Consolidated
Billing. Recommendations are only available for the Paying Account.",
    "id": "1qw23er45t",
    "metadata": {
        "0": "Region",
        "1": "Family",
        "2": "Node Type",

```



```

        "3": "Recommended number of Reserved Nodes to purchase",
        "4": "Expected Average Reserved Node Utilization",
        "5": "Estimated Savings with Recommendation (monthly)",
        "6": "Upfront Cost of Reserved Nodes",
        "7": "Estimated cost of Reserved Nodes (monthly)",
        "8": "Estimated On-Demand Cost Post Recommended Reserved Nodes
Purchase (monthly)",
        "9": "Estimated Break Even (months)",
        "10": "Lookback Period (days)",
        "11": "Term (years)",
    },
    "name": "Amazon Redshift Reserved Node Optimization",
    "pillars": [
        "cost_optimizing"
    ],
    "source": "ta_check"
},
],
"nextToken": "REDACTED"
}

```

For more information, see [Get started with the Trusted Advisor API](#) in the *AWS Trusted Advisor User Guide*.

- For API details, see [ListChecks](#) in *AWS CLI Command Reference*.

list-organization-recommendation-accounts

The following code example shows how to use `list-organization-recommendation-accounts`.

AWS CLI

To list organization recommendation accounts

The following `list-organization-recommendation-accounts` example lists all account recommendation summaries for an organization recommendation by its identifier.

```

aws trustedadvisor list-organization-recommendation-accounts \
  --organization-recommendation-identifier arn:aws:trustedadvisor:::organization-
  recommendation/9534ec9b-bf3a-44e8-8213-2ed68b39d9d5

```

Output:

```
{
  "accountRecommendationLifecycleSummaries": [{
    "accountId": "000000000000",
    "accountRecommendationArn":
    "arn:aws:trustedadvisor::000000000000:recommendation/9534ec9b-
    bf3a-44e8-8213-2ed68b39d9d5",
    "lifecycleStage": "resolved",
    "updateReason": "Resolved issue",
    "updateReasonCode": "valid_business_case",
    "lastUpdatedAt": "2023-01-17T18:25:44.552Z"
  }],
  "nextToken": "REDACTED"
}
```

For more information, see [Get started with the Trusted Advisor API](#) in the *AWS Trusted Advisor User Guide*.

- For API details, see [ListOrganizationRecommendationAccounts](#) in *AWS CLI Command Reference*.

list-organization-recommendation-resources

The following code example shows how to use `list-organization-recommendation-resources`.

AWS CLI

To list organization recommendation resources

The following `list-organization-recommendation-resources` example lists all resources for an organization recommendation by its identifier.

```
aws trustedadvisor list-organization-recommendation-resources \
  --organization-recommendation-identifier arn:aws:trustedadvisor::organization-
  recommendation/5a694939-2e54-45a2-ae72-730598fa89d0
```

Output:

```
{
  "organizationRecommendationResourceSummaries": [
    {
```

```

    "arn": "arn:aws:trustedadvisor::000000000000:recommendation-
resource/5a694939-2e54-45a2-ae72-730598fa89d0/
bb38affc0ce0681d9a6cd13f30238ba03a8f63dfe7a379dc403c619119d86af",
    "awsResourceId": "database-1-instance-1",
    "id":
"bb38affc0ce0681d9a6cd13f302383ba03a8f63dfe7a379dc403c619119d86af",
    "lastUpdatedAt": "2023-11-01T15:09:51.891Z",
    "metadata": {
      "0": "14",
      "1": "208.79999999999998",
      "2": "database-1-instance-1",
      "3": "db.r5.large",
      "4": "false",
      "5": "us-west-2",
      "6": "arn:aws:rds:us-west-2:000000000000:db:database-1-instance-1",
      "7": "1"
    },
    "recommendationArn": "arn:aws:trustedadvisor::organization-
recommendation/5a694939-2e54-45a2-ae72-730598fa89d0",
    "regionCode": "us-west-2",
    "status": "warning"
  },
  {
    "arn": "arn:aws:trustedadvisor::000000000000:recommendation-
resource/5a694939-2e54-45a2-
ae72-730598fa89d0/51fded4d7a3278818df9cfe344ff5762cec46c095a6763d1ba1ba53bd0e1b0e6",
    "awsResourceId": "database-1",
    "id":
"51fded4d7a3278818df9cfe344ff5762cec46c095a6763d1ba1ba53bd0e1b0e6",
    "lastUpdatedAt": "2023-11-01T15:09:51.891Z",
    "metadata": {
      "0": "14",
      "1": "31.679999999999996",
      "2": "database-1",
      "3": "db.t3.small",
      "4": "false",
      "5": "us-west-2",
      "6": "arn:aws:rds:us-west-2:000000000000:db:database-1",
      "7": "20"
    },
    "recommendationArn": "arn:aws:trustedadvisor::organization-
recommendation/5a694939-2e54-45a2-ae72-730598fa89d0",
    "regionCode": "us-west-2",
    "status": "warning"
  }

```

```

    },
    {
      "arn": "arn:aws:trustedadvisor::000000000000:recommendation-
resource/5a694939-2e54-45a2-ae72-730598fa89d0/
f4d01bd20f4cd5372062aafc8786c489e48f0ead7cdab121463bf9f89e40a36b",
      "awsResourceId": "database-2-instance-1-us-west-2a",
      "id":
"f4d01bd20f4cd5372062aafc8786c489e48f0ead7cdab121463bf9f89e40a36b",
      "lastUpdatedAt": "2023-11-01T15:09:51.891Z",
      "metadata": {
        "0": "14",
        "1": "187.200000000000002",
        "2": "database-2-instance-1-us-west-2a",
        "3": "db.r6g.large",
        "4": "true",
        "5": "us-west-2",
        "6": "arn:aws:rds:us-west-2:000000000000:db:database-2-instance-1-
us-west-2a",
        "7": "1"
      },
      "recommendationArn": "arn:aws:trustedadvisor::organization-
recommendation/5a694939-2e54-45a2-ae72-730598fa89d0",
      "regionCode": "us-west-2",
      "status": "warning"
    },
  ],
  "nextToken": "REDACTED"
}

```

For more information, see [Get started with the Trusted Advisor API](#) in the *AWS Trusted Advisor User Guide*.

- For API details, see [ListOrganizationRecommendationResources](#) in *AWS CLI Command Reference*.

list-organization-recommendations

The following code example shows how to use `list-organization-recommendations`.

AWS CLI

Example 1: To list organization recommendations

The following `list-organization-recommendations` example lists all organization recommendations and does not include a filter.

```
aws trustedadvisor list-organization-recommendations
```

Output:

```
{
  "organizationRecommendationSummaries": [
    {
      "arn": "arn:aws:trustedadvisor::organization-recommendation/9534ec9b-
bf3a-44e8-8213-2ed68b39d9d5",
      "name": "Lambda Runtime Deprecation Warning",
      "awsServices": [
        "lambda"
      ],
      "checkArn": "arn:aws:trustedadvisor::check/L4dfs2Q4C5",
      "id": "9534ec9b-bf3a-44e8-8213-2ed68b39d9d5",
      "lifecycleStage": "resolved",
      "pillars": [
        "security"
      ],
      "resourcesAggregates": {
        "errorCount": 0,
        "okCount": 0,
        "warningCount": 0
      },
      "source": "ta_check",
      "status": "warning",
      "type": "priority"
    },
    {
      "arn": "arn:aws:trustedadvisor::organization-
recommendation/4ecff4d4-1bc1-4c99-a5b8-0fff9ee500d6",
      "name": "Lambda Runtime Deprecation Warning",
      "awsServices": [
        "lambda"
      ],
      "checkArn": "arn:aws:trustedadvisor::check/L4dfs2Q4C5",
      "id": "4ecff4d4-1bc1-4c99-a5b8-0fff9ee500d6",
      "lifecycleStage": "resolved",
      "pillars": [
        "security"
      ]
    }
  ]
}
```

```

    ],
    "resourcesAggregates": {
      "errorCount": 0,
      "okCount": 0,
      "warningCount": 0
    },
    "source": "ta_check",
    "status": "warning",
    "type": "priority"
  },
],
"nextToken": "REDACTED"
}

```

For more information, see [Get started with the Trusted Advisor API](#) in the *AWS Trusted Advisor User Guide*.

Example 2: To list organization recommendations with a filter

The following `list-organization-recommendations` example filters and returns a max of one organization recommendation that is a part of the "security" pillar.

```

aws trustedadvisor list-organization-recommendations \
  --pillar security \
  --max-items 100

```

Output:

```

{
  "organizationRecommendationSummaries": [{
    "arn": "arn:aws:trustedadvisor::organization-recommendation/9534ec9b-
bf3a-44e8-8213-2ed68b39d9d5",
    "name": "Lambda Runtime Deprecation Warning",
    "awsServices": [
      "lambda"
    ],
    "checkArn": "arn:aws:trustedadvisor::check/L4dfs2Q4C5",
    "id": "9534ec9b-bf3a-44e8-8213-2ed68b39d9d5",
    "lifecycleStage": "resolved",
    "pillars": [
      "security"
    ],
    "resourcesAggregates": {

```

```

        "errorCount": 0,
        "okCount": 0,
        "warningCount": 0
    },
    "source": "ta_check",
    "status": "warning",
    "type": "priority"
}],
"nextToken": "REDACTED"
}

```

For more information, see [Get started with the Trusted Advisor API](#) in the *AWS Trusted Advisor User Guide*.

Example 3: To list organization recommendations with a pagination token

The following `list-organization-recommendations` example uses the `"nextToken"` returned from a previous request to fetch the next page of organization recommendations.

```

aws trustedadvisor list-organization-recommendations \
  --pillar security \
  --max-items 100 \
  --starting-token <next-token>

```

Output:

```

{
  "organizationRecommendationSummaries": [{
    "arn": "arn:aws:trustedadvisor::organization-recommendation/4ecff4d4-1bc1-4c99-a5b8-0fff9ee500d6",
    "name": "Lambda Runtime Deprecation Warning",
    "awsServices": [
      "lambda"
    ],
    "checkArn": "arn:aws:trustedadvisor::check/L4dfs2Q4C5",
    "id": "4ecff4d4-1bc1-4c99-a5b8-0fff9ee500d6",
    "lifecycleStage": "resolved",
    "pillars": [
      "security"
    ],
    "resourcesAggregates": {
      "errorCount": 0,
      "okCount": 0,

```

```

        "warningCount": 0
      },
      "source": "ta_check",
      "status": "warning",
      "type": "priority"
    }]
  }

```

For more information, see [Get started with the Trusted Advisor API](#) in the *AWS Trusted Advisor User Guide*.

- For API details, see [ListOrganizationRecommendations](#) in *AWS CLI Command Reference*.

list-recommendation-resources

The following code example shows how to use `list-recommendation-resources`.

AWS CLI

To list recommendation resources

The following `list-recommendation-resources` example lists all resources for a recommendation by its identifier.

```

aws trustedadvisor list-recommendation-resources \
  --recommendation-identifier
  arn:aws:trustedadvisor::000000000000:recommendation/55fa4d2e-
  bbb7-491a-833b-5773e9589578

```

Output:

```

{
  "recommendationResourceSummaries": [
    {
      "arn": "arn:aws:trustedadvisor::000000000000:recommendation-
resource/55fa4d2e-
bbb7-491a-833b-5773e9589578/18959a1f1973cff8e706e9d9bde28bba36cd602a6b2cb86c8b61252835236010",
      "id":
"18959a1f1973cff8e706e9d9bde28bba36cd602a6b2cb86c8b61252835236010",
      "awsResourceId": "webcms-dev-01",
      "lastUpdatedAt": "2023-11-01T15:09:51.891Z",
      "metadata": {
        "0": "14",

```



```

        "1": "123.120000000000002",
        "2": "webcms-dev-01",
        "3": "db.m6i.large",
        "4": "false",
        "5": "us-east-1",
        "6": "arn:aws:rds:us-east-1:000000000000:db:webcms-dev-01",
        "7": "20"
    },
    "recommendationArn":
"arn:aws:trustedadvisor::000000000000:recommendation/55fa4d2e-
bbb7-491a-833b-5773e9589578",
    "regionCode": "us-east-1",
    "status": "warning"
},
{
    "arn": "arn:aws:trustedadvisor::000000000000:recommendation-
resource/55fa4d2e-bbb7-491a-833b-5773e9589578/
e6367ff500ac90db8e4adeb4892e39ee9c36bbf812dcbce4b9e4fefcec9eb63e",
    "id":
"e6367ff500ac90db8e4adeb4892e39ee9c36bbf812dcbce4b9e4fefcec9eb63e",
    "awsResourceId": "aws-dev-db-stack-instance-1",
    "lastUpdatedAt": "2023-11-01T15:09:51.891Z",
    "metadata": {
        "0": "14",
        "1": "29.52",
        "2": "aws-dev-db-stack-instance-1",
        "3": "db.t2.small",
        "4": "false",
        "5": "us-east-1",
        "6": "arn:aws:rds:us-east-1:000000000000:db:aws-dev-db-stack-
instance-1",
        "7": "1"
    },
    "recommendationArn":
"arn:aws:trustedadvisor::000000000000:recommendation/55fa4d2e-
bbb7-491a-833b-5773e9589578",
    "regionCode": "us-east-1",
    "status": "warning"
},
{
    "arn": "arn:aws:trustedadvisor::000000000000:recommendation-
resource/55fa4d2e-
bbb7-491a-833b-5773e9589578/31aa78ba050a5015d2d38cca7f5f1ce88f70857c4e1c3ad03f8f9fd95dad7459

```

```

        "id":
        "31aa78ba050a5015d2d38cca7f5f1ce88f70857c4e1c3ad03f8f9fd95dad7459",
        "awsResourceId": "aws-awesome-apps-stack-db",
        "lastUpdatedAt": "2023-11-01T15:09:51.891Z",
        "metadata": {
            "0": "14",
            "1": "114.48000000000002",
            "2": "aws-awesome-apps-stack-db",
            "3": "db.m6g.large",
            "4": "false",
            "5": "us-east-1",
            "6": "arn:aws:rds:us-east-1:000000000000:db:aws-awesome-apps-stack-
db",
            "7": "100"
        },
        "recommendationArn":
        "arn:aws:trustedadvisor::000000000000:recommendation/55fa4d2e-
bbb7-491a-833b-5773e9589578",
        "regionCode": "us-east-1",
        "status": "warning"
    }
],
    "nextToken": "REDACTED"
}

```

For more information, see [Get started with the Trusted Advisor API](#) in the *AWS Trusted Advisor User Guide*.

- For API details, see [ListRecommendationResources](#) in *AWS CLI Command Reference*.

list-recommendations

The following code example shows how to use `list-recommendations`.

AWS CLI

Example 1: To list recommendations

The following `list-recommendations` example lists all recommendations and does not include a filter.

```
aws trustedadvisor list-recommendations
```

Output:

```
{
  "recommendationSummaries": [
    {
      "arn": "arn:aws:trustedadvisor::000000000000:recommendation/55fa4d2e-
bbb7-491a-833b-5773e9589578",
      "name": "MFA Recommendation",
      "awsServices": [
        "iam"
      ],
      "checkArn": "arn:aws:trustedadvisor:::check/7DAFEmoDos",
      "id": "55fa4d2e-bbb7-491a-833b-5773e9589578",
      "lastUpdatedAt": "2023-11-01T15:57:58.673Z",
      "pillarSpecificAggregates": {
        "costOptimizing": {
          "estimatedMonthlySavings": 0.0,
          "estimatedPercentMonthlySavings": 0.0
        }
      },
      "pillars": [
        "security"
      ],
      "resourcesAggregates": {
        "errorCount": 1,
        "okCount": 0,
        "warningCount": 0
      },
      "source": "ta_check",
      "status": "error",
      "type": "standard"
    },
    {
      "arn":
"arn:aws:trustedadvisor::000000000000:recommendation/8b602b6f-452d-4cb2-8a9e-
c7650955d9cd",
      "name": "RDS clusters quota warning",
      "awsServices": [
        "rds"
      ],
      "checkArn": "arn:aws:trustedadvisor:::check/gjqMBn6pjz",
      "id": "8b602b6f-452d-4cb2-8a9e-c7650955d9cd",
      "lastUpdatedAt": "2023-11-01T15:58:17.397Z",
      "pillarSpecificAggregates": {
```

```

        "costOptimizing": {
            "estimatedMonthlySavings": 0.0,
            "estimatedPercentMonthlySavings": 0.0
        },
    },
    "pillars": [
        "service_limits"
    ],
    "resourcesAggregates": {
        "errorCount": 0,
        "okCount": 3,
        "warningCount": 6
    },
    "source": "ta_check",
    "status": "warning",
    "type": "standard"
}
],
"nextToken": "REDACTED"
}

```

For more information, see [Get started with the Trusted Advisor API](#) in the *AWS Trusted Advisor User Guide*.

Example 2: To list recommendations with a filter

The following `list-recommendations` example lists recommendations and includes a filter.

```

aws trustedadvisor list-recommendations \
  --aws-service iam \
  --max-items 100

```

Output:

```

{
  "recommendationSummaries": [{
    "arn": "arn:aws:trustedadvisor::000000000000:recommendation/55fa4d2e-
bbb7-491a-833b-5773e9589578",
    "name": "MFA Recommendation",
    "awsServices": [
      "iam"
    ],
    "checkArn": "arn:aws:trustedadvisor:::check/7DAFEmoDos",

```

```

    "id": "55fa4d2e-bbb7-491a-833b-5773e9589578",
    "lastUpdatedAt": "2023-11-01T15:57:58.673Z",
    "pillarSpecificAggregates": {
      "costOptimizing": {
        "estimatedMonthlySavings": 0.0,
        "estimatedPercentMonthlySavings": 0.0
      }
    },
    "pillars": [
      "security"
    ],
    "resourcesAggregates": {
      "errorCount": 1,
      "okCount": 0,
      "warningCount": 0
    },
    "source": "ta_check",
    "status": "error",
    "type": "standard"
  }],
  "nextToken": "REDACTED"
}

```

For more information, see [Get started with the Trusted Advisor API](#) in the *AWS Trusted Advisor User Guide*.

Example 3: To list recommendations with a pagination token

The following `list-recommendations` example uses the `nextToken` returned from a previous request to fetch the next page of filtered Recommendations.

```

aws trustedadvisor list-recommendations \
  --aws-service rds \
  --max-items 100 \
  --starting-token <next-token>

```

Output:

```

{
  "recommendationSummaries": [{
    "arn":
    "arn:aws:trustedadvisor::000000000000:recommendation/8b602b6f-452d-4cb2-8a9e-
    c7650955d9cd",

```

```

    "name": "RDS clusters quota warning",
    "awsServices": [
      "rds"
    ],
    "checkArn": "arn:aws:trustedadvisor:::check/gjqMBn6pjz",
    "id": "8b602b6f-452d-4cb2-8a9e-c7650955d9cd",
    "lastUpdatedAt": "2023-11-01T15:58:17.397Z",
    "pillarSpecificAggregates": {
      "costOptimizing": {
        "estimatedMonthlySavings": 0.0,
        "estimatedPercentMonthlySavings": 0.0
      }
    },
    "pillars": [
      "service_limits"
    ],
    "resourcesAggregates": {
      "errorCount": 0,
      "okCount": 3,
      "warningCount": 6
    },
    "source": "ta_check",
    "status": "warning",
    "type": "standard"
  }
}

```

For more information, see [Get started with the Trusted Advisor API](#) in the *AWS Trusted Advisor User Guide*.

- For API details, see [ListRecommendations](#) in *AWS CLI Command Reference*.

update-organization-recommendation-lifecycle

The following code example shows how to use `update-organization-recommendation-lifecycle`.

AWS CLI

To update an organization recommendation lifecycle

The following `update-organization-recommendation-lifecycle` example updates the lifecycle of an organization recommendation by its identifier.

```
aws trustedadvisor update-organization-recommendation-lifecycle \  
  --organization-recommendation-identifier arn:aws:trustedadvisor:::organization-  
recommendation/96b5e5ca-7930-444c-90c6-06d386128100 \  
  --lifecycle-stage dismissed \  
  --update-reason-code not_applicable
```

This command produces no output.

For more information, see [Get started with the Trusted Advisor API](#) in the *AWS Trusted Advisor User Guide*.

- For API details, see [UpdateOrganizationRecommendationLifecycle](#) in *AWS CLI Command Reference*.

update-recommendation-lifecycle

The following code example shows how to use `update-recommendation-lifecycle`.

AWS CLI

To update a recommendation lifecycle

The following `update-recommendation-lifecycle` example updates the lifecycle of a recommendation by its identifier.

```
aws trustedadvisor update-recommendation-lifecycle \  
  --recommendation-identifier  
arn:aws:trustedadvisor:::000000000000:recommendation/861c9c6e-  
f169-405a-8b59-537a8cacc7a \  
  --lifecycle-stage resolved \  
  --update-reason-code valid_business_case
```

This command produces no output.

For more information, see [Get started with the Trusted Advisor API](#) in the *AWS Trusted Advisor User Guide*.

- For API details, see [UpdateRecommendationLifecycle](#) in *AWS CLI Command Reference*.

Verified Permissions examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Verified Permissions.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-identity-source

The following code example shows how to use `create-identity-source`.

AWS CLI

To create an identity source

The following `create-identity-source` example creates an identity source that lets you reference identities stored in the specified Amazon Cognito user pool. Those identities are available in Verified Permissions as entities of type `User`.

```
aws verifiedpermissions create-identity-source \  
  --configuration file://config.txt \  
  --principal-entity-type "User" \  
  --policy-store-id PSEXAMPLEEabcdefg111111
```

Contents of `config.txt`:

```
{
```



```
"cognitoUserPoolConfiguration": {
  "userPoolArn": "arn:aws:cognito-idp:us-west-2:123456789012:userpool/us-
west-2_1a2b3c4d5",
  "clientIds":["a1b2c3d4e5f6g7h8i9j0kalbmc"]
}
}
```

Output:

```
{
  "createdDate": "2023-05-19T20:30:28.214829+00:00",
  "identitySourceId": "ISEXAMPLEabcdefg111111",
  "lastUpdatedDate": "2023-05-19T20:30:28.214829+00:00",
  "policyStoreId": "PSEXAMPLEabcdefg111111"
}
```

For more information about identity sources, see [Using Amazon Verified Permissions with identity providers](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [CreateIdentitySource](#) in *AWS CLI Command Reference*.

create-policy-store

The following code example shows how to use create-policy-store.

AWS CLI

To create a policy store

The following create-policy-store example creates a policy store in the current AWS Region.

```
aws verifiedpermissions create-policy-store \
  --validation-settings "mode=STRICT"
```

Output:

```
{
  "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111",
  "createdDate": "2023-05-16T17:41:29.103459+00:00",
```

```
"lastUpdatedDate": "2023-05-16T17:41:29.103459+00:00",  
"policyStoreId": "PSEXAMPLEEabcdefg111111"  
}
```

For more information about policy stores, see [Amazon Verified Permissions policy stores](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [CreatePolicyStore](#) in *AWS CLI Command Reference*.

create-policy-template

The following code example shows how to use `create-policy-template`.

AWS CLI

Example 1: To create a policy template

The following `create-policy-template` example creates a policy template with a statement that contains a placeholder for the principal.

```
aws verifiedpermissions create-policy-template \  
  --definition file://template1.txt \  
  --policy-store-id PSEXAMPLEEabcdefg111111
```

Contents of file `template1.txt`:

```
permit(  
  principal in ?principal,  
  action == Action::"view",  
  resource == Photo::"VacationPhoto94.jpg"  
);
```

Output:

```
{  
  "createdDate": "2023-06-12T20:47:42.804511+00:00",  
  "lastUpdatedDate": "2023-06-12T20:47:42.804511+00:00",  
  "policyStoreId": "PSEXAMPLEEabcdefg111111",  
  "policyTemplateId": "PTEXAMPLEEabcdefg111111"  
}
```

For more information about policy templates, see [Amazon Verified Permissions policy templates](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [CreatePolicyTemplate](#) in *AWS CLI Command Reference*.

create-policy

The following code example shows how to use create-policy.

AWS CLI

Example 1: To create a static policy

The following create-policy example creates a static policy with a policy scope that specifies both a principal and a resource.

```
aws verifiedpermissions create-policy \  
  --definition file://definition1.txt \  
  --policy-store-id PSEXAMPLEEabcdefg111111
```

Contents of file definition1.txt:

```
{  
  "static": {  
    "description": "Grant everyone of janeFriends UserGroup access to the  
vacationFolder Album",  
    "statement": "permit(principal in UserGroup::\\"janeFriends\\", action,  
resource in Album::\\"vacationFolder\\" );"  
  }  
}
```

Output:

```
{  
  "createdDate": "2023-06-12T20:33:37.382907+00:00",  
  "lastUpdatedDate": "2023-06-12T20:33:37.382907+00:00",  
  "policyId": "SPEXAMPLEEabcdefg111111",  
  "policyStoreId": "PSEXAMPLEEabcdefg111111",  
  "policyType": "STATIC",  
  "principal": {  
    "entityId": "janeFriends",  
    "entityType": "UserGroup"  }  
}
```

```
    },
    "resource": {
      "entityId": "vacationFolder",
      "entityType": "Album"
    }
  }
}
```

Example 2: To create a static policy that grants access to a resource to everyone

The following `create-policy` example creates a static policy with a policy scope that specifies only a resource.

```
aws verifiedpermissions create-policy \
  --definition file://definition2.txt \
  --policy-store-id PSEXAMPLEEabcdefg111111
```

Contents of file `definition2.txt`:

```
{
  "static": {
    "description": "Grant everyone access to the publicFolder Album",
    "statement": "permit(principal, action, resource in Album:\""publicFolder
  \");"
  }
}
```

Output:

```
{
  "createdDate": "2023-06-12T20:39:44.975897+00:00",
  "lastUpdatedDate": "2023-06-12T20:39:44.975897+00:00",
  "policyId": "PbfR73F8oh5MMfr9uRtFDB",
  "policyStoreId": "PSEXAMPLEEabcdefg222222",
  "policyType": "STATIC",
  "resource": {
    "entityId": "publicFolder",
    "entityType": "Album"
  }
}
```

Example 3: To create a template-linked policy that is associated with the specified template

The following `create-policy` example creates a template-linked policy using the specified policy template and associates the specified principal to use with the new template-linked policy.

```
aws verifiedpermissions create-policy \  
  --definition file://definition.txt \  
  --policy-store-id PSEXAMPLEabcdefg111111
```

Contents of `definition.txt`:

```
{  
  "templateLinked": {  
    "policyTemplateId": "PTEXAMPLEabcdefg111111",  
    "principal": {  
      "entityType": "User",  
      "entityId": "alice"  
    }  
  }  
}
```

Output:

```
{  
  "createdDate": "2023-06-12T20:49:51.490211+00:00",  
  "lastUpdatedDate": "2023-06-12T20:49:51.490211+00:00",  
  "policyId": "TPEXAMPLEabcdefg111111",  
  "policyStoreId": "PSEXAMPLEabcdefg111111",  
  "policyType": "TEMPLATE_LINKED",  
  "principal": {  
    "entityId": "alice",  
    "entityType": "User"  
  },  
  "resource": {  
    "entityId": "VacationPhoto94.jpg",  
    "entityType": "Photo"  
  }  
}
```

For more information about policies, see [Amazon Verified Permissions policies](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [CreatePolicy](#) in *AWS CLI Command Reference*.

delete-identity-source

The following code example shows how to use delete-identity-source.

AWS CLI

To delete an identity source

The following delete-identity-source example deletes the identity source that has the specified Id.

```
aws verifiedpermissions delete-identity-source \  
  --identity-source-id ISEXAMPLEabcdefg111111 \  
  --policy-store-id PSEXAMPLEabcdefg111111
```

This command produces no output.

For more information about identity sources, see [Using Amazon Verified Permissions with identity providers](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [DeleteIdentitySource](#) in *AWS CLI Command Reference*.

delete-policy-store

The following code example shows how to use delete-policy-store.

AWS CLI

To delete a policy store

The following delete-policy-store example deletes the policy store that has the specified Id.

```
aws verifiedpermissions delete-policy-store \  
  --policy-store-id PSEXAMPLEabcdefg111111
```

This command produces no output.

For more information about policy stores, see [Amazon Verified Permissions policy stores](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [DeletePolicyStore](#) in *AWS CLI Command Reference*.

delete-policy-template

The following code example shows how to use delete-policy-template.

AWS CLI

To delete a policy template

The following delete-policy-template example deletes the policy template that has the specified Id.

```
aws verifiedpermissions delete-policy \  
  --policy-template-id PTEXTAMPEabcdefg111111 \  
  --policy-store-id PSEXAMPEabcdefg111111
```

This command produces no output.

For more information about policy templates, see [Amazon Verified Permissions policy templates](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [DeletePolicyTemplate](#) in *AWS CLI Command Reference*.

delete-policy

The following code example shows how to use delete-policy.

AWS CLI

To delete a static or template-linked policy

The following delete-policy example deletes the policy that has the specified Id.

```
aws verifiedpermissions delete-policy \  
  --policy-id SPEXAMPEabcdefg111111 \  
  --policy-store-id PSEXAMPEabcdefg111111
```

This command produces no output.

For more information about policies, see [Amazon Verified Permissions policies](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [DeletePolicy](#) in *AWS CLI Command Reference*.

get-identity-source

The following code example shows how to use `get-identity-source`.

AWS CLI

To retrieve details about an identity source

The following `get-identity-source` example displays the details for the identity source with the specified `Id`.

```
aws verifiedpermissions get-identity-source \  
  --identity-source ISEXAMPLEabcdefg111111 \  
  --policy-store-id PSEXAMPLEabcdefg111111
```

Output:

```
{  
  "createdDate": "2023-06-12T22:27:49.150035+00:00",  
  "details": {  
    "clientIds": [ "a1b2c3d4e5f6g7h8i9j0kalbmc" ],  
    "discoveryUrl": "https://cognito-idp.us-west-2.amazonaws.com/us-  
west-2_1a2b3c4d5",  
    "openIdIssuer": "COGNITO",  
    "userPoolArn": "arn:aws:cognito-idp:us-west-2:123456789012:userpool/us-  
west-2_1a2b3c4d5"  
  },  
  "identitySourceId": "ISEXAMPLEabcdefg111111",  
  "lastUpdatedDate": "2023-06-12T22:27:49.150035+00:00",  
  "policyStoreId": "PSEXAMPLEabcdefg111111",  
  "principalEntityType": "User"  
}
```

For more information about identity sources, see [Using Amazon Verified Permissions with identity providers](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [GetIdentitySource](#) in *AWS CLI Command Reference*.

get-policy-store

The following code example shows how to use `get-policy-store`.

AWS CLI

To retrieve details about a policy store

The following `get-policy-store` example displays the details for the policy store with the specified Id.

```
aws verifiedpermissions get-policy-store \  
  --policy-store-id PSEXAMPLEEabcdefg111111
```

Output:

```
{  
  "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/  
PSEXAMPLEEabcdefg111111",  
  "createdDate": "2023-06-05T20:16:46.225598+00:00",  
  "lastUpdatedDate": "2023-06-08T20:40:23.173691+00:00",  
  "policyStoreId": "PSEXAMPLEEabcdefg111111",  
  "validationSettings": { "mode": "OFF" }  
}
```

For more information about policy stores, see [Amazon Verified Permissions policy stores](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [GetPolicyStore](#) in *AWS CLI Command Reference*.

get-policy-template

The following code example shows how to use `get-policy-template`.

AWS CLI

To retrieve details about a policy template

The following `get-policy-template` example displays the details for the policy template with the specified ID.

```
aws verifiedpermissions get-policy-template \  
  --policy-template-id PTEXAMPLEEabcdefg111111 \  
  --policy-store-id PSEXAMPLEEabcdefg111111
```

Output:

```
{
  "createdDate": "2023-06-12T20:47:42.804511+00:00",
  "lastUpdatedDate": "2023-06-12T20:47:42.804511+00:00",
  "policyStoreId": "PSEXAMPLEabcdefg111111",
  "policyTemplateId": "PTEXAMPLEabcdefg111111",
  "statement": "permit(\n  principal in ?principal,\n  action == Action::\n  \"view\", \n  resource == Photo::\"VacationPhoto94.jpg\"\\n);"
}
```

For more information about policy templates, see [Amazon Verified Permissions policy templates](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [GetPolicyTemplate](#) in *AWS CLI Command Reference*.

get-policy

The following code example shows how to use `get-policy`.

AWS CLI**To retrieve details about a policy**

The following `get-policy` example displays the details for the policy with the specified ID.

```
aws verifiedpermissions get-policy \
  --policy-id PSEXAMPLEabcdefg111111 \
  --policy-store-id PSEXAMPLEabcdefg111111
```

Output:

```
{
  "createdDate": "2023-06-12T20:33:37.382907+00:00",
  "definition": {
    "static": {
      "description": "Grant everyone of janeFriends UserGroup access to the vacationFolder Album",
      "statement": "permit(principal in UserGroup::\"janeFriends\", action, resource in Album::\"vacationFolder\" );"
    }
  },
}
```

```

    "lastUpdatedDate": "2023-06-12T20:33:37.382907+00:00",
    "policyId": "SPEXAMPLEabcdefg111111",
    "policyStoreId": "PSEXAMPLEabcdefg111111",
    "policyType": "STATIC",
    "principal": {
      "entityId": "janeFriends",
      "entityType": "UserGroup"
    },
    "resource": {
      "entityId": "vacationFolder",
      "entityType": "Album"
    }
  }
}

```

For more information about policies, see [Amazon Verified Permissions policies](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [GetPolicy](#) in *AWS CLI Command Reference*.

get-schema

The following code example shows how to use `get-schema`.

AWS CLI

To retrieve the schema in a policy store

The following `get-schema` example displays the details of the schema in the specified policy store.

```

aws verifiedpermissions get-schema \
  --policy-store-id PSEXAMPLEabcdefg111111

```

Output:

```

{
  "policyStoreId": "PSEXAMPLEabcdefg111111",
  "schema": "{\"MySampleNamespace\":{\"entityTypes\":{\"Employee\":{\"shape\":"
  "\":{\"attributes\":{\"jobLevel\":{\"type\":\"Long\"},\"name\":{\"type\":\"String\":"
  "\":\"Record\"}}},\"actions\":{\"remoteAccess\":{\"appliesTo\":{\"principalTypes\":"
  "\":[\"Employee\"]}}}}}}",
  "createdDate": "2023-06-14T17:47:13.999885+00:00",

```

```
"lastUpdatedDate": "2023-06-14T17:47:13.999885+00:00"
}
```

For more information about schema, see [Policy store schema](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [GetSchema](#) in *AWS CLI Command Reference*.

is-authorized-with-token

The following code example shows how to use `is-authorized-with-token`.

AWS CLI

Example 1: To request an authorization decision for a user request (allow)

The following `is-authorized-with-token` example requests an authorization decision for a user who was authenticated by Amazon Cognito. The request uses the identity token provided by Cognito rather than the access token. In this example, the specified information store is configured to return principals as entities of type `CognitoUser`.

```
aws verifiedpermissions is-authorized-with-token \
  --action actionId="View",actionType="Action" \
  --resource entityId="vacationPhoto94.jpg",entityType="Photo" \
  --policy-store-id PSEXAMPLEEabcdefg111111 \
  --identity-token "AbCdE12345...long.string...54321EdCbA"
```

The policy store contains a policy with the following statement that accepts identities from the specified Cognito user pool and application Id.

```
permit(
  principal == CognitoUser::"us-east-1_1a2b3c4d5|a1b2c3d4e5f6g7h8i9j0ka1bmc",
  action,
  resource == Photo::"VacationPhoto94.jpg"
);
```

Output:

```
{
  "decision":"Allow",
```

```
"determiningPolicies":[
  {
    "determiningPolicyId":"SPEXAMPLEabcdefg111111"
  }
],
"errors":[]
}
```

For more information about using identities from a Cognito user pool, see [Using Amazon Verified Permissions with identity providers](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [IsAuthorizedWithToken](#) in *AWS CLI Command Reference*.

is-authorized

The following code example shows how to use `is-authorized`.

AWS CLI

Example 1: To request an authorization decision for a user request (allow)

The following `is-authorized` example requests an authorization decision for a principal of type `User` named `Alice`, who wants to perform the `updatePhoto` operation, on a resource of type `Photo` named `VacationPhoto94.jpg`.

The response shows that the request is allowed by one policy.

```
aws verifiedpermissions is-authorized \
  --principal entityType=User,entityId=alice \
  --action actionType=Action,actionId=view \
  --resource entityType=Photo,entityId=VacationPhoto94.jpg \
  --policy-store-id SPEXAMPLEabcdefg111111
```

Output:

```
{
  "decision": "ALLOW",
  "determiningPolicies": [
    {
      "policyId": "SPEXAMPLEabcdefg111111"
    }
  ],
}
```

```
"errors": []
}
```

Example 2: To request an authorization decision for a user request (deny)

The following example is the same as the previous example, except that the principal is `User::"Bob"`. The policy store doesn't contain any policy that allows that user access to `Album::"alice_folder"`.

The output indicates that the Deny was implicit because the list of `DeterminingPolicies` is empty.

```
aws verifiedpermissions create-policy \
  --definition file://definition2.txt \
  --policy-store-id PSEXAMPLEEabcdefg111111
```

Output:

```
{
  "decision": "DENY",
  "determiningPolicies": [],
  "errors": []
}
```

For more information, see the [Amazon Verified Permissions User Guide](#).

- For API details, see [IsAuthorized](#) in *AWS CLI Command Reference*.

list-identity-sources

The following code example shows how to use `list-identity-sources`.

AWS CLI

To list the available identity sources

The following `list-identity-sources` example lists all identity sources in the specified policy store.

```
aws verifiedpermissions list-identity-sources \
```

```
--policy-store-id PSEXAMPLEEabcdefg111111
```

Output:

```
{
  "identitySources": [
    {
      "createdDate": "2023-06-12T22:27:49.150035+00:00",
      "details": {
        "clientIds": [ "a1b2c3d4e5f6g7h8i9j0kalbmc" ],
        "discoveryUrl": "https://cognito-idp.us-west-2.amazonaws.com/us-west-2_1a2b3c4d5",
        "openIdIssuer": "COGNITO",
        "userPoolArn": "arn:aws:cognito-idp:us-west-2:123456789012:userpool/us-west-2_1a2b3c4d5"
      },
      "identitySourceId": "ISEXAMPLEEabcdefg111111",
      "lastUpdatedDate": "2023-06-12T22:27:49.150035+00:00",
      "policyStoreId": "PSEXAMPLEEabcdefg111111",
      "principalEntityType": "User"
    }
  ]
}
```

For more information about identity sources, see [Using Amazon Verified Permissions with identity providers](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [ListIdentitySources](#) in *AWS CLI Command Reference*.

list-policies

The following code example shows how to use `list-policies`.

AWS CLI

To list the available policies

The following `list-policies` example lists all policies in the specified policy store.

```
aws verifiedpermissions list-policies \  
  --policy-store-id PSEXAMPLEEabcdefg111111
```

Output:

```
{
  "policies": [
    {
      "createdDate": "2023-06-12T20:33:37.382907+00:00",
      "definition": {
        "static": {
          "description": "Grant everyone of janeFriends UserGroup access
to the vacationFolder Album"
        }
      },
      "lastUpdatedDate": "2023-06-12T20:33:37.382907+00:00",
      "policyId": "SPEXAMPLEabcdefg111111",
      "policyStoreId": "PSEXAMPLEabcdefg111111",
      "policyType": "STATIC",
      "principal": {
        "entityId": "janeFriends",
        "entityType": "UserGroup"
      },
      "resource": {
        "entityId": "vacationFolder",
        "entityType": "Album"
      }
    },
    {
      "createdDate": "2023-06-12T20:39:44.975897+00:00",
      "definition": {
        "static": {
          "description": "Grant everyone access to the publicFolder Album"
        }
      },
      "lastUpdatedDate": "2023-06-12T20:39:44.975897+00:00",
      "policyId": "SPEXAMPLEabcdefg222222",
      "policyStoreId": "PSEXAMPLEabcdefg111111",
      "policyType": "STATIC",
      "resource": {
        "entityId": "publicFolder",
        "entityType": "Album"
      }
    },
    {
      "createdDate": "2023-06-12T20:49:51.490211+00:00",
      "definition": {
```



```
        "templateLinked": {
            "policyTemplateId": "PTEXAMPLEEabcdefg111111"
        }
    },
    "lastUpdatedDate": "2023-06-12T20:49:51.490211+00:00",
    "policyId": "SPEXAMPLEEabcdefg333333",
    "policyStoreId": "PSEXAMPLEEabcdefg111111",
    "policyType": "TEMPLATE_LINKED",
    "principal": {
        "entityId": "alice",
        "entityType": "User"
    },
    "resource": {
        "entityId": "VacationPhoto94.jpg",
        "entityType": "Photo"
    }
}
]
```

For more information about policies, see [Amazon Verified Permissions policies](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [ListPolicies](#) in *AWS CLI Command Reference*.

list-policy-stores

The following code example shows how to use `list-policy-stores`.

AWS CLI

To list the available policy stores

The following `list-policy-stores` example lists all policy stores in the AWS Region. All commands for Verified Permissions except `create-policy-store` and `list-policy-stores` require that you specify the Id of the policy store you want to work with.

```
aws verifiedpermissions list-policy-stores
```

Output:

```
{
```

```
"policyStores": [  
  {  
    "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/  
PSEXAMPLEabcdefg111111",  
    "createdDate": "2023-06-05T20:16:46.225598+00:00",  
    "policyStoreId": "PSEXAMPLEabcdefg111111"  
  },  
  {  
    "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/  
PSEXAMPLEabcdefg222222",  
    "createdDate": "2023-06-08T18:09:37.364356+00:00",  
    "policyStoreId": "PSEXAMPLEabcdefg222222"  
  },  
  {  
    "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/  
PSEXAMPLEabcdefg333333",  
    "createdDate": "2023-06-08T18:09:46.920600+00:00",  
    "policyStoreId": "PSEXAMPLEabcdefg333333"  
  }  
]
```

For more information about policy stores, see [Amazon Verified Permissions policy stores](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [ListPolicyStores](#) in *AWS CLI Command Reference*.

list-policy-templates

The following code example shows how to use `list-policy-templates`.

AWS CLI

To list the available policy templates

The following `list-policy-templates` example lists all policy templates in the specified policy store.

```
aws verifiedpermissions list-policy-templates \  
  --policy-store-id PSEXAMPLEabcdefg111111
```

Output:

```
{
  "policyTemplates": [
    {
      "createdDate": "2023-06-12T20:47:42.804511+00:00",
      "lastUpdatedDate": "2023-06-12T20:47:42.804511+00:00",
      "policyStoreId": "PSEXAMPLEabcdefg111111",
      "policyTemplateId": "PTEXAMPLEabcdefg111111"
    }
  ]
}
```

For more information about policy templates, see [Amazon Verified Permissions policy templates](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [ListPolicyTemplates](#) in *AWS CLI Command Reference*.

put -schema

The following code example shows how to use `put -schema`.

AWS CLI

To save a schema to a policy store

The following `put -schema` example creates or replaces the schema in the specified policy store.

The `cedarJson` parameter in the input file takes a string representation of a JSON object. It contains embedded quotation marks (") within the outermost quotation mark pair. This requires you to convert the JSON to a string by preceding all embedded quotation marks with a backslash character (\) and combining all lines into a single text line with no line breaks.

Example strings can be displayed wrapped across multiple lines here for readability, but the operation requires the parameters be submitted as single line strings.

```
aws verifiedpermissions put-schema --definition file://schema.txt --policy-store-id
PSEXAMPLEabcdefg111111
```

Contents of `schema.txt`:

```
{
  "cedarJson": "{\"MySampleNamespace\": {\"actions\": {\"remoteAccess\": {
```

```

    \"appliesTo\": {\"principalTypes\": [\"Employee\"]}},\"entityTypes\": {
    \"Employee\": {\"shape\": {\"attributes\": {\"jobLevel\": {\"type\":
    \"Long\"}},\"name\": {\"type\": \"String\"}},\"type\": \"Record\"}}}}}"
  }

```

Output:

```

{
  "policyStoreId": "PSEXAMPLEabcdefg111111",
  "namespaces": [
    "MySampleNamespace"
  ],
  "createdDate": "2023-06-14T17:47:13.999885+00:00",
  "lastUpdatedDate": "2023-06-14T17:47:13.999885+00:00"
}

```

For more information about schema, see [Policy store schema](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [PutSchema](#) in *AWS CLI Command Reference*.

update-identity-source

The following code example shows how to use `update-identity-source`.

AWS CLI

To update an identity source

The following `update-identity-source` example modifies the specified identity source by providing a new Cognito user pool configuration and changing the entity type returned by the identity source.

```

aws verifiedpermissions update-identity-source
  --identity-source-id ISEXAMPLEabcdefg111111 \
  --update-configuration file://config.txt \
  --principal-entity-type "Employee" \
  --policy-store-id PSEXAMPLEabcdefg111111

```

Contents of `config.txt`:

```

{

```

```
    "cognitoUserPoolConfiguration": {
      "userPoolArn": "arn:aws:cognito-idp:us-west-2:123456789012:userpool/
us-west-2_1a2b3c4d5",
      "clientIds":["a1b2c3d4e5f6g7h8i9j0kalbmc"]
    }
  }
```

Output:

```
{
  "createdDate": "2023-05-19T20:30:28.214829+00:00",
  "identitySourceId": "ISEXAMPLEabcdefg111111",
  "lastUpdatedDate": "2023-05-19T20:30:28.214829+00:00",
  "policyStoreId": "PSEXAMPLEabcdefg111111"
}
```

For more information about identity sources, see [Using Amazon Verified Permissions with identity providers](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [UpdateIdentitySource](#) in *AWS CLI Command Reference*.

update-policy-store

The following code example shows how to use `update-policy-store`.

AWS CLI

To update a policy store

The following `update-policy-store` example modifies a policy store by changing its validation setting.

```
aws verifiedpermissions update-policy-store \
  --validation-settings "mode=STRICT" \
  --policy-store-id PSEXAMPLEabcdefg111111
```

Output:

```
{
  "arn": "arn:aws:verifiedpermissions::123456789012:policy-store/
PSEXAMPLEabcdefg111111",
  "createdDate": "2023-05-16T17:41:29.103459+00:00",
```

```
"lastUpdatedDate": "2023-05-16T17:41:29.103459+00:00",  
"policyStoreId": "PSEXAMPLEEabcdefg111111"  
}
```

For more information about policy stores, see [Amazon Verified Permissions policy stores](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [UpdatePolicyStore](#) in *AWS CLI Command Reference*.

update-policy-template

The following code example shows how to use `update-policy-template`.

AWS CLI

Example 1: To update a policy template

The following `update-policy-template` example modifies the specified template-linked policy to replace its policy statement.

```
aws verifiedpermissions update-policy-template \  
  --policy-template-id PTEXAMPLEEabcdefg111111 \  
  --statement file://template1.txt \  
  --policy-store-id PSEXAMPLEEabcdefg111111
```

Contents of file `template1.txt`:

```
permit(  
  principal in ?principal,  
  action == Action::"view",  
  resource == Photo::"VacationPhoto94.jpg"  
);
```

Output:

```
{  
  "createdDate": "2023-06-12T20:47:42.804511+00:00",  
  "lastUpdatedDate": "2023-06-12T20:47:42.804511+00:00",  
  "policyStoreId": "PSEXAMPLEEabcdefg111111",  
  "policyTemplateId": "PTEXAMPLEEabcdefg111111"  
}
```

For more information about policy templates, see [Amazon Verified Permissions policy templates](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [UpdatePolicyTemplate](#) in *AWS CLI Command Reference*.

update-policy

The following code example shows how to use `update-policy`.

AWS CLI

Example 1: To create a static policy

The following `create-policy` example creates a static policy with a policy scope that specifies both a principal and a resource.

```
aws verifiedpermissions create-policy \  
  --definition file://definition.txt \  
  --policy-store-id PSEXAMPLEEabcdefg111111
```

The `statement` parameter takes a string representation of a JSON object. It contains embedded quotation marks (") within the outermost quotation mark pair. This requires you to convert the JSON to a string by preceding all embedded quotation marks with a backslash character (\) and combining all lines into a single text line with no line breaks.

Example strings can be displayed wrapped across multiple lines here for readability, but the operation requires the parameters be submitted as single line strings.

Contents of file `definition.txt`:

```
{  
  "static": {  
    "description": "Grant everyone of janeFriends UserGroup access to the  
vacationFolder Album",  
    "statement": "permit(principal in UserGroup::\"janeFriends\", action,  
resource in Album::\"vacationFolder\" );"  
  }  
}
```

Output:

```
{
  "createdDate": "2023-06-12T20:33:37.382907+00:00",
  "lastUpdatedDate": "2023-06-12T20:33:37.382907+00:00",
  "policyId": "SPEXAMPLEabcdefg111111",
  "policyStoreId": "PSEXAMPLEabcdefg111111",
  "policyType": "STATIC",
  "principal": {
    "entityId": "janeFriends",
    "entityType": "UserGroup"
  },
  "resource": {
    "entityId": "vacationFolder",
    "entityType": "Album"
  }
}
```

Example 2: To create a static policy that grants access to a resource to everyone

The following create-policy example creates a static policy with a policy scope that specifies only a resource.

```
aws verifiedpermissions create-policy \
  --definition file://definition2.txt \
  --policy-store-id PSEXAMPLEabcdefg111111
```

Contents of file definition2.txt:

```
{
  "static": {
    "description": "Grant everyone access to the publicFolder Album",
    "statement": "permit(principal, action, resource in Album:\""publicFolder
  \");"
  }
}
```

Output:

```
{
  "createdDate": "2023-06-12T20:39:44.975897+00:00",
  "lastUpdatedDate": "2023-06-12T20:39:44.975897+00:00",
  "policyId": "PbfR73F8oh5MMfr9uRtFDB",
```



```

    "policyStoreId": "PSEXAMPLEEabcdefg222222",
    "policyType": "STATIC",
    "resource": {
      "entityId": "publicFolder",
      "entityType": "Album"
    }
  }
}

```

Example 3: To create a template-linked policy that is associated with the specified template

The following `create-policy` example creates a template-linked policy using the specified policy template and associates the specified principal to use with the new template-linked policy.

```

aws verifiedpermissions create-policy \
  --definition file://definition2.txt \
  --policy-store-id PSEXAMPLEEabcdefg111111

```

Contents of `definition3.txt`:

```

{
  "templateLinked": {
    "policyTemplateId": "PTEXAMPLEEabcdefg111111",
    "principal": {
      "entityType": "User",
      "entityId": "alice"
    }
  }
}

```

Output:

```

{
  "createdDate": "2023-06-12T20:49:51.490211+00:00",
  "lastUpdatedDate": "2023-06-12T20:49:51.490211+00:00",
  "policyId": "TPEXAMPLEEabcdefg111111",
  "policyStoreId": "PSEXAMPLEEabcdefg111111",
  "policyType": "TEMPLATE_LINKED",
  "principal": {
    "entityId": "alice",
    "entityType": "User"
  },
}

```

```
"resource": {
  "entityId": "VacationPhoto94.jpg",
  "entityType": "Photo"
}
```

For more information about policies, see [Amazon Verified Permissions policies](#) in the *Amazon Verified Permissions User Guide*.

- For API details, see [UpdatePolicy](#) in *AWS CLI Command Reference*.

VPC Lattice examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with VPC Lattice.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-listener

The following code example shows how to use `create-listener`.

AWS CLI

To create a listener

The following `create-listener` example creates an HTTPS listener with a default rule that forwards traffic to the specified VPC Lattice target group.

```
aws vpc-lattice create-listener \  
  --name my-service-listener \  
  --protocol HTTPS \  
  --port 443 \  
  --service-identifier svc-0285b53b2eEXAMPLE \  
  --default-action file://listener-config.json
```

Contents of listener-config.json:

```
{  
  "forward": {  
    "targetGroups": [  
      {  
        "targetGroupIdentifier": "tg-0eaa4b9ab4EXAMPLE"  
      }  
    ]  
  }  
}
```

Output:

```
{  
  "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:service/  
svc-0285b53b2eEXAMPLE/listener/listener-07cc7fb0abEXAMPLE",  
  "defaultAction": {  
    "forward": {  
      "targetGroups": [  
        {  
          "targetGroupIdentifier": "tg-0eaa4b9ab4EXAMPLE",  
          "weight": 100  
        }  
      ]  
    }  
  },  
  "id": "listener-07cc7fb0abEXAMPLE",  
  "name": "my-service-listener",  
  "port": 443,  
  "protocol": "HTTPS",  
  "serviceArn": "arn:aws:vpc-lattice:us-east-2:123456789012:service/  
svc-0285b53b2eEXAMPLE",  
  "serviceId": "svc-0285b53b2eEXAMPLE"  
}
```

For more information, see [Listeners](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [CreateListener](#) in *AWS CLI Command Reference*.

create-service-network-service-association

The following code example shows how to use `create-service-network-service-association`.

AWS CLI

To create a service association

The following `create-service-network-service-association` example associates the specified service with the specified service network.

```
aws vpc-lattice create-service-network-service-association \  
  --service-identifier svc-0285b53b2eEXAMPLE \  
  --service-network-identifier sn-080ec7dc93EXAMPLE
```

Output:

```
{  
  "arn": "arn:aws:vpc-lattice:us-  
east-2:123456789012:servicenetworkserviceassociation/snsa-0e16955a8cEXAMPLE",  
  "createdBy": "123456789012",  
  "dnsEntry": {  
    "domainName": "my-lattice-service-0285b53b2eEXAMPLE.7d67968.vpc-lattice-  
svcs.us-east-2.on.aws",  
    "hostedZoneId": "Z09127221KTH2CEXAMPLE"  
  },  
  "id": "snsa-0e16955a8cEXAMPLE",  
  "status": "CREATE_IN_PROGRESS"  
}
```

For more information, see [Manage service associations](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [CreateServiceNetworkServiceAssociation](#) in *AWS CLI Command Reference*.

create-service-network-vpc-association

The following code example shows how to use `create-service-network-vpc-association`.

AWS CLI

To create a VPC association

The following `create-service-network-vpc-association` example associates the specified vpc with the specified service network. The specified security group controls which resources in the VPC can access the service network and its services.

```
aws vpc-lattice create-service-network-vpc-association \  
  --vpc-identifier vpc-0a1b2c3d4eEXAMPLE \  
  --service-network-identifier sn-080ec7dc93EXAMPLE \  
  --security-group-ids sg-0aee16bc6cEXAMPLE
```

Output:

```
{  
  "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:servicenetworkvpcassociation/  
snva-0821fc8631EXAMPLE",  
  "createdBy": "123456789012",  
  "id": "snva-0821fc8631EXAMPLE",  
  "securityGroupIds": [  
    "sg-0aee16bc6cEXAMPLE"  
  ],  
  "status": "CREATE_IN_PROGRESS"  
}
```

For more information, see [Manage VPC associations](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [CreateServiceNetworkVpcAssociation](#) in *AWS CLI Command Reference*.

create-service-network

The following code example shows how to use `create-service-network`.

AWS CLI

To create a service network

The following `create-service-network` example creates a service network with the specified name.

```
aws vpc-lattice create-service-network \  
  --name my-service-network
```

```
--name my-service-network
```

Output:

```
{
  "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:servicenetwork/
sn-080ec7dc93EXAMPLE",
  "authType": "NONE",
  "id": "sn-080ec7dc93EXAMPLE",
  "name": "my-service-network"
}
```

For more information, see [Service networks](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [CreateServiceNetwork](#) in *AWS CLI Command Reference*.

create-service

The following code example shows how to use `create-service`.

AWS CLI

To create a service

The following `create-service` example creates a service with the specified name.

```
aws vpc-lattice create-service \
  --name my-lattice-service
```

Output:

```
{
  "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:service/
svc-0285b53b2eEXAMPLE",
  "authType": "NONE",
  "dnsEntry": {
    "domainName": "my-lattice-service-0285b53b2eEXAMPLE.1a2b3c4.vpc-lattice-
svcs.us-east-2.on.aws",
    "hostedZoneId": "Z09127221KTH2CEXAMPLE"
  },
  "id": "svc-0285b53b2eEXAMPLE",
  "name": "my-lattice-service",
  "status": "CREATE_IN_PROGRESS"
}
```

```
}
```

For more information, see [Services in VPC Lattice](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [CreateService](#) in *AWS CLI Command Reference*.

create-target-group

The following code example shows how to use `create-target-group`.

AWS CLI

Example 1: To create a target group of type INSTANCE

The following `create-target-group` example creates a target group with the specified name, type, and configuration.

```
aws vpc-lattice create-target-group \  
  --name my-lattice-target-group-instance \  
  --type INSTANCE \  
  --config file://tg-config.json
```

Contents of `tg-config.json`:

```
{  
  "port": 443,  
  "protocol": "HTTPS",  
  "protocolVersion": "HTTP1",  
  "vpcIdentifier": "vpc-f1663d9868EXAMPLE"  
}
```

Output:

```
{  
  "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:targetgroup/  
tg-0eaa4b9ab4EXAMPLE",  
  "config": {  
    "healthCheck": {  
      "enabled": true,  
      "healthCheckIntervalSeconds": 30,  
      "healthCheckTimeoutSeconds": 5,  
      "healthyThresholdCount": 5,  
    }  
  }  
}
```

```

    "matcher": {
      "httpCode": "200"
    },
    "path": "/",
    "protocol": "HTTPS",
    "protocolVersion": "HTTP1",
    "unhealthyThresholdCount": 2
  },
  "port": 443,
  "protocol": "HTTPS",
  "protocolVersion": "HTTP1",
  "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
},
"id": "tg-0eaa4b9ab4EXAMPLE",
"name": "my-lattice-target-group-instance",
"status": "CREATE_IN_PROGRESS",
"type": "INSTANCE"
}

```

Example 2: To create a target group of type IP

The following `create-target-group` example creates a target group with the specified name, type, and configuration.

```

aws vpc-lattice create-target-group \
  --name my-lattice-target-group-ip \
  --type IP \
  --config file://tg-config.json

```

Contents of `tg-config.json`:

```

{
  "ipAddressType": "IPV4",
  "port": 443,
  "protocol": "HTTPS",
  "protocolVersion": "HTTP1",
  "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
}

```

Output:

```

{

```



```

    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:targetgroup/
tg-0eaa4b9ab4EXAMPLE",
    "config": {
      "healthCheck": {
        "enabled": true,
        "healthCheckIntervalSeconds": 30,
        "healthCheckTimeoutSeconds": 5,
        "healthyThresholdCount": 5,
        "matcher": {
          "httpCode": "200"
        },
        "path": "/",
        "protocol": "HTTPS",
        "protocolVersion": "HTTP1",
        "unhealthyThresholdCount": 2
      },
      "ipAddressType": "IPV4",
      "port": 443,
      "protocol": "HTTPS",
      "protocolVersion": "HTTP1",
      "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
    },
    "id": "tg-0eaa4b9ab4EXAMPLE",
    "name": "my-lattice-target-group-ip",
    "status": "CREATE_IN_PROGRESS",
    "type": "IP"
  }

```

Example 3: To create a target group of type LAMBDA

The following `create-target-group` example creates a target group with the specified name, type, and configuration.

```

aws vpc-lattice create-target-group \
  --name my-lattice-target-group-lambda \
  --type LAMBDA

```

Output:

```

{
  "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:targetgroup/
tg-0eaa4b9ab4EXAMPLE",

```

```
"id": "tg-0eaa4b9ab4EXAMPLE",
"name": "my-lattice-target-group-lambda",
"status": "CREATE_IN_PROGRESS",
"type": "LAMBDA"
}
```

Example 4: To create a target group of type ALB

The following `create-target-group` example creates a target group with the specified name, type, and configuration.

```
aws vpc-lattice create-target-group \
  --name my-lattice-target-group-alb \
  --type ALB \
  --config file://tg-config.json
```

Contents of `tg-config.json`:

```
{
  "port": 443,
  "protocol": "HTTPS",
  "protocolVersion": "HTTP1",
  "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
}
```

Output:

```
{
  "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:targetgroup/
tg-0eaa4b9ab4EXAMPLE",
  "config": {
    "port": 443,
    "protocol": "HTTPS",
    "protocolVersion": "HTTP1",
    "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
  },
  "id": "tg-0eaa4b9ab4EXAMPLE",
  "name": "my-lattice-target-group-alb",
  "status": "CREATE_IN_PROGRESS",
  "type": "ALB"
}
```

For more information, see [Target groups](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [CreateTargetGroup](#) in *AWS CLI Command Reference*.

delete-auth-policy

The following code example shows how to use delete-auth-policy.

AWS CLI

To delete an auth policy

The following delete-auth-policy example deletes the auth policy for the specified service.

```
aws vpc-lattice delete-auth-policy \  
  --resource-identifier svc-0285b53b2eEXAMPLE
```

This command produces no output.

For more information, see [Auth policies](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [DeleteAuthPolicy](#) in *AWS CLI Command Reference*.

delete-listener

The following code example shows how to use delete-listener.

AWS CLI

To delete a listener

The following delete-listener example deletes the specified listener.

```
aws vpc-lattice delete-listener \  
  --listener-identifier listener-07cc7fb0abEXAMPLE \  
  --service-identifier svc-0285b53b2eEXAMPLE
```

This command produces no output.

For more information, see [Listeners](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [DeleteListener](#) in *AWS CLI Command Reference*.

delete-service-network-service-association

The following code example shows how to use `delete-service-network-service-association`.

AWS CLI

To delete a service association

The following `delete-service-network-service-association` example disassociates the specified service association.

```
aws vpc-lattice delete-service-network-service-association \  
  --service-network-service-association-identifier sns-a-031fabb4d8EXAMPLE
```

Output:

```
{  
  "arn": "arn:aws:vpc-lattice:us-  
east-2:123456789012:servicenetworkserviceassociation/sns-a-031fabb4d8EXAMPLE",  
  "id": "sns-a-031fabb4d8EXAMPLE",  
  "status": "DELETE_IN_PROGRESS"  
}
```

For more information, see [Manage service associations](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [DeleteServiceNetworkServiceAssociation](#) in *AWS CLI Command Reference*.

delete-service-network-vpc-association

The following code example shows how to use `delete-service-network-vpc-association`.

AWS CLI

To delete a VPC association

The following `delete-service-network-vpc-association` example disassociates the specified VPC association.

```
aws vpc-lattice delete-service-network-vpc-association \  
  --service-network-vpc-association-identifier snva-0821fc8631EXAMPLE
```

Output:

```
{
  "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:servicenetworkvpcassociation/
snva-0821fc8631EXAMPLE",
  "id": "snva-0821fc8631EXAMPLE",
  "status": "DELETE_IN_PROGRESS"
}
```

For more information, see [Manage VPC associations](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [DeleteServiceNetworkVpcAssociation](#) in *AWS CLI Command Reference*.

delete-service-network

The following code example shows how to use `delete-service-network`.

AWS CLI**To delete a service network**

The following `delete-service-network` example deletes the specified service network.

```
aws vpc-lattice delete-service-network \
  --service-network-identifier sn-080ec7dc93EXAMPLE
```

This command produces no output.

For more information, see [Service networks](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [DeleteServiceNetwork](#) in *AWS CLI Command Reference*.

delete-service

The following code example shows how to use `delete-service`.

AWS CLI**To delete a service**

The following `delete-service` example deletes the specified service.

```
aws vpc-lattice delete-service \  
  --service-identifier svc-0285b53b2eEXAMPLE
```

Output:

```
{  
  "arn": "arn:aws:vpc-lattice:us-west-2:123456789012:service/  
svc-0285b53b2eEXAMPLE",  
  "id": "svc-0285b53b2eEXAMPLE",  
  "name": "my-lattice-service",  
  "status": "DELETE_IN_PROGRESS"  
}
```

For more information, see [Services in VPC Lattice](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [DeleteService](#) in *AWS CLI Command Reference*.

delete-target-group

The following code example shows how to use delete-target-group.

AWS CLI

To delete a target group

The following delete-target-group example deletes the specified target group.

```
aws vpc-lattice delete-target-group \  
  --target-group-identifier tg-0eaa4b9ab4EXAMPLE
```

Output:

```
{  
  "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:targetgroup/  
tg-0eaa4b9ab4EXAMPLE",  
  "id": "tg-0eaa4b9ab4EXAMPLE",  
  "status": "DELETE_IN_PROGRESS"  
}
```

For more information, see [Target groups](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [DeleteTargetGroup](#) in *AWS CLI Command Reference*.

deregister-targets

The following code example shows how to use `deregister-targets`.

AWS CLI

To deregister a target

The following `deregister-targets` example deregisters the specified target from the specified target group.

```
aws vpc-lattice deregister-targets \  
  --targets i-07dd579bc5EXAMPLE \  
  --target-group-identifier tg-0eaa4b9ab4EXAMPLE
```

Output:

```
{  
  "successful": [  
    {  
      "id": "i-07dd579bc5EXAMPLE",  
      "port": 443  
    }  
  ],  
  "unsuccessful": []  
}
```

For more information, see [Register targets](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [DeregisterTargets](#) in *AWS CLI Command Reference*.

get-auth-policy

The following code example shows how to use `get-auth-policy`.

AWS CLI

To get information about an auth policy

The following `get-auth-policy` example gets information about the auth policy for the specified service.

```
aws vpc-lattice get-auth-policy \  
  --resource-identifier svc-0285b53b2eEXAMPLE
```

Output:

```
{  
  "createdAt": "2023-06-07T03:51:20.266Z",  
  "lastUpdatedAt": "2023-06-07T04:39:27.082Z",  
  "policy": "{\n\"Version\": \"2012-10-17\", \"Statement\": [\n{\n\"Effect\": \"Allow\", \"Principal\": {\n\"AWS\": \"arn:aws:iam:123456789012:role/my-clients\"},\n\"Action\": \"vpc-lattice-svcs:Invoke\", \"Resource\": \"arn:aws:vpc-lattice:us-east-2:123456789012:service/svc-0285b53b2eEXAMPLE\"}]]\",  
  \"state\": \"Active\"  
}
```

For more information, see [Auth policies](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [GetAuthPolicy](#) in *AWS CLI Command Reference*.

get-listener

The following code example shows how to use `get-listener`.

AWS CLI

To get information about a service listener

The following `get-listener` example gets information about the specified listener for the specified service.

```
aws vpc-lattice get-listener \  
  --listener-identifier listener-0ccf55918cEXAMPLE \  
  --service-identifier svc-0285b53b2eEXAMPLE
```

Output:

```
{  
  "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:service/  
svc-0285b53b2eEXAMPLE/listener/listener-0ccf55918cEXAMPLE",  
  "createdAt": "2023-05-07T05:08:45.192Z",  
  "defaultAction": {
```



```

    "forward": {
      "targetGroups": [
        {
          "targetGroupIdentifier": "tg-0ff213abb6EXAMPLE",
          "weight": 1
        }
      ]
    },
    "id": "listener-0ccf55918cEXAMPLE",
    "lastUpdatedAt": "2023-05-07T05:08:45.192Z",
    "name": "http-80",
    "port": 80,
    "protocol": "HTTP",
    "serviceArn": "arn:aws:vpc-lattice:us-east-2:123456789012:service/
svc-0285b53b2eEXAMPLE",
    "serviceId": "svc-0285b53b2eEXAMPLE"
  }
}

```

For more information, see [Define routing](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [GetListener](#) in *AWS CLI Command Reference*.

get-service-network-service-association

The following code example shows how to use `get-service-network-service-association`.

AWS CLI

To get information about a service association

The following `get-service-network-service-association` example gets information about the specified service association.

```

aws vpc-lattice get-service-network-service-association \
  --service-network-service-association-identifier sns-a-031fabb4d8EXAMPLE

```

Output:

```

{
  "arn": "arn:aws:vpc-lattice:us-
east-2:123456789012:servicenetworkserviceassociation/sns-a-031fabb4d8EXAMPLE",

```

```

    "createdAt": "2023-05-05T21:48:16.076Z",
    "createdBy": "123456789012",
    "dnsEntry": {
      "domainName": "my-lattice-service-0285b53b2eEXAMPLE.7d67968.vpc-lattice-
svcs.us-east-2.on.aws",
      "hostedZoneId": "Z09127221KTH2CEXAMPLE"
    },
    "id": "sna-031fabb4d8EXAMPLE",
    "serviceArn": "arn:aws:vpc-lattice:us-east-2:123456789012:service/
svc-0285b53b2eEXAMPLE",
    "serviceId": "svc-0285b53b2eEXAMPLE",
    "serviceName": "my-lattice-service",
    "serviceNetworkArn": "arn:aws:vpc-lattice:us-east-2:123456789012:servicenetwork/
sn-080ec7dc93EXAMPLE",
    "serviceNetworkId": "sn-080ec7dc93EXAMPLE",
    "serviceNetworkName": "my-service-network",
    "status": "ACTIVE"
  }
}

```

For more information, see [Manage service associations](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [GetServiceNetworkServiceAssociation](#) in *AWS CLI Command Reference*.

get-service-network-vpc-association

The following code example shows how to use `get-service-network-vpc-association`.

AWS CLI

To get information about a VPC association

The following `get-service-network-vpc-association` example gets information about the specified VPC association.

```

aws vpc-lattice get-service-network-vpc-association \
  --service-network-vpc-association-identifier snva-0821fc8631EXAMPLE

```

Output:

```

{
  "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:servicenetworkvpcassociation/
snva-0821fc8631EXAMPLE",

```

```

    "createdAt": "2023-06-06T23:41:08.421Z",
    "createdBy": "123456789012",
    "id": "snva-0c5dcb60d6EXAMPLE",
    "lastUpdatedAt": "2023-06-06T23:41:08.421Z",
    "securityGroupIds": [
      "sg-0aee16bc6cEXAMPLE"
    ],
    "serviceNetworkArn": "arn:aws:vpc-lattice:us-east-2:123456789012:servicenetwork/
sn-080ec7dc93EXAMPLE",
    "serviceNetworkId": "sn-080ec7dc93EXAMPLE",
    "serviceNetworkName": "my-service-network",
    "status": "ACTIVE",
    "vpcId": "vpc-0a1b2c3d4eEXAMPLE"
  }

```

For more information, see [Manage VPC associations](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [GetServiceNetworkVpcAssociation](#) in *AWS CLI Command Reference*.

get-service-network

The following code example shows how to use `get-service-network`.

AWS CLI

To get information about a service network

The following `get-service-network` example gets information about the specified service network.

```

aws vpc-lattice get-service-network \
  --service-network-identifier sn-080ec7dc93EXAMPLE

```

Output:

```

{
  "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:servicenetwork/
sn-080ec7dc93EXAMPLE",
  "authType": "AWS_IAM",
  "createdAt": "2023-05-05T15:26:08.417Z",
  "id": "sn-080ec7dc93EXAMPLE",
  "lastUpdatedAt": "2023-05-05T15:26:08.417Z",

```

```
"name": "my-service-network",
"numberOfAssociatedServices": 2,
"numberOfAssociatedVPCs": 3
}
```

For more information, see [Service networks](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [GetServiceNetwork](#) in *AWS CLI Command Reference*.

get-service

The following code example shows how to use `get-service`.

AWS CLI

To get information about a service

The following `get-service` example gets information about the specified service.

```
aws vpc-lattice get-service \
  --service-identifier svc-0285b53b2eEXAMPLE
```

Output:

```
{
  "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:service/
svc-0285b53b2eEXAMPLE",
  "authType": "AWS_IAM",
  "createdAt": "2023-05-05T21:35:29.339Z",
  "dnsEntry": {
    "domainName": "my-lattice-service-0285b53b2eEXAMPLE.7d67968.vpc-lattice-
svcs.us-east-2.on.aws",
    "hostedZoneId": "Z09127221KTH2CFU0HIZH"
  },
  "id": "svc-0285b53b2eEXAMPLE",
  "lastUpdatedAt": "2023-05-05T21:35:29.339Z",
  "name": "my-lattice-service",
  "status": "ACTIVE"
}
```

For more information, see [Services](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [GetService](#) in *AWS CLI Command Reference*.

get-target-group

The following code example shows how to use `get-target-group`.

AWS CLI

To get information about a target group

The following `get-target-group` example gets information about the specified target group, which has a target type of `INSTANCE`.

```
aws vpc-lattice get-target-group \  
  --target-group-identifier tg-0eaa4b9ab4EXAMPLE
```

Output:

```
{  
  "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:targetgroup/  
tg-0eaa4b9ab4EXAMPLE",  
  "config": {  
    "healthCheck": {  
      "enabled": true,  
      "healthCheckIntervalSeconds": 30,  
      "healthCheckTimeoutSeconds": 5,  
      "healthyThresholdCount": 5,  
      "matcher": {  
        "httpCode": "200"  
      },  
      "path": "/",  
      "protocol": "HTTPS",  
      "protocolVersion": "HTTP1",  
      "unhealthyThresholdCount": 2  
    },  
    "port": 443,  
    "protocol": "HTTPS",  
    "protocolVersion": "HTTP1",  
    "vpcIdentifier": "vpc-f1663d9868EXAMPLE"  
  },  
  "createdAt": "2023-05-06T04:41:04.122Z",  
  "id": "tg-0eaa4b9ab4EXAMPLE",  
  "lastUpdatedAt": "2023-05-06T04:41:04.122Z",  
  "name": "my-target-group",  
  "serviceArns": [  

```

```
    "arn:aws:vpc-lattice:us-east-2:123456789012:service/svc-0285b53b2eEXAMPLE"  
  ],  
  "status": "ACTIVE",  
  "type": "INSTANCE"  
}
```

For more information, see [Target groups](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [GetTargetGroup](#) in *AWS CLI Command Reference*.

list-listeners

The following code example shows how to use `list-listeners`.

AWS CLI

To list service listeners

The following `list-listeners` example lists the listeners for the specified service.

```
aws vpc-lattice list-listeners \  
  --service-identifier svc-0285b53b2eEXAMPLE
```

Output:

```
{  
  "items": [  
    {  
      "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:service/  
svc-0285b53b2eEXAMPLE/listener/listener-0ccf55918cEXAMPLE",  
      "createdAt": "2023-05-07T05:08:45.192Z",  
      "id": "listener-0ccf55918cEXAMPLE",  
      "lastUpdatedAt": "2023-05-07T05:08:45.192Z",  
      "name": "http-80",  
      "port": 80,  
      "protocol": "HTTP"  
    }  
  ]  
}
```

For more information, see [Define routing](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [ListListeners](#) in *AWS CLI Command Reference*.

list-service-network-service-associations

The following code example shows how to use `list-service-network-service-associations`.

AWS CLI

To list service associations

The following `list-service-network-service-associations` example lists the service associations for the specified service network. The `--query` option scopes the output to the IDs of the service associations.

```
aws vpc-lattice list-service-network-service-associations \  
  --service-network-identifier sn-080ec7dc93EXAMPLE \  
  --query items[*].id
```

Output:

```
[  
  "snsa-031fabb4d8EXAMPLE",  
  "snsa-0e16955a8cEXAMPLE"  
]
```

For more information, see [Manage service associations](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [ListServiceNetworkServiceAssociations](#) in *AWS CLI Command Reference*.

list-service-network-vpc-associations

The following code example shows how to use `list-service-network-vpc-associations`.

AWS CLI

To list VPC associations

The following `list-service-network-vpc-associations` example lists the VPC associations for the specified service network. The `--query` option scopes the output to the IDs of the VPC associations.

```
aws vpc-lattice list-service-network-vpc-associations \  
  --service-network-identifier sn-080ec7dc93EXAMPLE \  
  --query items[*].id
```

Output:

```
[  
  "snva-0821fc8631EXAMPLE",  
  "snva-0c5dcb60d6EXAMPLE"  
]
```

For more information, see [Manage VPC associations](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [ListServiceNetworkVpcAssociations](#) in *AWS CLI Command Reference*.

list-service-networks

The following code example shows how to use `list-service-networks`.

AWS CLI

To list your service networks

The following `list-service-networks` example lists the service networks owned or shared with the calling account. The `--query` option scopes the results to the Amazon Resource Names (ARN) of the service networks.

```
aws vpc-lattice list-service-networks \  
  --query items[*].arn
```

Output:

```
[  
  "arn:aws:vpc-lattice:us-east-2:123456789012:servicenetwork/  
sn-080ec7dc93EXAMPLE",  
  "arn:aws:vpc-lattice:us-east-2:111122223333:servicenetwork/sn-0ec4d436cfEXAMPLE"  
]
```

For more information, see [Service networks](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [ListServiceNetworks](#) in *AWS CLI Command Reference*.

list-services

The following code example shows how to use `list-services`.

AWS CLI

To list your services

The following `list-services` example lists the services owned or shared with the calling account. The `--query` option scopes the results to the Amazon Resource Names (ARN) of the services.

```
aws vpc-lattice list-services \  
  --query items[*].arn
```

Output:

```
[  
  "arn:aws:vpc-lattice:us-east-2:123456789012:service/svc-0285b53b2eEXAMPLE",  
  "arn:aws:vpc-lattice:us-east-2:111122223333:service/svc-0b8ac96550EXAMPLE"  
]
```

For more information, see [Services](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [ListServices](#) in *AWS CLI Command Reference*.

list-target-groups

The following code example shows how to use `list-target-groups`.

AWS CLI

To list your target groups

The following `list-target-groups` example lists the target groups with a target type of LAMBDA.

```
aws vpc-lattice list-target-groups \  
  --target-group-type LAMBDA
```

Output:

```
{
  "items": [
    {
      "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:targetgroup/
tg-045c1b7d9dEXAMPLE",
      "createdAt": "2023-05-06T05:22:16.637Z",
      "id": "tg-045c1b7d9dEXAMPLE",
      "lastUpdatedAt": "2023-05-06T05:22:16.637Z",
      "name": "my-target-group-lam",
      "serviceArns": [
        "arn:aws:vpc-lattice:us-east-2:123456789012:service/
svc-0285b53b2eEXAMPLE"
      ],
      "status": "ACTIVE",
      "type": "LAMBDA"
    }
  ]
}
```

For more information, see [Target groups](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [ListTargetGroups](#) in *AWS CLI Command Reference*.

list-targets

The following code example shows how to use `list-targets`.

AWS CLI

To list the targets for a target group

The following `list-targets` example lists the targets for the specified target group.

```
aws vpc-lattice list-targets \
  --target-group-identifier tg-0eaa4b9ab4EXAMPLE
```

Output:

```
{
  "items": [
    {
      "id": "i-07dd579bc5EXAMPLE",
```

```

        "port": 443,
        "status": "HEALTHY"
    },
    {
        "id": "i-047b3c9078EXAMPLE",
        "port": 443,
        "reasonCode": "HealthCheckFailed",
        "status": "UNHEALTHY"
    }
]
}

```

For more information, see [Target groups](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [ListTargets](#) in *AWS CLI Command Reference*.

put-auth-policy

The following code example shows how to use `put-auth-policy`.

AWS CLI

To create an auth policy for a service

The following `put-auth-policy` example grants access to requests from any authenticated principal that uses the specified IAM role. The resource is the ARN of the service to which the policy is attached.

```

aws vpc-lattice put-auth-policy \
  --resource-identifier svc-0285b53b2eEXAMPLE \
  --policy file://auth-policy.json

```

Contents of `auth-policy.json`:

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:role/my-clients"
      }
    },
  ],
}

```

```

        "Action": "vpc-lattice-svcs:Invoke",
        "Resource": "arn:aws:vpc-lattice:us-east-2:123456789012:service/
svc-0285b53b2eEXAMPLE"
    }
]
}

```

Output:

```

{
  "policy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",
\"Principal\":{\"AWS\":\"arn:aws:iam:123456789012:role/my-clients\"},
\"Action\":\"vpc-lattice-svcs:Invoke\",\"Resource\":\"arn:aws:vpc-lattice:us-
east-2:123456789012:service/svc-0285b53b2eEXAMPLE\"}]}",
  "state": "Active"
}

```

For more information, see [Auth policies](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [PutAuthPolicy](#) in *AWS CLI Command Reference*.

register-targets

The following code example shows how to use `register-targets`.

AWS CLI**To register a target**

The following `register-targets` example registers the specified targets with the specified target group.

```

aws vpc-lattice register-targets \
  --targets id=i-047b3c9078EXAMPLE id=i-07dd579bc5EXAMPLE \
  --target-group-identifier tg-0eaa4b9ab4EXAMPLE

```

Output:

```

{
  "successful": [
    {
      "id": "i-07dd579bc5EXAMPLE",

```

```
        "port": 443
      }
    ],
    "unsuccessful": [
      {
        "failureCode": "UnsupportedTarget",
        "failureMessage": "Instance targets must be in the same VPC as their
target group",
        "id": "i-047b3c9078EXAMPLE",
        "port": 443
      }
    ]
  ]
}
```

For more information, see [Register targets](#) in the *Amazon VPC Lattice User Guide*.

- For API details, see [RegisterTargets](#) in *AWS CLI Command Reference*.

AWS WAF Classic examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS WAF Classic.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

put-logging-configuration

The following code example shows how to use `put-logging-configuration`.

AWS CLI

To create a logging configuration for the web ACL ARN with the specified Kinesis Firehose stream ARN

The following `put-logging-configuration` example displays logging configuration for WAF with CloudFront.

```
aws waf put-logging-configuration \  
  --logging-configuration ResourceArn=arn:aws:waf::123456789012:webacl/3bffd3ed-  
fa2e-445e-869f-a6a7cf153fd3,LogDestinationConfigs=arn:aws:firehose:us-  
east-1:123456789012:deliverystream/aws-waf-logs-firehose-stream,RedactedFields=[]
```

Output:

```
{  
  "LoggingConfiguration": {  
    "ResourceArn": "arn:aws:waf::123456789012:webacl/3bffd3ed-fa2e-445e-869f-  
a6a7cf153fd3",  
    "LogDestinationConfigs": [  
      "arn:aws:firehose:us-east-1:123456789012:deliverystream/aws-waf-logs-  
firehose-stream"  
    ]  
  }  
}
```

- For API details, see [PutLoggingConfiguration](#) in *AWS CLI Command Reference*.

update-byte-match-set

The following code example shows how to use `update-byte-match-set`.

AWS CLI

To update a byte match set

The following `update-byte-match-set` command deletes a `ByteMatchTuple` object (`filter`) in a `ByteMatchSet`:

```
aws waf update-byte-match-set --byte-match-set-id a123fae4-  
b567-8e90-1234-5ab67ac8ca90 --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 --
```

```
updates
```

```
Action="DELETE",ByteMatchTuple={FieldToMatch={Type="HEADER",Data="referer"},TargetString="b
```

For more information, see [Working with String Match Conditions](#) in the *AWS WAF developer guide*.

- For API details, see [UpdateByteMatchSet](#) in *AWS CLI Command Reference*.

update-ip-set

The following code example shows how to use `update-ip-set`.

AWS CLI

To update an IP set

The following `update-ip-set` command updates an IPSet with an IPv4 address and deletes an IPv6 address:

```
aws waf update-ip-set --ip-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90
--change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 --updates
Action="INSERT",IPSetDescriptor={Type="IPV4",Value="12.34.56.78/16"},Action="DELETE",IPSetD
```

Alternatively you can use a JSON file to specify the input. For example:

```
aws waf update-ip-set --ip-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90 --change-
token 12cs345-67cd-890b-1cd2-c3a4567d89f1 --updates file://change.json
```

Where content of the JSON file is:

```
[
{
  "Action": "INSERT",
  "IPSetDescriptor":
  {
    "Type": "IPV4",
    "Value": "12.34.56.78/16"
  }
},
{
  "Action": "DELETE",
  "IPSetDescriptor":
```

```
{
  "Type": "IPV6",
  "Value": "1111:0000:0000:0000:0000:0000:0000:0111/128"
}
}
]
```

For more information, see *Working with IP Match Conditions* in the *AWS WAF developer guide*.

- For API details, see [UpdateIpSet](#) in *AWS CLI Command Reference*.

update-rule

The following code example shows how to use `update-rule`.

AWS CLI

To update a rule

The following `update-rule` command deletes a Predicate object in a rule:

```
aws waf update-rule --rule-id a123fae4-b567-8e90-1234-5ab67ac8ca90
--change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 --updates
Action="DELETE",Predicate={Negated=false,Type="ByteMatch",DataId="MyByteMatchSetID"}
```

For more information, see *Working with Rules* in the *AWS WAF developer guide*.

- For API details, see [UpdateRule](#) in *AWS CLI Command Reference*.

update-size-constraint-set

The following code example shows how to use `update-size-constraint-set`.

AWS CLI

To update a size constraint set

The following `update-size-constraint-set` command deletes a `SizeConstraint` object (filters) in a size constraint set:

```
aws waf update-size-constraint-set --size-constraint-set-id a123fae4-
b567-8e90-1234-5ab67ac8ca90 --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 --
```



```
updates
```

```
Action="DELETE",SizeConstraint={FieldToMatch={Type="QUERY_STRING"},TextTransformation="NONE"
```

For more information, see [Working with Size Constraint Conditions](#) in the *AWS WAF developer guide*.

- For API details, see [UpdateSizeConstraintSet](#) in *AWS CLI Command Reference*.

update-sql-injection-match-set

The following code example shows how to use `update-sql-injection-match-set`.

AWS CLI

To update a SQL Injection Match Set

The following `update-sql-injection-match-set` command deletes a `SqlInjectionMatchTuple` object (filters) in a SQL injection match set:

```
aws waf update-sql-injection-match-set --sql-injection-  
match-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90 --  
change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 --updates  
Action="DELETE",SqlInjectionMatchTuple={FieldToMatch={Type="QUERY_STRING"},TextTransformation="NONE"
```

For more information, see [Working with SQL Injection Match Conditions](#) in the *AWS WAF developer guide*.

- For API details, see [UpdateSqlInjectionMatchSet](#) in *AWS CLI Command Reference*.

update-web-acl

The following code example shows how to use `update-web-acl`.

AWS CLI

To update a web ACL

The following `update-web-acl` command deletes an `ActivatedRule` object in a WebACL.

```
aws waf update-web-acl --web-acl-id a123fae4-b567-8e90-1234-5ab67ac8ca90  
--change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 --updates
```

```
Action="DELETE",ActivatedRule='{Priority=1,RuleId="WAFRule-1-Example",Action={Type="ALLOW"},Type="REGULAR"}'
```

Output:

```
{
  "ChangeToken": "12cs345-67cd-890b-1cd2-c3a4567d89f1"
}
```

For more information, see [Working with Web ACLs](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [UpdateWebAcl](#) in *AWS CLI Command Reference*.

update-xss-match-set

The following code example shows how to use `update-xss-match-set`.

AWS CLI

To update an XSSMatchSet

The following `update-xss-match-set` command deletes an `XssMatchTuple` object (filters) in an `XssMatchSet`:

```
aws waf update-xss-match-set --xss-match-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90
--change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 --updates
Action="DELETE",XssMatchTuple={FieldToMatch={Type="QUERY_STRING"},TextTransformation="URL_D
```

For more information, see [Working with Cross-site Scripting Match Conditions](#) in the *AWS WAF developer guide*.

- For API details, see [UpdateXssMatchSet](#) in *AWS CLI Command Reference*.

AWS WAF Classic Regional examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS WAF Classic Regional.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-web-acl

The following code example shows how to use `associate-web-acl`.

AWS CLI

To associate a web ACL with a resource

The following `associate-web-acl` command associates a web ACL, specified by the `web-acl-id`, with a resource, specified by the `resource-arn`. The resource ARN can refer to either a application load balancer or an API Gateway:

```
aws waf-regional associate-web-acl \  
  --web-acl-id a123fae4-b567-8e90-1234-5ab67ac8ca90 \  
  --resource-arn 12cs345-67cd-890b-1cd2-c3a4567d89f1
```

For more information, see [Working with Web ACLs](#) in the *AWS WAF Developer Guide*.

- For API details, see [AssociateWebAcl](#) in *AWS CLI Command Reference*.

put-logging-configuration

The following code example shows how to use `put-logging-configuration`.

AWS CLI

To create a logging configuration for the web ACL ARN with the specified Kinesis Firehose stream ARN

The following `put-logging-configuration` example displays logging configuration for WAF with ALB/APIGateway in Region `us-east-1`.

```
aws waf-regional put-logging-configuration \  
  --logging-configuration ResourceArn=arn:aws:waf-  
regional:us-east-1:123456789012:webacl/3bffd3ed-fa2e-445e-869f-  
a6a7cf153fd3,LogDestinationConfigs=arn:aws:firehose:us-  
east-1:123456789012:deliverystream/aws-waf-logs-firehose-stream,RedactedFields=[] \  
  --region us-east-1
```

Output:

```
{  
  "LoggingConfiguration": {  
    "ResourceArn": "arn:aws:waf-regional:us-east-1:123456789012:webacl/3bffd3ed-  
fa2e-445e-869f-a6a7cf153fd3",  
    "LogDestinationConfigs": [  
      "arn:aws:firehose:us-east-1:123456789012:deliverystream/aws-waf-logs-  
firehose-stream"  
    ]  
  }  
}
```

- For API details, see [PutLoggingConfiguration](#) in *AWS CLI Command Reference*.

update-byte-match-set

The following code example shows how to use update-byte-match-set.

AWS CLI

To update a byte match set

The following update-byte-match-set command deletes a ByteMatchTuple object (filter) in a ByteMatchSet. Because the updates value has embedded double quotes, you must surround the value with single quotes.

```
aws waf-regional update-byte-match-set \  
  --byte-match-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90 \  
  --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 \  
  --updates  
'Action="DELETE",ByteMatchTuple={FieldToMatch={Type="HEADER",Data="referer"},TargetString="'
```

For more information, see [Working with String Match Conditions](#) in the *AWS WAF Developer Guide*.

- For API details, see [UpdateByteMatchSet](#) in *AWS CLI Command Reference*.

update-ip-set

The following code example shows how to use `update-ip-set`.

AWS CLI

To update an IP set

The following `update-ip-set` command updates an IPSet with an IPv4 address and deletes an IPv6 address. Get the value for `change-token` by running the `get-change-token` command. Because the value for `updates` includes embedded double-quotes, you must surround the value with single quotes.

```
aws waf update-ip-set \  
  --ip-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90 \  
  --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 \  
  --updates  
  'Action="INSERT",IPSetDescriptor={Type="IPV4",Value="12.34.56.78/16"},Action="DELETE",IPSet'
```

Alternatively you can use a JSON file to specify the input. For example:

```
aws waf-regional update-ip-set \  
  --ip-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90 \  
  --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 \  
  --updates file://change.json
```

Content of the `change.json`

```
[  
  {  
    "Action": "INSERT",  
    "IPSetDescriptor":  
    {  
      "Type": "IPV4",  
      "Value": "12.34.56.78/16"  
    }  
  },  
]
```

```

    {
      "Action": "DELETE",
      "IPSetDescriptor":
      {
        "Type": "IPV6",
        "Value": "1111:0000:0000:0000:0000:0000:0000:0111/128"
      }
    }
  ]

```

For more information, see [Working with IP Match Conditions](#) in the *AWS WAF Developer Guide*.

- For API details, see [UpdateIpSet](#) in *AWS CLI Command Reference*.

update-rule

The following code example shows how to use `update-rule`.

AWS CLI

To update a rule

The following `update-rule` command deletes a Predicate object in a rule. Because the updates value has embedded double quotes, you must surround the entire value with single quotes.

```

aws waf-regional update-rule \
  --rule-id a123fae4-b567-8e90-1234-5ab67ac8ca90 \
  --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 \
  --updates
  'Action="DELETE",Predicate={Negated=false,Type="ByteMatch",DataId="MyByteMatchSetID"}'

```

For more information, see [Working with Rules](#) in the *AWS WAF Developer Guide*.

- For API details, see [UpdateRule](#) in *AWS CLI Command Reference*.

update-size-constraint-set

The following code example shows how to use `update-size-constraint-set`.

AWS CLI

To update a size constraint set

The following `update-size-constraint-set` command deletes a `SizeConstraint`` object (filters) in a size constraint set. Because the updates value contains embedded double quotes, you must surround the entire value with single quotes.

```
aws waf-regional update-size-constraint-set \  
  --size-constraint-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90 \  
  --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 \  
  --updates  
'Action="DELETE",SizeConstraint={FieldToMatch={Type="QUERY_STRING"},TextTransformation="NON
```

For more information, see [Working with Size Constraint Conditions](#) in the *AWS WAF Developer Guide*.

- For API details, see [UpdateSizeConstraintSet](#) in *AWS CLI Command Reference*.

update-sql-injection-match-set

The following code example shows how to use `update-sql-injection-match-set`.

AWS CLI

To update a SQL Injection Match Set

The following `update-sql-injection-match-set` command deletes a `SqlInjectionMatchTuple` object (filters) in a SQL injection match set. Because the updates value contains embedded double quotes, you must surround the entire value in single quotes. :

```
aws waf-regional update-sql-injection-match-set --sql-injection-match-set-id a123fae4-  
b567-8e90-1234-5ab67ac8ca90 --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 --  
updates  
'Action="DELETE",SqlInjectionMatchTuple={FieldToMatch={Type="QUERY_STRING"},TextTransformation
```

For more information, see [Working with SQL Injection Match Conditions](#) in the *AWS WAF Developer Guide*.

- For API details, see [UpdateSqlInjectionMatchSet](#) in *AWS CLI Command Reference*.

update-web-acl

The following code example shows how to use `update-web-acl`.

AWS CLI

To update a web ACL

The following `update-web-acl` command deletes an `ActivatedRule` object in a `WebACL`. Because the `updates` value contains embedded double quotes, you must surround the entire value in single quotes.

```
aws waf-regional update-web-acl \  
  --web-acl-id a123fae4-b567-8e90-1234-5ab67ac8ca90 \  
  --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 \  
  --updates Action="DELETE",ActivatedRule='{Priority=1,RuleId="WAFRule-1-  
Example",Action={Type="ALLOW"},Type="ALLOW"}'
```

For more information, see [Working with Web ACLs](#) in the *AWS WAF Developer Guide*.

- For API details, see [UpdateWebAcl](#) in *AWS CLI Command Reference*.

update-xss-match-set

The following code example shows how to use `update-xss-match-set`.

AWS CLI

To update an XSSMatchSet

The following `update-xss-match-set` command deletes an `XssMatchTuple` object (filters) in an `XssMatchSet`. Because the `updates` value contains embedded double quotes, you must surround the entire value with single quotes.

```
aws waf-regional update-xss-match-set \  
  --xss-match-set-id a123fae4-b567-8e90-1234-5ab67ac8ca90 \  
  --change-token 12cs345-67cd-890b-1cd2-c3a4567d89f1 \  
  --updates  
'Action="DELETE",XssMatchTuple={FieldToMatch={Type="QUERY_STRING"},TextTransformation="URL_
```

For more information, see [Working with Cross-site Scripting Match Conditions](#) in the *AWS WAF Developer Guide*.

- For API details, see [UpdateXssMatchSet](#) in *AWS CLI Command Reference*.

AWS WAFV2 examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with AWS WAFV2.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-web-acl

The following code example shows how to use `associate-web-acl`.

AWS CLI

To associate a web ACL with a regional AWS resource

The following `associate-web-acl` example associates the specified web ACL with an Application Load Balancer.

```
aws wafv2 associate-web-acl \  
  --web-acl-arn arn:aws:wafv2:us-west-2:123456789012:regional/webacl/test-cli/  
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \  
  --resource-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/  
app/waf-cli-alb/1ea17125f8b25a2a \  
  --region us-west-2
```

This command produces no output.

For more information, see [Associating or Disassociating a Web ACL with an AWS Resource](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [AssociateWebAcl](#) in *AWS CLI Command Reference*.

check-capacity

The following code example shows how to use check-capacity.

AWS CLI

To obtain the capacity used by a set of rules

The following check-capacity retrieves the capacity requirements for a rule set that contains a rate-based rule statement, and an AND rule statement that contains nested rules.

```
aws wafv2 check-capacity \  
  --scope REGIONAL \  
  --rules file://waf-rule-list.json \  
  --region us-west-2
```

Contents of file://waf-rule-list.json:

```
[  
  {  
    "Name":"basic-rule",  
    "Priority":0,  
    "Statement":{  
      "AndStatement":{  
        "Statements":[  
          {  
            "ByteMatchStatement":{  
              "SearchString":"example.com",  
              "FieldToMatch":{  
                "SingleHeader":{  
                  "Name":"host"  
                }  
              }  
            },  
            "TextTransformations":[  
              {  
                "Priority":0,  
                "Type":"LOWERCASE"  
              }  
            ],  
            "PositionalConstraint":"EXACTLY"  
          }  
        ]  
      }  
    }  
  ]  
]
```

```
    },
    {
      "GeoMatchStatement":{
        "CountryCodes":[
          "US",
          "IN"
        ]
      }
    ]
  },
  "Action":{
    "Allow":{
      }
    },
  "VisibilityConfig":{
    "SampledRequestsEnabled":true,
    "CloudWatchMetricsEnabled":true,
    "MetricName":"basic-rule"
  }
},
{
  "Name":"rate-rule",
  "Priority":1,
  "Statement":{
    "RateBasedStatement":{
      "Limit":1000,
      "AggregateKeyType":"IP"
    }
  },
  "Action":{
    "Block":{
      }
    },
  "VisibilityConfig":{
    "SampledRequestsEnabled":true,
    "CloudWatchMetricsEnabled":true,
    "MetricName":"rate-rule"
  }
}
```

```
]
```

Output:

```
{
  "Capacity":15
}
```

For more information, see [AWS WAF Web ACL Capacity Units \(WCU\)](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [CheckCapacity](#) in *AWS CLI Command Reference*.

create-ip-set

The following code example shows how to use `create-ip-set`.

AWS CLI**To create an IP set for use in your web ACLs and rule groups**

The following `create-ip-set` command creates an IP set with a single address range specification.

```
aws wafv2 create-ip-set \
  --name testip \
  --scope REGIONAL \
  --ip-address-version IPV4 \
  --addresses 198.51.100.0/16
```

Output:

```
{
  "Summary":{
    "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/ipset/testip/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "Description":"",
    "Name":"testip",
    "LockToken":"447e55ac-0000-0000-0000-86b67c17f8b5",
    "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
  }
}
```

```
}
```

For more information, see [IP Sets and Regex Pattern Sets](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [CreateIpSet](#) in *AWS CLI Command Reference*.

create-regex-pattern-set

The following code example shows how to use `create-regex-pattern-set`.

AWS CLI

To create a regex pattern set for use in your web ACLs and rule groups

The following `create-regex-pattern-set` command creates a regex pattern set with two regex patterns specified.

```
aws wafv2 create-regex-pattern-set \  
  --name regexPatterSet01 \  
  --scope REGIONAL \  
  --description 'Test web-acl' \  
  --regular-expression-list '[{"RegexString": "/[0-9]*/"}, {"RegexString": "/[a-z]*/"}]'
```

Output:

```
{  
  "Summary": {  
    "ARN": "arn:aws:wafv2:us-west-2:123456789012:regional/regexpatternset/  
regexPatterSet01/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
    "Description": "Test web-acl",  
    "Name": "regexPatterSet01",  
    "LockToken": "0bc01e21-03c9-4b98-9433-6229cbf1ef1c",  
    "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"  
  }  
}
```

For more information, see [IP Sets and Regex Pattern Sets](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [CreateRegexPatternSet](#) in *AWS CLI Command Reference*.

create-rule-group

The following code example shows how to use `create-rule-group`.

AWS CLI

To create a custom rule group for use in your web ACLs

The following `create-rule-group` command creates a custom rule group for regional use. The rule statements for the group are provided in a JSON-formatted file.

```
aws wafv2 create-rule-group \  
  --name "TestRuleGroup" \  
  --scope REGIONAL \  
  --capacity 250 \  
  --rules file://waf-rule.json \  
  --visibility-config  
  SampledRequestsEnabled=true,CloudWatchMetricsEnabled=true,MetricName=TestRuleGroupMetrics  
  \  
  --region us-west-2
```

Contents of `file://waf-rule.json`:

```
[  
  {  
    "Name":"basic-rule",  
    "Priority":0,  
    "Statement":{  
      "AndStatement":{  
        "Statements":[  
          {  
            "ByteMatchStatement":{  
              "SearchString":"example.com",  
              "FieldToMatch":{  
                "SingleHeader":{  
                  "Name":"host"  
                }  
              },  
              "TextTransformations":[  
                {  
                  "Priority":0,  
                  "Type":"LOWERCASE"  
                }  
              ]  
            }  
          ]  
        }  
      }  
    }  
  ]  
]
```

```

        ],
        "PositionalConstraint":"EXACTLY"
    }
},
{
    "GeoMatchStatement":{
        "CountryCodes":[
            "US",
            "IN"
        ]
    }
}
]
},
"Action":{
    "Allow":{

    }
},
"VisibilityConfig":{
    "SampledRequestsEnabled":true,
    "CloudWatchMetricsEnabled":true,
    "MetricName":"basic-rule"
}
}
]

```

Output:

```

{
    "Summary":{
        "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/rulegroup/
TestRuleGroup/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "Description":"",
        "Name":"TestRuleGroup",
        "LockToken":"7b3bcec2-374e-4c5a-b2b9-563bf47249f0",
        "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    }
}

```

For more information, see [Managing Your Own Rule Groups](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [CreateRuleGroup](#) in *AWS CLI Command Reference*.

create-web-acl

The following code example shows how to use `create-web-acl`.

AWS CLI

To create a web ACL

The following `create-web-acl` command creates a web ACL for regional use. The rule statements for the web ACL are provided in a JSON-formatted file.

```
aws wafv2 create-web-acl \
  --name TestWebAcl \
  --scope REGIONAL \
  --default-action Allow={} \
  --visibility-config
  SampledRequestsEnabled=true,CloudWatchMetricsEnabled=true,MetricName=TestWebAclMetrics
  \
  --rules file://waf-rule.json \
  --region us-west-2
```

Contents of `file://waf-rule.json`:

```
[
  {
    "Name":"basic-rule",
    "Priority":0,
    "Statement":{
      "AndStatement":{
        "Statements":[
          {
            "ByteMatchStatement":{
              "SearchString":"example.com",
              "FieldToMatch":{
                "SingleHeader":{
                  "Name":"host"
                }
              },
            },
            "TextTransformations":[
              {
```



```

        "Priority":0,
        "Type":"LOWERCASE"
      }
    ],
    "PositionalConstraint":"EXACTLY"
  }
},
{
  "GeoMatchStatement":{
    "CountryCodes":[
      "US",
      "IN"
    ]
  }
}
]
},
"Action":{
  "Allow":{
  }
},
"VisibilityConfig":{
  "SampledRequestsEnabled":true,
  "CloudWatchMetricsEnabled":true,
  "MetricName":"basic-rule"
}
}
]

```

Output:

```

{
  "Summary":{
    "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/webacl/TestWebAcl/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "Description":"","
    "Name":"TestWebAcl",
    "LockToken":"2294b3a1-eb60-4aa0-a86f-a3ae04329de9",
    "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
  }
}

```

For more information, see [Managing and Using a Web Access Control List \(Web ACL\)](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [CreateWebAcl](#) in *AWS CLI Command Reference*.

delete-ip-set

The following code example shows how to use `delete-ip-set`.

AWS CLI

To delete an IP set

The following `delete-ip-set` deletes the specified IP set. This call requires an ID, which you can obtain from the call, `list-ip-sets`, and a lock token, which you can obtain from the calls, `list-ip-sets` and `get-ip-set`.

```
aws wafv2 delete-ip-set \  
  --name test1 \  
  --scope REGIONAL \  
  --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \  
  --lock-token 46851772-db6f-459d-9385-49428812e357
```

This command produces no output.

For more information, see [IP Sets and Regex Pattern Sets](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [DeleteIpSet](#) in *AWS CLI Command Reference*.

delete-logging-configuration

The following code example shows how to use `delete-logging-configuration`.

AWS CLI

To disable logging for a web ACL

The following `delete-logging-configuration` removes any logging configuration from the specified web ACL.

```
aws wafv2 delete-logging-configuration \  
  --web-ACL-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

```
--resource-arn arn:aws:wafv2:us-west-2:123456789012:regional/webacl/test/
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222
```

This command produces no output.

For more information, see [Logging Web ACL Traffic Information](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [DeleteLoggingConfiguration](#) in *AWS CLI Command Reference*.

delete-regex-pattern-set

The following code example shows how to use `delete-regex-pattern-set`.

AWS CLI

To delete a regex pattern set

The following `delete-regex-pattern-set` updates the settings for the specified regex pattern set. This call requires an ID, which you can obtain from the call, `list-regex-pattern-sets`, and a lock token, which you can obtain from the call `list-regex-pattern-sets` or the call `get-regex-pattern-set`.

```
aws wafv2 delete-regex-pattern-set \
  --name regexPatterSet01 \
  --scope REGIONAL \
  --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
  --lock-token 0bc01e21-03c9-4b98-9433-6229cbf1ef1c
```

This command produces no output.

For more information, see [IP Sets and Regex Pattern Sets](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [DeleteRegexPatternSet](#) in *AWS CLI Command Reference*.

delete-rule-group

The following code example shows how to use `delete-rule-group`.

AWS CLI

To delete a custom rule group

The following `delete-rule-group` deletes the specified custom rule group. This call requires an ID, which you can obtain from the call `list-rule-groups`, and a lock token, which you can obtain from the call `list-rule-groups` or the call `get-rule-group`.

```
aws wafv2 delete-rule-group \  
  --name TestRuleGroup \  
  --scope REGIONAL \  
  --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \  
  --lock-token 7b3bcec2-0000-0000-0000-563bf47249f0
```

This command produces no output.

For more information, see [Managing Your Own Rule Groups](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [DeleteRuleGroup](#) in *AWS CLI Command Reference*.

delete-web-acl

The following code example shows how to use `delete-web-acl`.

AWS CLI

To delete a web ACL

The following `delete-web-acl` deletes the specified web ACL from your account. A web ACL can only be deleted when it's not associated with any resources. This call requires an ID, which you can obtain from the call `list-web-acls`, and a lock token, which you can obtain from the call `list-web-acls` or the call `get-web-acl`.

```
aws wafv2 delete-web-acl \  
  --name test \  
  --scope REGIONAL \  
  --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \  
  --lock-token ebab4ed2-155e-4c9a-9efb-e4c45665b1f5
```

This command produces no output.

For more information, see [Managing and Using a Web Access Control List \(Web ACL\)](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [DeleteWebAcl](#) in *AWS CLI Command Reference*.

describe-managed-rule-group

The following code example shows how to use describe-managed-rule-group.

AWS CLI

To retrieve the description for a managed rule group

The following describe-managed-rule-group retrieves the description for an AWS managed rule group.

```
aws wafv2 describe-managed-rule-group \  
  --vendor-name AWS \  
  --name AWSManagedRulesCommonRuleSet \  
  --scope REGIONAL
```

Output:

```
{  
  "Capacity": 700,  
  "Rules": [  
    {  
      "Name": "NoUserAgent_HEADER",  
      "Action": {  
        "Block": {}  
      }  
    },  
    {  
      "Name": "UserAgent_BadBots_HEADER",  
      "Action": {  
        "Block": {}  
      }  
    },  
    {  
      "Name": "SizeRestrictions_QUERYSTRING",  
      "Action": {  
        "Block": {}  
      }  
    },  
    {  
      "Name": "SizeRestrictions_Cookie_HEADER",  
      "Action": {  
        "Block": {}  
      }  
    }  
  ]  
}
```

```
    }
  },
  {
    "Name": "SizeRestrictions_BODY",
    "Action": {
      "Block": {}
    }
  },
  {
    "Name": "SizeRestrictions_URIPATH",
    "Action": {
      "Block": {}
    }
  },
  {
    "Name": "EC2MetaDataSSRF_BODY",
    "Action": {
      "Block": {}
    }
  },
  {
    "Name": "EC2MetaDataSSRF_COOKIE",
    "Action": {
      "Block": {}
    }
  },
  {
    "Name": "EC2MetaDataSSRF_URIPATH",
    "Action": {
      "Block": {}
    }
  },
  {
    "Name": "EC2MetaDataSSRF_QUERYARGUMENTS",
    "Action": {
      "Block": {}
    }
  },
  {
    "Name": "GenericLFI_QUERYARGUMENTS",
    "Action": {
      "Block": {}
    }
  },
},
```

```
{
  }
  "Name": "GenericLFI_URIPATH",
  "Action": {
    "Block": {}
  }
},
{
  "Name": "GenericLFI_BODY",
  "Action": {
    "Block": {}
  }
},
{
  "Name": "RestrictedExtensions_URIPATH",
  "Action": {
    "Block": {}
  }
},
{
  "Name": "RestrictedExtensions_QUERYARGUMENTS",
  "Action": {
    "Block": {}
  }
},
{
  "Name": "GenericRFI_QUERYARGUMENTS",
  "Action": {
    "Block": {}
  }
},
{
  "Name": "GenericRFI_BODY",
  "Action": {
    "Block": {}
  }
},
{
  "Name": "GenericRFI_URIPATH",
  "Action": {
    "Block": {}
  }
},
{
```

```

        "Name": "CrossSiteScripting_COOKIE",
        "Action": {
            "Block": {}
        }
    },
    {
        "Name": "CrossSiteScripting_QUERYARGUMENTS",
        "Action": {
            "Block": {}
        }
    },
    {
        "Name": "CrossSiteScripting_BODY",
        "Action": {
            "Block": {}
        }
    },
    {
        "Name": "CrossSiteScripting_URI_PATH",
        "Action": {
            "Block": {}
        }
    }
]
}

```

For more information, see [Managed Rule Groups](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [DescribeManagedRuleGroup](#) in *AWS CLI Command Reference*.

disassociate-web-acl

The following code example shows how to use `disassociate-web-acl`.

AWS CLI

To disassociate a web ACL from a regional AWS resource

The following `disassociate-web-acl` example removes any existing web ACL association from the specified Application Load Balancer.

```
aws wafv2 disassociate-web-acl \
```



```
--resource-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/
app/waf-cli-alb/1ea17125f8b25a2a \
--region us-west-2
```

This command produces no output.

For more information, see [Associating or Disassociating a Web ACL with an AWS Resource](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [DisassociateWebAcl](#) in *AWS CLI Command Reference*.

get-ip-set

The following code example shows how to use `get-ip-set`.

AWS CLI

To retrieve a specific IP set

The following `get-ip-set` retrieves the IP set with the specified name, scope, and ID. You can get the ID for an IP set from the commands `create-ip-set` and `list-ip-sets`.

```
aws wafv2 get-ip-set \
  --name testip \
  --scope REGIONAL \
  --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{
  "IPSet":{
    "Description":"","
    "Name":"testip",
    "IPAddressVersion":"IPV4",
    "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE1111",
    "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/ipset/testip/
a1b2c3d4-5678-90ab-cdef-EXAMPLE1111",
    "Addresses":[
      "192.0.2.0/16"
    ]
  },
  "LockToken":"447e55ac-2396-4c6d-b9f9-86b67c17f8b5"
```

```
}
```

For more information, see [IP Sets and Regexp Pattern Sets](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [GetIpSet](#) in *AWS CLI Command Reference*.

get-logging-configuration

The following code example shows how to use `get-logging-configuration`.

AWS CLI

To retrieve the logging configurations for a web ACL

The following `get-logging-configuration` retrieves the logging configuration for the specified web ACL.

```
aws wafv2 get-logging-configuration \
  --resource-arn arn:aws:wafv2:us-west-2:123456789012:regional/webacl/test/
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222 \
  --region us-west-2
```

Output:

```
{
  "LoggingConfiguration":{
    "ResourceArn":"arn:aws:wafv2:us-west-2:123456789012:regional/webacl/test/
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "RedactedFields":[
      {
        "Method":{

        }
      }
    ],
    "LogDestinationConfigs":[
      "arn:aws:firehose:us-west-2:123456789012:deliverystream/aws-waf-logs-
custom-transformation"
    ]
  }
}
```

For more information, see [Logging Web ACL Traffic Information](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [GetLoggingConfiguration](#) in *AWS CLI Command Reference*.

get-rate-based-statement-managed-keys

The following code example shows how to use `get-rate-based-statement-managed-keys`.

AWS CLI

To retrieve a list of IP addresses that are blocked by a rate-based rule

The following `get-rate-based-statement-managed-keys` retrieves the IP addresses currently blocked by a rate-based rule that's being used for a regional application.

```
aws wafv2 get-rate-based-statement-managed-keys \
  --scope REGIONAL \
  --web-acl-name testwebacl2 \
  --web-acl-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
  --rule-name ratebasedtest
```

Output:

```
{
  "ManagedKeysIPV4":{
    "IPAddressVersion":"IPV4",
    "Addresses":[
      "198.51.100.0/32"
    ]
  },
  "ManagedKeysIPV6":{
    "IPAddressVersion":"IPV6",
    "Addresses":[
    ]
  }
}
```

For more information, see [Rate-Based Rule Statement](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [GetRateBasedStatementManagedKeys](#) in *AWS CLI Command Reference*.

get-regex-pattern-set

The following code example shows how to use `get-regex-pattern-set`.

AWS CLI

To retrieve a specific regex pattern set

The following `get-regex-pattern-set` retrieves the regex pattern set with the specified name, scope, region, and ID. You can get the ID for a regex pattern set from the commands `create-regex-pattern-set` and `list-regex-pattern-sets`.

```
aws wafv2 get-regex-pattern-set \
  --name regexPatterSet01 \
  --scope REGIONAL \
  --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
  --region us-west-2
```

Output:

```
{
  "RegexPatternSet":{
    "Description":"Test web-acl",
    "RegularExpressionList":[
      {
        "RegexString":"/[0-9]*/"
      },
      {
        "RegexString":"/[a-z]*/"
      }
    ],
    "Name":"regexPatterSet01",
    "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/regexpatternset/
regexPatterSet01/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
  },
  "LockToken":"c8abf33f-b6fc-46ae-846e-42f994d57b29"
}
```

For more information, see [IP Sets and Regex Pattern Sets](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [GetRegexPatternSet](#) in *AWS CLI Command Reference*.

get-rule-group

The following code example shows how to use `get-rule-group`.

AWS CLI

To retrieve a specific custom rule group

The following `get-rule-group` retrieves the custom rule group with the specified name, scope, and ID. You can get the ID for a rule group from the commands `create-rule-group` and `list-rule-groups`.

```
aws wafv2 get-rule-group \
  --name ff \
  --scope REGIONAL \
  --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{
  "RuleGroup":{
    "Capacity":1,
    "Description":"",
    "Rules":[
      {
        "Priority":0,
        "Action":{
          "Block":{

          }
        },
        "VisibilityConfig":{
          "SampledRequestsEnabled":true,
          "CloudWatchMetricsEnabled":true,
          "MetricName":"jj"
        },
        "Name":"jj",
        "Statement":{
          "SizeConstraintStatement":{
            "ComparisonOperator":"LE",
            "TextTransformations":[
              {
                "Priority":0,
```

```

        "Type": "NONE"
      }
    ],
    "FieldToMatch": {
      "UriPath": {
        "Size": 7
      }
    }
  },
  "VisibilityConfig": {
    "SampledRequestsEnabled": true,
    "CloudWatchMetricsEnabled": true,
    "MetricName": "ff"
  },
  "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "ARN": "arn:aws:wafv2:us-west-2:123456789012:regional/rulegroup/ff/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "Name": "ff"
},
"LockToken": "485458c9-1830-4234-af31-ec4d52ced1b3"
}

```

For more information, see [Managing Your Own Rule Groups](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [GetRuleGroup](#) in *AWS CLI Command Reference*.

get-sampled-requests

The following code example shows how to use `get-sampled-requests`.

AWS CLI

To retrieve a sample of web requests for a web ACL

The following `get-sampled-requests` retrieves the sampled web requests for the specified web ACL, rule metric, and time frame.

```
aws wafv2 get-sampled-requests \
```

```
--web-acl-arn arn:aws:wafv2:us-west-2:123456789012:regional/webacl/test-cli/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
--rule-metric-name AWS-AWSManagedRulesSQLiRuleSet \
--scope=REGIONAL \
--time-window StartTime=2020-02-12T20:00Z,EndTime=2020-02-12T21:10Z \
--max-items 100
```

Output:

```
{
  "TimeWindow": {
    "EndTime": 1581541800.0,
    "StartTime": 1581537600.0
  },
  "SampledRequests": [
    {
      "Action": "BLOCK",
      "Timestamp": 1581541799.564,
      "RuleNameWithinRuleGroup": "AWS#AWSManagedRulesSQLiRuleSet#SQLi_BODY",
      "Request": {
        "Country": "US",
        "URI": "/",
        "Headers": [
          {
            "Name": "Host",
            "Value": "alb-test-1EXAMPLE1.us-east-1.elb.amazonaws.com"
          },
          {
            "Name": "Content-Length",
            "Value": "7456"
          },
          {
            "Name": "User-Agent",
            "Value": "curl/7.53.1"
          },
          {
            "Name": "Accept",
            "Value": "/"
          },
          {
            "Name": "Content-Type",
            "Value": "application/x-www-form-urlencoded"
          }
        ]
      }
    }
  ]
}
```

```
    ],
    "ClientIP": "198.51.100.08",
    "Method": "POST",
    "HTTPVersion": "HTTP/1.1"
  },
  "Weight": 1
},
{
  "Action": "BLOCK",
  "Timestamp": 1581541799.988,
  "RuleNameWithinRuleGroup": "AWS#AWSManagedRulesSQLiRuleSet#SQLi_BODY",
  "Request": {
    "Country": "US",
    "URI": "/",
    "Headers": [
      {
        "Name": "Host",
        "Value": "alb-test-1EXAMPLE1.us-east-1.elb.amazonaws.com"
      },
      {
        "Name": "Content-Length",
        "Value": "7456"
      },
      {
        "Name": "User-Agent",
        "Value": "curl/7.53.1"
      },
      {
        "Name": "Accept",
        "Value": "/"
      },
      {
        "Name": "Content-Type",
        "Value": "application/x-www-form-urlencoded"
      }
    ]
  },
  "ClientIP": "198.51.100.08",
  "Method": "POST",
  "HTTPVersion": "HTTP/1.1"
},
  "Weight": 3
},
{
  "Action": "BLOCK",
```



```
"Timestamp": 1581541799.846,
"RuleNameWithinRuleGroup": "AWS#AWSManagedRulesSQLiRuleSet#SQLi_BODY",
"Request": {
  "Country": "US",
  "URI": "/",
  "Headers": [
    {
      "Name": "Host",
      "Value": "alb-test-1EXAMPLE1.us-east-1.elb.amazonaws.com"
    },
    {
      "Name": "Content-Length",
      "Value": "7456"
    },
    {
      "Name": "User-Agent",
      "Value": "curl/7.53.1"
    },
    {
      "Name": "Accept",
      "Value": "/"
    },
    {
      "Name": "Content-Type",
      "Value": "application/x-www-form-urlencoded"
    }
  ],
  "ClientIP": "198.51.100.08",
  "Method": "POST",
  "HTTPVersion": "HTTP/1.1"
},
"Weight": 1
},
{
  "Action": "BLOCK",
  "Timestamp": 1581541799.4,
  "RuleNameWithinRuleGroup": "AWS#AWSManagedRulesSQLiRuleSet#SQLi_BODY",
  "Request": {
    "Country": "US",
    "URI": "/",
    "Headers": [
      {
        "Name": "Host",
        "Value": "alb-test-1EXAMPLE1.us-east-1.elb.amazonaws.com"
```

```

        },
        {
            "Name": "Content-Length",
            "Value": "7456"
        },
        {
            "Name": "User-Agent",
            "Value": "curl/7.53.1"
        },
        {
            "Name": "Accept",
            "Value": "/"
        },
        {
            "Name": "Content-Type",
            "Value": "application/x-www-form-urlencoded"
        }
    ],
    "ClientIP": "198.51.100.08",
    "Method": "POST",
    "HTTPVersion": "HTTP/1.1"
},
"Weight": 1
}
],
"PopulationSize": 4
}

```

For more information, see [Viewing a Sample of Web Requests](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [GetSampledRequests](#) in *AWS CLI Command Reference*.

get-web-acl-for-resource

The following code example shows how to use `get-web-acl-for-resource`.

AWS CLI

To retrieve the web ACL that's associated with an AWS resource

The following `get-web-acl-for-resource` retrieves the JSON for the web ACL that's associated with the specified resource.

```
aws wafv2 get-web-acl-for-resource \  
  --resource-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/  
  app/waf-cli-alb/1ea17125f8b25a2a
```

Output:

```
{  
  "WebACL":{  
    "Capacity":3,  
    "Description":"","  
    "Rules":[  
      {  
        "Priority":1,  
        "Action":{  
          "Block":{  
  
          }  
        },  
        "VisibilityConfig":{  
          "SampledRequestsEnabled":true,  
          "CloudWatchMetricsEnabled":true,  
          "MetricName":"testrule01"  
        },  
        "Name":"testrule01",  
        "Statement":{  
          "AndStatement":{  
            "Statements":[  
              {  
                "ByteMatchStatement":{  
                  "PositionalConstraint":"EXACTLY",  
                  "TextTransformations":[  
                    {  
                      "Priority":0,  
                      "Type":"NONE"  
                    }  
                  ],  
                  "SearchString":"dGVzdHN0cm1uZw==",  
                  "FieldToMatch":{  
                    "UriPath":{  
  
                    }  
                  }  
                }  
              ]  
            }  
          }  
        }  
      ]  
    }  
  }  
}
```

```

        },
        {
            "SizeConstraintStatement":{
                "ComparisonOperator":"EQ",
                "TextTransformations":[
                    {
                        "Priority":0,
                        "Type":"NONE"
                    }
                ],
                "FieldToMatch":{
                    "QueryString":{

                    }
                },
                "Size":0
            }
        }
    ]
}
},
"VisibilityConfig":{
    "SampledRequestsEnabled":true,
    "CloudWatchMetricsEnabled":true,
    "MetricName":"test01"
},
"DefaultAction":{
    "Allow":{

    }
},
"Id":"9a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 ",
"ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/webacl/test01/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 ",
"Name":"test01"
}
}

```

For more information, see [Associating or Disassociating a Web ACL with an AWS Resource](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [GetWebAclForResource](#) in *AWS CLI Command Reference*.

get-web-acl

The following code example shows how to use `get-web-acl`.

AWS CLI

To retrieve a web ACL

The following `get-web-acl` retrieves the web ACL with the specified name, scope, and ID. You can get the ID for a web ACL from the commands `create-web-acl` and `list-web-acls`.

```
aws wafv2 get-web-acl \  
  --name test01 \  
  --scope REGIONAL \  
  --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Output:

```
{  
  "WebACL":{  
    "Capacity":3,  
    "Description":"","  
    "Rules":[  
      {  
        "Priority":1,  
        "Action":{  
          "Block":{  
  
          }  
        },  
        "VisibilityConfig":{  
          "SampledRequestsEnabled":true,  
          "CloudWatchMetricsEnabled":true,  
          "MetricName":"testrule01"  
        },  
        "Name":"testrule01",  
        "Statement":{  
          "AndStatement":{  
            "Statements":[  
              {  
                "ByteMatchStatement":{  
                  "PositionalConstraint":"EXACTLY",  
                  "TextTransformations":[
```

```

        {
            "Priority":0,
            "Type":"NONE"
        }
    ],
    "SearchString":"dGVzdHN0cmlyZw==",
    "FieldToMatch":{
        "UriPath":{

        }
    }
}
},
{
    "SizeConstraintStatement":{
        "ComparisonOperator":"EQ",
        "TextTransformations":[
            {
                "Priority":0,
                "Type":"NONE"
            }
        ],
        "FieldToMatch":{
            "QueryString":{

            }
        },
        "Size":0
    }
}
]
}
}
},
"VisibilityConfig":{
    "SampledRequestsEnabled":true,
    "CloudWatchMetricsEnabled":true,
    "MetricName":"test01"
},
"DefaultAction":{
    "Allow":{

    }
}

```

```

    },
    "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "ARN": "arn:aws:wafv2:us-west-2:123456789012:regional/webacl/test01/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "Name": "test01"
  },
  "LockToken": "e3db7e2c-d58b-4ee6-8346-6aec5511c6fb"
}

```

For more information, see [Managing and Using a Web Access Control List \(Web ACL\)](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [GetWebAcl](#) in *AWS CLI Command Reference*.

list-available-managed-rule-groups

The following code example shows how to use `list-available-managed-rule-groups`.

AWS CLI

To retrieve the managed rule groups

The following `list-available-managed-rule-groups` returns the list of all managed rule groups that are currently available for use in your web ACLs.

```
aws wafv2 list-available-managed-rule-groups \
  --scope REGIONAL
```

Output:

```

{
  "ManagedRuleGroups": [
    {
      "VendorName": "AWS",
      "Name": "AWSManagedRulesCommonRuleSet",
      "Description": "Contains rules that are generally applicable to web applications. This provides protection against exploitation of a wide range of vulnerabilities, including those described in OWASP publications and common Common Vulnerabilities and Exposures (CVE).",
    },
    {
      "VendorName": "AWS",
      "Name": "AWSManagedRulesAdminProtectionRuleSet",
    }
  ]
}

```

```
    "Description": "Contains rules that allow you to block external access
to exposed admin pages. This may be useful if you are running third-party software
or would like to reduce the risk of a malicious actor gaining administrative access
to your application."
  },
  {
    "VendorName": "AWS",
    "Name": "AWSManagedRulesKnownBadInputsRuleSet",
    "Description": "Contains rules that allow you to block request patterns
that are known to be invalid and are associated with exploitation or discovery of
vulnerabilities. This can help reduce the risk of a malicious actor discovering a
vulnerable application."
  },
  {
    "VendorName": "AWS",
    "Name": "AWSManagedRulesSQLiRuleSet",
    "Description": "Contains rules that allow you to block request patterns
associated with exploitation of SQL databases, like SQL injection attacks. This can
help prevent remote injection of unauthorized queries."
  },
  {
    "VendorName": "AWS",
    "Name": "AWSManagedRulesLinuxRuleSet",
    "Description": "Contains rules that block request patterns associated
with exploitation of vulnerabilities specific to Linux, including LFI attacks. This
can help prevent attacks that expose file contents or execute code for which the
attacker should not have had access."
  },
  {
    "VendorName": "AWS",
    "Name": "AWSManagedRulesUnixRuleSet",
    "Description": "Contains rules that block request patterns associated
with exploiting vulnerabilities specific to POSIX/POSIX-like OS, including LFI
attacks. This can help prevent attacks that expose file contents or execute code
for which access should not been allowed."
  },
  {
    "VendorName": "AWS",
    "Name": "AWSManagedRulesWindowsRuleSet",
    "Description": "Contains rules that block request patterns associated
with exploiting vulnerabilities specific to Windows, (e.g., PowerShell commands).
This can help prevent exploits that allow attacker to run unauthorized commands or
execute malicious code."
  },
  },
```



```

    {
      "VendorName": "AWS",
      "Name": "AWSManagedRulesPHPRuleSet",
      "Description": "Contains rules that block request patterns associated
with exploiting vulnerabilities specific to the use of the PHP, including injection
of unsafe PHP functions. This can help prevent exploits that allow an attacker to
remotely execute code or commands."
    },
    {
      "VendorName": "AWS",
      "Name": "AWSManagedRulesWordPressRuleSet",
      "Description": "The WordPress Applications group contains rules that
block request patterns associated with the exploitation of vulnerabilities specific
to WordPress sites."
    },
    {
      "VendorName": "AWS",
      "Name": "AWSManagedRulesAmazonIpReputationList",
      "Description": "This group contains rules that are based on Amazon
threat intelligence. This is useful if you would like to block sources associated
with bots or other threats."
    }
  ]
}

```

For more information, see [Managed Rule Groups](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [ListAvailableManagedRuleGroups](#) in *AWS CLI Command Reference*.

list-ip-sets

The following code example shows how to use `list-ip-sets`.

AWS CLI

To retrieve a list of IP sets

The following `list-ip-sets` retrieves all IP sets for the account that have regional scope.

```
aws wafv2 list-ip-sets \
  --scope REGIONAL
```

Output:

```
{
  "IPSets":[
    {
      "ARN":"arn:aws:wafv2:us-west-2:123456789012:regional/ipset/testip/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "Description":"",
      "Name":"testip",
      "LockToken":"0674c84b-0304-47fe-8728-c6bff46af8fc",
      "Id":"a1b2c3d4-5678-90ab-cdef-EXAMPLE11111  "
    }
  ],
  "NextMarker":"testip"
}
```

For more information, see [IP Sets and Regex Pattern Sets](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [ListIpSets](#) in *AWS CLI Command Reference*.

list-logging-configurations

The following code example shows how to use `list-logging-configurations`.

AWS CLI**To retrieve a list of all logging configurations for a region**

The following `list-logging-configurations` retrieves the all logging configurations for web ACLs that are scoped for regional use in the `us-west-2` region.

```
aws wafv2 list-logging-configurations \
  --scope REGIONAL \
  --region us-west-2
```

Output:

```
{
  "LoggingConfigurations":[
    {
```

```

        "ResourceArn": "arn:aws:wafv2:us-west-2:123456789012:regional/webacl/
test-2/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "RedactedFields": [
            {
                "QueryString": {
                    }
            }
        ],
        "LogDestinationConfigs": [
            "arn:aws:firehose:us-west-2:123456789012:deliverystream/aws-waf-
logs-test"
        ]
    },
    {
        "ResourceArn": "arn:aws:wafv2:us-west-2:123456789012:regional/webacl/
test/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "RedactedFields": [
            {
                "Method": {
                    }
            }
        ],
        "LogDestinationConfigs": [
            "arn:aws:firehose:us-west-2:123456789012:deliverystream/aws-waf-
logs-custom-transformation"
        ]
    }
]
}

```

For more information, see [Logging Web ACL Traffic Information](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [ListLoggingConfigurations](#) in *AWS CLI Command Reference*.

list-regex-pattern-sets

The following code example shows how to use `list-regex-pattern-sets`.

AWS CLI

To retrieve a list of regex pattern sets

The following `list-regex-pattern-sets` retrieves all regex pattern sets for the account that are defined in the region `us-west-2`.

```
aws wafv2 list-regex-pattern-sets \  
--scope REGIONAL \  
--region us-west-2
```

Output:

```
{  
  "NextMarker": "regexPatterSet01",  
  "RegexPatternSets": [  
    {  
      "ARN": "arn:aws:wafv2:us-west-2:123456789012:regional/regexpatternset/  
regexPatterSet01/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
      "Description": "Test web-acl",  
      "Name": "regexPatterSet01",  
      "LockToken": "f17743f7-0000-0000-0000-19a8b93bfb01",  
      "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"  
    }  
  ]  
}
```

For more information, see [IP Sets and Regex Pattern Sets](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [ListRegexPatternSets](#) in *AWS CLI Command Reference*.

`list-resources-for-web-acl`

The following code example shows how to use `list-resources-for-web-acl`.

AWS CLI

To retrieve the resources associated with a web ACL

The following `list-resources-for-web-acl` retrieves the API Gateway REST API resources that are currently associated with the specified web ACL in the region `us-west-2`.

```
aws wafv2 list-resources-for-web-acl \  
  --web-acl-arn arn:aws:wafv2:us-west-2:123456789012:regional/webacl/TestWebAcl/  
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \  
  --resource-type API_GATEWAY \  
  --region us-west-2
```

Output:

```
{  
  "ResourceArns": [  
    "arn:aws:apigateway:us-west-2::/restapis/EXAMPLE111/stages/testing"  
  ]  
}
```

For more information, see [Associating or Disassociating a Web ACL with an AWS Resource](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [ListResourcesForWebAcl](#) in *AWS CLI Command Reference*.

list-rule-groups

The following code example shows how to use `list-rule-groups`.

AWS CLI

To retrieve a list of custom rule groups

The following `list-rule-groups` retrieves all custom rule groups that are defined for the account for the specified scope and region location.

```
aws wafv2 list-rule-groups \  
  --scope REGIONAL \  
  --region us-west-2
```

Output:

```
{  
  "RuleGroups": [  
    {  
      "ARN": "arn:aws:wafv2:us-west-2:123456789012:regional/rulegroup/  
TestRuleGroup/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
```

```

        "Description": "",
        "Name": "TestRuleGroup",
        "LockToken": "1eb5ec48-0000-0000-0000-ee9b906c541e",
        "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    },
    {
        "ARN": "arn:aws:wafv2:us-west-2:123456789012:regional/rulegroup/test/
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "Description": "",
        "Name": "test",
        "LockToken": "b0f4583e-998b-4880-9069-3fbe45738b43",
        "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
    }
],
"NextMarker": "test"
}

```

For more information, see [Managing Your Own Rule Groups](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [ListRuleGroups](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To retrieve all tags for an AWS WAF resource

The following `list-tags-for-resource` retrieves the list of all tag key, value pairs for the specified web ACL.

```

aws wafv2 list-tags-for-resource \
  --resource-arn arn:aws:wafv2:us-west-2:123456789012:regional/webacl/testwebacl2/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111

```

Output:

```

{
  "NextMarker": "",
  "TagInfoForResource": {

```

```
    "ResourceARN": "arn:aws:wafv2:us-west-2:123456789012:regional/webacl/
testwebacl2/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "TagList": [
        ]
    }
}
```

For more information, see [Getting Started with AWS WAF](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

list-web-acls

The following code example shows how to use `list-web-acls`.

AWS CLI

To retrieve the web ACLs for a scope

The following `list-web-acls` retrieves all web ACLs that are defined for the account for the specified scope.

```
aws wafv2 list-web-acls \
  --scope REGIONAL
```

Output:

```
{
  "NextMarker": "Testt",
  "WebACLs": [
    {
      "ARN": "arn:aws:wafv2:us-west-2:123456789012:regional/webacl/Testt/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "Description": "sssss",
      "Name": "Testt",
      "LockToken": "7f36cb30-74ef-4cff-8cd4-a77e1aba1746",
      "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    }
  ]
}
```

For more information, see [Managing and Using a Web Access Control List \(Web ACL\)](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [ListWebAcls](#) in *AWS CLI Command Reference*.

put-logging-configuration

The following code example shows how to use `put-logging-configuration`.

AWS CLI

To add a logging configuration to a web ACL

The following `put-logging-configuration` adds the Amazon Kinesis Data Firehose logging configuration `aws-waf-logs-custom-transformation` to the specified web ACL, with no fields redacted from the logs.

```
aws wafv2 put-logging-configuration \  
  --logging-configuration ResourceArn=arn:aws:wafv2:us-  
west-2:123456789012:regional/webacl/test-cli/a1b2c3d4-5678-90ab-  
cdef-EXAMPLE11111,LogDestinationConfigs=arn:aws:firehose:us-  
west-2:123456789012:deliverystream/aws-waf-logs-custom-transformation \  
  --region us-west-2
```

Output:

```
{  
  "LoggingConfiguration":{  
    "ResourceArn":"arn:aws:wafv2:us-west-2:123456789012:regional/webacl/test-  
cli/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",  
    "LogDestinationConfigs":[  
      "arn:aws:firehose:us-west-2:123456789012:deliverystream/aws-waf-logs-  
custom-transformation"  
    ]  
  }  
}
```

For more information, see [Logging Web ACL Traffic Information](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [PutLoggingConfiguration](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To add tags to an AWS WAF resource

The following `tag-resource` example adds a tag with a key of `Name` and value set to `AWSWAF` to the specified web ACL.

```
aws wafv2 tag-resource \  
  --resource-arn arn:aws:wafv2:us-west-2:123456789012:regional/webacl/  
  apiGatewayWebAcl/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \  
  --tags Key=Name,Value=AWSWAF
```

This command produces no output.

For more information, see [Getting Started with AWS WAF](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To remove tags from an AWS WAF resource

The following `untag-resource` example removes the tag with the key `KeyName` from the specified web ACL.

```
aws wafv2 untag-resource \  
  --resource-arn arn:aws:wafv2:us-west-2:123456789012:regional/webacl/  
  apiGatewayWebAcl/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \  
  --tag-keys "KeyName"
```

This command produces no output.

For more information, see [Getting Started with AWS WAF](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-ip-set

The following code example shows how to use `update-ip-set`.

AWS CLI

To modify the settings for an existing IP set

The following `update-ip-set` updates the settings for the specified IP set. This call requires an ID, which you can obtain from the call, `list-ip-sets`, and a lock token which you can obtain from the calls, `list-ip-sets` and `get-ip-set`. This call also returns a lock token that you can use for a subsequent update.

```
aws wafv2 update-ip-set \  
  --name testip \  
  --scope REGIONAL \  
  --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \  
  --addresses 198.51.100.0/16 \  
  --lock-token 447e55ac-2396-4c6d-b9f9-86b67c17f8b5
```

Output:

```
{  
  "NextLockToken": "0674c84b-0304-47fe-8728-c6bff46af8fc"  
}
```

For more information, see [IP Sets and Regex Pattern Sets](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [UpdateIpSet](#) in *AWS CLI Command Reference*.

update-regex-pattern-set

The following code example shows how to use `update-regex-pattern-set`.

AWS CLI

To modify the settings for an existing regex pattern set

The following `update-regex-pattern-set` updates the settings for the specified regex pattern set. This call requires an ID, which you can obtain from the call, `list-regex-pattern-sets`, and a lock token which you can obtain from the calls, `list-regex-pattern-sets` and `get-regex-pattern-set`. This call also returns a lock token that you can use for a subsequent update.

```
aws wafv2 update-regex-pattern-set \  
  --name ExampleRegex \  
  --scope REGIONAL \  
  --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \  
  --regular-expression-list RegexString="^.+ $" \  
  --lock-token ed207e9c-82e9-4a77-aadd-81e6173ab7eb
```

Output:

```
{  
  "NextLockToken": "12ebc73e-fa68-417d-a9b8-2bdd761a4fa5"  
}
```

For more information, see [IP Sets and Regex Pattern Sets](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [UpdateRegexPatternSet](#) in *AWS CLI Command Reference*.

update-rule-group

The following code example shows how to use `update-rule-group`.

AWS CLI

To update a custom rule group

The following `update-rule-group` changes the visibility configuration for an existing custom rule group. This call requires an ID, which you can obtain from the call, `list-rule-groups`, and a lock token which you can obtain from the calls, `list-rule-groups` and `get-rule-group`. This call also returns a lock token that you can use for a subsequent update.

```
aws wafv2 update-rule-group \  
  --name TestRuleGroup \  
  --scope REGIONAL \  
  --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \  
  --lock-token ed207e9c-82e9-4a77-aadd-81e6173ab7eb
```

```

--lock-token 7b3bcec2-0000-0000-0000-563bf47249f0 \
--visibility-config
SampledRequestsEnabled=false,CloudWatchMetricsEnabled=false,MetricName=TestMetricsForRuleGr
\
--region us-west-2

```

Output:

```

{
  "NextLockToken": "1eb5ec48-0000-0000-0000-ee9b906c541e"
}

```

For more information, see [Managing Your Own Rule Groups](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [UpdateRuleGroup](#) in *AWS CLI Command Reference*.

update-web-acl

The following code example shows how to use `update-web-acl`.

AWS CLI**To update a web ACL**

The following `update-web-acl` changes settings for an existing web ACL. This call requires an ID, which you can obtain from the call, `list-web-acls`, and a lock token and other settings, which you can obtain from the call `get-web-acl`. This call also returns a lock token that you can use for a subsequent update.

```

aws wafv2 update-web-acl \
  --name TestWebAcl \
  --scope REGIONAL \
  --id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
  --lock-token 2294b3a1-0000-0000-0000-a3ae04329de9 \
  --default-action Block={} \
  --visibility-config
SampledRequestsEnabled=false,CloudWatchMetricsEnabled=false,MetricName=NewMetricTestWebAcl
\
  --rules file://waf-rule.json \
  --region us-west-2

```

Output:

```
{
  "NextLockToken": "714a0cfb-0000-0000-0000-2959c8b9a684"
}
```

For more information, see [Managing and Using a Web Access Control List \(Web ACL\)](#) in the *AWS WAF, AWS Firewall Manager, and AWS Shield Advanced Developer Guide*.

- For API details, see [UpdateWebAcl](#) in *AWS CLI Command Reference*.

Amazon WorkDocs examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon WorkDocs.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

abort-document-version-upload

The following code example shows how to use `abort-document-version-upload`.

AWS CLI

To stop a document version upload

This example stops a previously initiated document version upload.

Command:

```
aws workdocs abort-document-version-upload --document-id
feaba64d4efdf271c2521b60a2a44a8f057e84beaabbe22f01267313209835f2 --version-id
1536773972914-ddb67663e782e7ce8455ebc962217cf9f9e47b5a9a702e5c84dccc417da9313
```

Output:

```
None
```

- For API details, see [AbortDocumentVersionUpload](#) in *AWS CLI Command Reference*.

activate-user

The following code example shows how to use `activate-user`.

AWS CLI**To activate a user**

This example activates an inactive user.

Command:

```
aws workdocs activate-user --user-id
"S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c"
```

Output:

```
{
  "User": {
    "Id": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
    "Username": "exampleUser",
    "EmailAddress": "exampleUser@site.awsapps.com",
    "GivenName": "Example",
    "Surname": "User",
    "OrganizationId": "d-926726012c",
    "RootFolderId":
"75f67c183aa1217409ac87576a45c03a5df5e6d8c51c35c01669970538e86cd0",
    "RecycleBinFolderId":
"642b7dd3e60b14204534f3df7b1959e01b5d170f8c2707f410e40a8149120a57",
    "Status": "ACTIVE",
```

```

    "Type": "MINIMALUSER",
    "CreatedTimestamp": 1521226107.747,
    "ModifiedTimestamp": 1525297406.462,
    "Storage": {
      "StorageUtilizedInBytes": 0,
      "StorageRule": {
        "StorageAllocatedInBytes": 0,
        "StorageType": "QUOTA"
      }
    }
  }
}
}
}

```

- For API details, see [ActivateUser](#) in *AWS CLI Command Reference*.

add-resource-permissions

The following code example shows how to use `add-resource-permissions`.

AWS CLI

To add permissions for a resource

This example adds permissions to the resource for the specified principals.

Command:

```

aws workdocs add-resource-permissions --resource-id
d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65 --principals
Id=anonymous,Type=ANONYMOUS,Role=VIEWER

```

Output:

```

{
  "ShareResults": [
    {
      "PrincipalId": "anonymous",
      "Role": "VIEWER",
      "Status": "SUCCESS",
      "ShareId":
"d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65",
      "StatusMessage": ""
    }
  ]
}

```

```

    }
  ]
}

```

- For API details, see [AddResourcePermissions](#) in *AWS CLI Command Reference*.

create-comment

The following code example shows how to use `create-comment`.

AWS CLI

To add a new comment

This example adds a new comment to the specified document version.

Command:

```

aws workdocs create-comment --document-id
15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3 --version-id
1521672507741-9f7df0ea5dd0b121c4f3564a0c7c0b4da95cd12c635d3c442af337a88e297920 --
text "This is a comment."

```

Output:

```

{
  "Comment": {
    "CommentId": "1534799058197-
c7f5c84de9115875bbca93e0367bbebac609541d461636b760849b88b1609dd5",
    "ThreadId": "1534799058197-
c7f5c84de9115875bbca93e0367bbebac609541d461636b760849b88b1609dd5",
    "Text": "This is a comment.",
    "Contributor": {
      "Id": "arn:aws:iam::123456789123:user/exampleUser",
      "Username": "exampleUser",
      "GivenName": "Example",
      "Surname": "User",
      "Status": "ACTIVE"
    },
    "CreatedTimestamp": 1534799058.197,
    "Status": "PUBLISHED",
    "Visibility": "PUBLIC"
  }
}

```



```
}  
}
```

- For API details, see [CreateComment](#) in *AWS CLI Command Reference*.

create-custom-metadata

The following code example shows how to use `create-custom-metadata`.

AWS CLI

To create custom metadata

This example creates custom metadata for the specified document.

Command:

```
aws workdocs create-custom-metadata --resource-id  
d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65 --custom-metadata  
KeyName1=example,KeyName2=example2
```

Output:

```
None
```

- For API details, see [CreateCustomMetadata](#) in *AWS CLI Command Reference*.

create-folder

The following code example shows how to use `create-folder`.

AWS CLI

To create a folder

This example creates a folder.

Command:

```
aws workdocs create-folder --name documents --parent-folder-id  
1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678
```

Output:

```
{
  "Metadata": {
    "Id": "50893c0af679524d1a0e0651130ed6d073e1a05f95bd12c42dcde5d35634ed08",
    "Name": "documents",
    "CreatorId": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
    "ParentFolderId":
"1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678",
    "CreatedTimestamp": 1534450467.622,
    "ModifiedTimestamp": 1534450467.622,
    "ResourceState": "ACTIVE",
    "Signature": "",
    "Size": 0,
    "LatestVersionSize": 0
  }
}
```

- For API details, see [CreateFolder](#) in *AWS CLI Command Reference*.

create-labels

The following code example shows how to use `create-labels`.

AWS CLI**To create labels**

This example creates a series of labels for a document.

Command:

```
aws workdocs create-labels --resource-id
d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65 --labels
"documents" "examples" "my_documents"
```

Output:

```
None
```

- For API details, see [CreateLabels](#) in *AWS CLI Command Reference*.

create-notification-subscription

The following code example shows how to use `create-notification-subscription`.

AWS CLI

To create a notification subscription

The following `create-notification-subscription` example configures a notification subscription for the specified Amazon WorkDocs organization.

```
aws workdocs create-notification-subscription \  
  --organization-id d-123456789c \  
  --protocol HTTPS \  
  --subscription-type ALL \  
  --notification-endpoint "https://example.com/example"
```

Output:

```
{  
  "Subscription": {  
    "SubscriptionId": "123ab4c5-678d-901e-f23g-45h6789j0123",  
    "EndPoint": "https://example.com/example",  
    "Protocol": "HTTPS"  
  }  
}
```

For more information, see [Subscribe to Notifications](#) in the *Amazon WorkDocs Developer Guide*.

- For API details, see [CreateNotificationSubscription](#) in *AWS CLI Command Reference*.

create-user

The following code example shows how to use `create-user`.

AWS CLI

To create a new user

This example creates a new user in a Simple AD or Microsoft AD directory.

Command:

```
aws workdocs create-user --organization-id d-926726012c --username exampleUser2
--email-address exampleUser2@site.awsapps.com --given-name example2Name --surname
example2Surname --password examplePa$$w0rd
```

Output:

```
{
  "User": {
    "Id": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
    "Username": "exampleUser2",
    "EmailAddress": "exampleUser2@site.awsapps.com",
    "GivenName": "example2Name",
    "Surname": "example2Surname",
    "OrganizationId": "d-926726012c",
    "RootFolderId":
"35b886cb17198cbd547655e58b025dff0cf34aaed638be52009567e23dc67390",
    "RecycleBinFolderId":
"9858c3e9ed4c2460dde9aadb4c69fde998070dd46e5e985bd08ec6169ea249ff",
    "Status": "ACTIVE",
    "Type": "MINIMALUSER",
    "CreatedTimestamp": 1535478836.584,
    "ModifiedTimestamp": 1535478836.584,
    "Storage": {
      "StorageUtilizedInBytes": 0,
      "StorageRule": {
        "StorageAllocatedInBytes": 0,
        "StorageType": "QUOTA"
      }
    }
  }
}
```

- For API details, see [CreateUser](#) in *AWS CLI Command Reference*.

deactivate-user

The following code example shows how to use deactivate-user.

AWS CLI

To deactivate a user

This example deactivates an active user.

Command:

```
aws workdocs deactivate-user --user-id
"S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c"
```

Output:

```
None
```

- For API details, see [DeactivateUser](#) in *AWS CLI Command Reference*.

delete-comment

The following code example shows how to use `delete-comment`.

AWS CLI

To delete a specified comment from a document version

This example deletes the specified comment from the specified document version.

Command:

```
aws workdocs delete-comment --document-id
15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3 --version-id
1521672507741-9f7df0ea5dd0b121c4f3564a0c7c0b4da95cd12c635d3c442af337a88e297920 --
comment-id 1534799058197-
c7f5c84de9115875bbca93e0367bbebac609541d461636b760849b88b1609dd5
```

Output:

```
None
```

- For API details, see [DeleteComment](#) in *AWS CLI Command Reference*.

delete-custom-metadata

The following code example shows how to use `delete-custom-metadata`.

AWS CLI

To delete custom metadata from a resource

This example deletes all custom metadata from the specified resource.

Command:

```
aws workdocs delete-custom-metadata --resource-id
d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65 --delete-all
```

Output:

```
None
```

- For API details, see [DeleteCustomMetadata](#) in *AWS CLI Command Reference*.

delete-document

The following code example shows how to use `delete-document`.

AWS CLI

To delete a document

This example deletes the specified document.

Command:

```
aws workdocs delete-document --document-id
b83ed5e5b167b65ef69de9d597627ff1a0d4f07a45e67f1fab7d26b54427de0a
```

Output:

```
None
```

- For API details, see [DeleteDocument](#) in *AWS CLI Command Reference*.

delete-folder-contents

The following code example shows how to use `delete-folder-contents`.

AWS CLI

To delete the contents of a folder

This example deletes the contents of the specified folder.

Command:

```
aws workdocs delete-folder-contents --folder-id
26fa8aa4ba2071447c194f7b150b07149dbdb9e1c8a301872dcd93a4735ce65d
```

Output:

```
None
```

- For API details, see [DeleteFolderContents](#) in *AWS CLI Command Reference*.

delete-folder

The following code example shows how to use delete-folder.

AWS CLI

To delete a folder

This example deletes the specified folder.

Command:

```
aws workdocs delete-folder --folder-id
26fa8aa4ba2071447c194f7b150b07149dbdb9e1c8a301872dcd93a4735ce65d
```

Output:

```
None
```

- For API details, see [DeleteFolder](#) in *AWS CLI Command Reference*.

delete-labels

The following code example shows how to use delete-labels.

AWS CLI

To delete labels

This example deletes the specified labels from a document.

Command:

```
aws workdocs delete-labels --resource-id
d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65 --labels
"documents" "examples"
```

Output:

```
None
```

- For API details, see [DeleteLabels](#) in *AWS CLI Command Reference*.

delete-notification-subscription

The following code example shows how to use delete-notification-subscription.

AWS CLI

To delete a notification subscription

The following delete-notification-subscription example deletes the specified notification subscription.

```
aws workdocs delete-notification-subscription \
--subscription-id 123ab4c5-678d-901e-f23g-45h6789j0123 \
--organization-id d-123456789c
```

This command produces no output.

For more information, see [Subscribe to Notifications](#) in the *Amazon WorkDocs Developer Guide*.

- For API details, see [DeleteNotificationSubscription](#) in *AWS CLI Command Reference*.

delete-user

The following code example shows how to use delete-user.

AWS CLI

To delete a user

This example deletes a user.

Command:

```
aws workdocs delete-user --user-id
"S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c"
```

Output:

```
None
```

- For API details, see [DeleteUser](#) in *AWS CLI Command Reference*.

describe-activities

The following code example shows how to use describe-activities.

AWS CLI

To get a list of user activities

This example returns a list of the latest user activities for the specified organization, with a limit set for the latest two activities.

Command:

```
aws workdocs describe-activities --organization-id d-926726012c --limit 2
```

Output:

```
{
  "UserActivities": [
    {
      "Type": "DOCUMENT_VERSION_DOWNLOADED",
      "TimeStamp": 1534800122.17,
```

```

    "Initiator": {
      "Id": "arn:aws:iam::123456789123:user/exampleUser"
    },
    "ResourceMetadata": {
      "Type": "document",
      "Name": "updatedDoc",
      "Id":
"15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3",
      "Owner": {
        "Id":
"S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
        "GivenName": "exampleName",
        "Surname": "exampleSurname"
      }
    }
  },
  {
    "Type": "DOCUMENT_VERSION_VIEWED",
    "TimeStamp": 1534799079.207,
    "Initiator": {
      "Id": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
      "GivenName": "exampleName",
      "Surname": "exampleSurname"
    },
    "ResourceMetadata": {
      "Type": "document",
      "Name": "updatedDoc",
      "Id":
"15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3",
      "Owner": {
        "Id":
"S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
        "GivenName": "exampleName",
        "Surname": "exampleSurname"
      }
    }
  }
],
"Marker":
"DnF1ZXJ5VGhlbkZldGNoAgAAAAAAS7Fm1TaU10d1FTU1h1UU00VVFibD1RWhcAAAAAAAJTRY3bWh5eUgzaVF1ZX"
}

```

- For API details, see [DescribeActivities](#) in *AWS CLI Command Reference*.

describe-comments

The following code example shows how to use `describe-comments`.

AWS CLI

To list all comments for a specified document version

This example lists all the comments for the specified document version.

Command:

```
aws workdocs describe-comments --document-id
15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3 --version-id
1521672507741-9f7df0ea5dd0b121c4f3564a0c7c0b4da95cd12c635d3c442af337a88e297920
```

Output:

```
{
  "Comments": [
    {
      "CommentId": "1534799058197-
c7f5c84de9115875bbca93e0367bbebac609541d461636b760849b88b1609dd5",
      "ThreadId": "1534799058197-
c7f5c84de9115875bbca93e0367bbebac609541d461636b760849b88b1609dd5",
      "Text": "This is a comment.",
      "Contributor": {
        "Username": "arn:aws:iam::123456789123:user/exampleUser",
        "Type": "USER"
      },
      "CreatedTimestamp": 1534799058.197,
      "Status": "PUBLISHED",
      "Visibility": "PUBLIC"
    }
  ]
}
```

- For API details, see [DescribeComments](#) in *AWS CLI Command Reference*.

describe-document-versions

The following code example shows how to use `describe-document-versions`.

AWS CLI

To retrieve a document's versions

This example retrieves the document versions for the specified document, including initialized versions and a URL for the source document.

Command:

```
aws workdocs describe-document-versions --document-id
d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65 --fields SOURCE
```

Output:

```
{
  "DocumentVersions": [
    {
      "Id":
      "1534452029587-15e129dfc187505c407588df255be83de2920d733859f1d2762411d22a83e3ef",
      "Name": "exampleDoc.docx",
      "ContentType": "application/vnd.openxmlformats-
officedocument.wordprocessingml.document",
      "Size": 13922,
      "Signature": "1a23456b78901c23d4ef56gh7EXAMPLE",
      "Status": "ACTIVE",
      "CreatedTimestamp": 1534452029.587,
      "ModifiedTimestamp": 1534452029.849,
      "CreatorId":
      "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
      "Source": {
        "ORIGINAL": "https://gb-us-west-2-prod-doc-source.s3.us-
west-2.amazonaws.com/
d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65/1534452029587-15e129dfc1875
response-content-disposition=attachment%3B%20filename%2A
%3DUTF-8%27%27exampleDoc29.docx&X-Amz-Algorithm=AWS1-ABCD-EFG234&X-Amz-
Date=20180816T204149Z&X-Amz-SignedHeaders=host&X-Amz-Expires=900&X-Amz-
Credential=AKIAIOSFODNN7EXAMPLE%2F20180816%2Fus-west-2%2Fs3%2Faws1_request&X-Amz-
Signature=01Ab2c34d567e8f90123g456hi78j901k23456781901234mno56pqr78EXAMPLE"
      }
    },
    {
      "Id": "1529005196082-
bb75fa19abc287699cb07147f75816dce43a53a10f28dc001bf61ef2fab01c59",
```

```

    "Name": "exampleDoc.pdf",
    "ContentType": "application/pdf",
    "Size": 425916,
    "Signature": "1a23456b78901c23d4ef56gh7EXAMPLE",
    "Status": "ACTIVE",
    "CreatedTimestamp": 1529005196.082,
    "ModifiedTimestamp": 1529005196.796,
    "CreatorId":
  "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
    "Source": {
      "ORIGINAL": "https://gb-us-west-2-prod-doc-source.s3.us-
west-2.amazonaws.com/
d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65/1529005196082-
bb75fa19abc287699cb07147f75816dce43a53a10f28dc001bf61ef2fab01c59?
response-content-disposition=attachment%3B%20filename%2A
%3DUTF-8%27%27exampleDoc29.pdf&X-Amz-Algorithm=AWS1-ABCD-EFG234&X-Amz-
Date=20180816T204149Z&X-Amz-SignedHeaders=host&X-Amz-Expires=900&X-Amz-
Credential=AKIAIOSFODNN7EXAMPLE%2F20180816%2Fus-west-2%2Fs3%2Faws1_request&X-Amz-
Signature=01Ab2c34d567e8f90123g456hi78j901k23456781901234mno56pqr78EXAMPLE"
    }
  }
]
}

```

- For API details, see [DescribeDocumentVersions](#) in *AWS CLI Command Reference*.

describe-folder-contents

The following code example shows how to use `describe-folder-contents`.

AWS CLI

To describe the contents of a folder

This example describes all the active contents of the specified folder, including its documents and subfolders, sorted by date in ascending order.

Command:

```

aws workdocs describe-folder-contents --folder-id
1e9e93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678 --sort DATE --
order ASCENDING --type ALL

```

Output:

```
{
  "Folders": [
    {
      "Id": "50893c0af679524d1a0e0651130ed6d073e1a05f95bd12c42dcde5d35634ed08",
      "Name": "testing",
      "CreatorId":
      "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
      "ParentFolderId":
      "1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678",
      "CreatedTimestamp": 1534450467.622,
      "ModifiedTimestamp": 1534451113.504,
      "ResourceState": "ACTIVE",
      "Signature": "1a23456b78901c23d4ef56gh7EXAMPLE",
      "Size": 23019,
      "LatestVersionSize": 11537
    }
  ],
  "Documents": [
    {
      "Id": "d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65",
      "CreatorId":
      "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
      "ParentFolderId":
      "1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678",
      "CreatedTimestamp": 1529005196.082,
      "ModifiedTimestamp": 1534452483.01,
      "LatestVersionMetadata": {
        "Id":
        "1534452029587-15e129dfc187505c407588df255be83de2920d733859f1d2762411d22a83e3ef",
        "Name": "exampleDoc.docx",
        "ContentType": "application/vnd.openxmlformats-
officedocument.wordprocessingml.document",
        "Size": 13922,
        "Signature": "1a23456b78901c23d4ef56gh7EXAMPLE",
        "Status": "ACTIVE",
        "CreatedTimestamp": 1534452029.587,
        "ModifiedTimestamp": 1534452029.587,
        "CreatorId":
        "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c"
      },
      "ResourceState": "ACTIVE"
    }
  ]
}
```

```
]
}
```

- For API details, see [DescribeFolderContents](#) in *AWS CLI Command Reference*.

describe-groups

The following code example shows how to use `describe-groups`.

AWS CLI

To retrieve a list of groups

The following `describe-groups` example lists the groups associated with the specified Amazon WorkDocs organization.

```
aws workdocs describe-groups \
  --search-query "e" \
  --organization-id d-123456789c
```

Output:

```
{
  "Groups": [
    {
      "Id": "S-1-1-11-1122222222-2222233333-3333334444-4444&d-123456789c",
      "Name": "Example Group 1"
    },
    {
      "Id": "S-1-1-11-1122222222-2222233333-3333334444-5555&d-123456789c",
      "Name": "Example Group 2"
    }
  ]
}
```

For more information, see [Getting Started with Amazon WorkDocs](#) in the *Amazon WorkDocs Administration Guide*.

- For API details, see [DescribeGroups](#) in *AWS CLI Command Reference*.

describe-notification-subscriptions

The following code example shows how to use `describe-notification-subscriptions`.

AWS CLI

To retrieve a list of notification subscriptions

The following `describe-notification-subscriptions` example retrieves the notification subscriptions for the specified Amazon WorkDocs organization.

```
aws workdocs describe-notification-subscriptions \  
  --organization-id d-123456789c
```

Output:

```
{  
  "Subscriptions": [  
    {  
      "SubscriptionId": "123ab4c5-678d-901e-f23g-45h6789j0123",  
      "EndPoint": "https://example.com/example",  
      "Protocol": "HTTPS"  
    }  
  ]  
}
```

For more information, see [Subscribe to Notifications](#) in the *Amazon WorkDocs Developer Guide*.

- For API details, see [DescribeNotificationSubscriptions](#) in *AWS CLI Command Reference*.

describe-resource-permissions

The following code example shows how to use `describe-resource-permissions`.

AWS CLI

To get a list of permissions for a resource

This example returns a list of the permissions for the specified resource (document or folder).

Command:


```
aws workdocs describe-resource-permissions --resource-id
15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3
```

Output:

```
{
  "Principals": [
    {
      "Id": "anonymous",
      "Type": "ANONYMOUS",
      "Roles": [
        {
          "Role": "VIEWER",
          "Type": "DIRECT"
        }
      ]
    },
    {
      "Id": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
      "Type": "USER",
      "Roles": [
        {
          "Role": "OWNER",
          "Type": "DIRECT"
        }
      ]
    },
    {
      "Id": "d-926726012c",
      "Type": "ORGANIZATION",
      "Roles": [
        {
          "Role": "VIEWER",
          "Type": "INHERITED"
        }
      ]
    }
  ]
}
```

- For API details, see [DescribeResourcePermissions](#) in *AWS CLI Command Reference*.

describe-users

The following code example shows how to use describe-users.

AWS CLI

To retrieve details for specified users

This example retrieves details for all the users in the specified organization.

Command:

```
aws workdocs describe-users --organization-id d-926726012c
```

Output:

```
{
  "Users": [
    {
      "Id": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
      "Username": "example1User",
      "OrganizationId": "d-926726012c",
      "RootFolderId":
"3c0e3f849dd20a9771d937b9bbcc97e18796150ae56c26d64a4fa0320a2dedc9",
      "RecycleBinFolderId":
"c277f4c4d647be1f5147b3184ffa96e1e2bf708278b696cacba68ba13b91f4fe",
      "Status": "INACTIVE",
      "Type": "USER",
      "CreatedTimestamp": 1535478999.452,
      "ModifiedTimestamp": 1535478999.452
    },
    {
      "Id": "S-1-1-11-1111111111-2222222222-3333333333-4444&d-926726012c",
      "Username": "example2User",
      "EmailAddress": "example2User@site.awsapps.com",
      "GivenName": "example2Name",
      "Surname": "example2Surname",
      "OrganizationId": "d-926726012c",
      "RootFolderId":
"35b886cb17198cbd547655e58b025dff0cf34aaed638be52009567e23dc67390",
      "RecycleBinFolderId":
"9858c3e9ed4c2460dde9aadb4c69fde998070dd46e5e985bd08ec6169ea249ff",
      "Status": "ACTIVE",
      "Type": "MINIMALUSER",
```

```
        "CreatedTimestamp": 1535478836.584,  
        "ModifiedTimestamp": 1535478836.584  
    }  
]  
}
```

- For API details, see [DescribeUsers](#) in *AWS CLI Command Reference*.

get-document-path

The following code example shows how to use `get-document-path`.

AWS CLI

To retrieve a document's path information

This example retrieves the path information (hierarchy from the root folder) for the specified document, and includes the names of the parent folders.

Command:

```
aws workdocs get-document-path --document-id  
d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65 --fields NAME
```

Output:

```
{  
  "Path": {  
    "Components": [  
      {  
        "Id":  
"a43d29cbb8e7c4d25cfee8b803a504b0dc63e760b55ad0c611c6b87691eb6ff3",  
        "Name": "/"  
      },  
      {  
        "Id":  
"1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678",  
        "Name": "Top Level Folder"  
      },  
      {  
        "Id":  
"d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65",
```

```

        "Name": "exampleDoc.docx"
      }
    ]
  }
}

```

- For API details, see [GetDocumentPath](#) in *AWS CLI Command Reference*.

get-document-version

The following code example shows how to use `get-document-version`.

AWS CLI

To retrieve version metadata for a specified document

This example retrieves version metadata for the specified document, including a source URL and custom metadata.

Command:

```

aws workdocs get-document-version --document-id
15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3 --version-id
1521672507741-9f7df0ea5dd0b121c4f3564a0c7c0b4da95cd12c635d3c442af337a88e297920 --
fields SOURCE --include-custom-metadata

```

Output:

```

{
  "Metadata": {
    "Id":
"1521672507741-9f7df0ea5dd0b121c4f3564a0c7c0b4da95cd12c635d3c442af337a88e297920",
    "Name": "exampleDoc",
    "ContentType": "application/vnd.openxmlformats-
officedocument.wordprocessingml.document",
    "Size": 11537,
    "Signature": "1a23456b78901c23d4ef56gh7EXAMPLE",
    "Status": "ACTIVE",
    "CreatedTimestamp": 1521672507.741,
    "ModifiedTimestamp": 1534451113.504,
    "CreatorId": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
    "Source": {

```

```

    "ORIGINAL": "https://gb-us-west-2-prod-doc-source.s3.us-
west-2.amazonaws.com/15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3/152167
response-content-disposition=attachment%3B%20filename%2A
%3DUTF-8%27%27exampleDoc&X-Amz-Algorithm=AWS1-ABCD-EFG234&X-Amz-
Date=20180820T212202Z&X-Amz-SignedHeaders=host&X-Amz-Expires=900&X-Amz-
Credential=AKIAIOSFODNN7EXAMPLE%2F20180820%2Fus-west-2%2Fs3%2Faws1_request&X-Amz-
Signature=01Ab2c34d567e8f90123g456hi78j901k23456781901234mno56pqr78EXAMPLE"
  }
}
}

```

- For API details, see [GetDocumentVersion](#) in *AWS CLI Command Reference*.

get-document

The following code example shows how to use `get-document`.

AWS CLI

To retrieve document details

This example retrieves the details of the specified document.

Command:

```
aws workdocs get-document --document-id
d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65
```

Output:

```
{
  "Metadata": {
    "Id": "d90d93c1fe44bad0c8471e973ebaab339090401a95e777cffa58e977d2983b65",
    "CreatorId": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
    "ParentFolderId":
"1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678",
    "CreatedTimestamp": 1529005196.082,
    "ModifiedTimestamp": 1534452483.01,
    "LatestVersionMetadata": {
      "Id":
"1534452029587-15e129dfc187505c407588df255be83de2920d733859f1d2762411d22a83e3ef",
      "Name": "exampleDoc.docx",

```

```

        "ContentType": "application/vnd.openxmlformats-officedocument.wordprocessingml.document",
        "Size": 13922,
        "Signature": "1a23456b78901c23d4ef56gh7EXAMPLE",
        "Status": "ACTIVE",
        "CreatedTimestamp": 1534452029.587,
        "ModifiedTimestamp": 1534452029.587,
        "CreatorId": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c"
    },
    "ResourceState": "ACTIVE"
}
}

```

- For API details, see [GetDocument](#) in *AWS CLI Command Reference*.

get-folder-path

The following code example shows how to use `get-folder-path`.

AWS CLI

To retrieve path information for a folder

This example retrieves the path information (hierarchy from the root folder) for the specified folder, and includes the names of the parent folders.

Command:

```
aws workdocs get-folder-path --folder-id
50893c0af679524d1a0e0651130ed6d073e1a05f95bd12c42dcde5d35634ed08 --fields NAME
```

Output:

```

{
  "Path": {
    "Components": [
      {
        "Id":
"a43d29cbb8e7c4d25cfee8b803a504b0dc63e760b55ad0c611c6b87691eb6ff3",
        "Name": "/"
      },
      {

```

```

        "Id":
        "1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678",
        "Name": "Top Level Folder"
    },
    {
        "Id":
        "50893c0af679524d1a0e0651130ed6d073e1a05f95bd12c42dcde5d35634ed08",
        "Name": "Sublevel Folder"
    }
]
}
}

```

- For API details, see [GetFolderPath](#) in *AWS CLI Command Reference*.

get-folder

The following code example shows how to use `get-folder`.

AWS CLI

To retrieve the metadata for a folder

This example retrieves the metadata for the specified folder.

Command:

```
aws workdocs get-folder --folder-id
50893c0af679524d1a0e0651130ed6d073e1a05f95bd12c42dcde5d35634ed08
```

Output:

```

{
  "Metadata": {
    "Id": "50893c0af679524d1a0e0651130ed6d073e1a05f95bd12c42dcde5d35634ed08",
    "Name": "exampleFolder",
    "CreatorId": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
    "ParentFolderId":
    "1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678",
    "CreatedTimestamp": 1534450467.622,
    "ModifiedTimestamp": 1534451113.504,
    "ResourceState": "ACTIVE",

```

```
"Signature": "1a23456b78901c23d4ef56gh7EXAMPLE",
"Size": 23019,
"LatestVersionSize": 11537
}
}
```

- For API details, see [GetFolder](#) in *AWS CLI Command Reference*.

get-resources

The following code example shows how to use `get-resources`.

AWS CLI

To retrieve shared resources

The following `get-resources` example retrieves the resources shared with the specified Amazon WorkDocs user.

```
aws workdocs get-resources \
  --user-id "S-1-1-11-1111111111-2222222222-3333333333-3333" \
  --collection-type SHARED_WITH_ME
```

Output:

```
{
  "Folders": [],
  "Documents": []
}
```

For more information, see [Sharing Files and Folders](#) in the *Amazon WorkDocs User Guide*.

- For API details, see [GetResources](#) in *AWS CLI Command Reference*.

initiate-document-version-upload

The following code example shows how to use `initiate-document-version-upload`.

AWS CLI

To initiate a document version upload

The following `initiate-document-upload` example creates a new document object and version object.

```
aws workdocs initiate-document-version-upload \
  --name exampledocname \
  --parent-folder-id
eacd546d952531c633452ed67cac23161aa0d5df2e8061223a59e8f67e7b6189
```

Output:

```
{
  "Metadata": {
    "Id": "feaba64d4efdf271c2521b60a2a44a8f057e84beaabbe22f01267313209835f2",
    "CreatorId": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
    "ParentFolderId":
    "eacd546d952531c633452ed67cac23161aa0d5df2e8061223a59e8f67e7b6189",
    "CreatedTimestamp": 1536773972.914,
    "ModifiedTimestamp": 1536773972.914,
    "LatestVersionMetadata": {
      "Id": "1536773972914-
ddb67663e782e7ce8455ebc962217cf9f9e47b5a9a702e5c84dccccd417da9313",
      "Name": "exampledocname",
      "ContentType": "application/octet-stream",
      "Size": 0,
      "Status": "INITIALIZED",
      "CreatedTimestamp": 1536773972.914,
      "ModifiedTimestamp": 1536773972.914,
      "CreatorId": "arn:aws:iam::123456789123:user/EXAMPLE"
    },
    "ResourceState": "ACTIVE"
  },
  "UploadMetadata": {
    "UploadUrl": "https://gb-us-west-2-prod-doc-source.s3.us-
west-2.amazonaws.com/
feaba64d4efdf271c2521b60a2a44a8f057e84beaabbe22f01267313209835f2/1536773972914-
ddb67663e782e7ce8455ebc962217cf9f9e47b5a9a702e5c84dccccd417da9313?X-Amz-
Algorithm=AWS1-ABCD-EFG234&X-Amz-Date=20180912T173932Z&X-Amz-SignedHeaders=content-
type%3Bhost%3Bx-amz-server-side-encryption&X-Amz-Expires=899&X-Amz-
Credential=AKIAIOSFODNN7EXAMPLE%2F20180912%2Fus-west-2%2Fs3%2Faws1_request&X-Amz-
Signature=01Ab2c34d567e8f90123g456hi78j901k23456781901234mno56pqr78EXAMPLE",
    "SignedHeaders": {
      "Content-Type": "application/octet-stream",
      "x-amz-server-side-encryption": "ABC123"
    }
  }
}
```

```
}  
  }  
}
```

- For API details, see [InitiateDocumentVersionUpload](#) in *AWS CLI Command Reference*.

remove-all-resource-permissions

The following code example shows how to use `remove-all-resource-permissions`.

AWS CLI

To remove all permissions from a specified resource

This example removes all permissions from the specified resource.

Command:

```
aws workdocs remove-all-resource-permissions --resource-id  
1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678
```

Output:

```
None
```

- For API details, see [RemoveAllResourcePermissions](#) in *AWS CLI Command Reference*.

remove-resource-permission

The following code example shows how to use `remove-resource-permission`.

AWS CLI

To remove permissions from a resource

This example removes permissions from the resource for the specified principal.

Command:

```
aws workdocs remove-resource-permission --resource-id  
1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678 --principal-id  
anonymous
```

Output:

None

- For API details, see [RemoveResourcePermission](#) in *AWS CLI Command Reference*.

update-document-version

The following code example shows how to use `update-document-version`.

AWS CLI**To change a document version status to Active**

This example changes the status of the document version to Active.

Command:

```
aws workdocs update-document-version --document-id
15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3 --version-id
1521672507741-9f7df0ea5dd0b121c4f3564a0c7c0b4da95cd12c635d3c442af337a88e297920 --
version-status ACTIVE
```

Output:

None

- For API details, see [UpdateDocumentVersion](#) in *AWS CLI Command Reference*.

update-document

The following code example shows how to use `update-document`.

AWS CLI**To update a document**

This example updates a document's name and parent folder.

Command:

```
aws workdocs update-document --document-id
15df51e0335cfcc6a2e4de9dd8be9f22ee40545ad9176f54758dcf903be982d3 --name updatedDoc
--parent-folder-id 50893c0af679524d1a0e0651130ed6d073e1a05f95bd12c42dcde5d35634ed08
```

Output:

```
None
```

- For API details, see [UpdateDocument](#) in *AWS CLI Command Reference*.

update-folder

The following code example shows how to use `update-folder`.

AWS CLI

To update a folder

This example updates a folder's name and parent folder.

Command:

```
aws workdocs update-folder --folder-id
50893c0af679524d1a0e0651130ed6d073e1a05f95bd12c42dcde5d35634ed08 --name
exampleFolder1 --parent-folder-id
1ece93e5fe75315c7407c4967918b4fd9da87ddb2a588e67b7fdaf4a98fde678
```

Output:

```
None
```

- For API details, see [UpdateFolder](#) in *AWS CLI Command Reference*.

update-user

The following code example shows how to use `update-user`.

AWS CLI

To update a user

This example updates the time zone for the specified user.

Command:

```
aws workdocs update-user --user-id
"S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c" --time-zone-id
"America/Los_Angeles"
```

Output:

```
{
  "User": {
    "Id": "S-1-1-11-1111111111-2222222222-3333333333-3333&d-926726012c",
    "Username": "exampleUser",
    "EmailAddress": "exampleUser@site.awsapps.com",
    "GivenName": "Example",
    "Surname": "User",
    "OrganizationId": "d-926726012c",
    "RootFolderId":
    "c5eceb5e1a2d1d460c9d1af8330ae117fc8d39bb1d3ed6acd0992d5ff192d986",
    "RecycleBinFolderId":
    "6ca20102926ad15f04b1d248d6d6e44f2449944eda5c758f9a1e9df6a6b7fa66",
    "Status": "ACTIVE",
    "Type": "USER",
    "TimeZoneId": "America/Los_Angeles",
    "Storage": {
      "StorageUtilizedInBytes": 0,
      "StorageRule": {
        "StorageAllocatedInBytes": 53687091200,
        "StorageType": "QUOTA"
      }
    }
  }
}
```

- For API details, see [UpdateUser](#) in *AWS CLI Command Reference*.

Amazon WorkMail examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon WorkMail.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

associate-delegate-to-resource

The following code example shows how to use `associate-delegate-to-resource`.

AWS CLI

To add a delegate to a resource

The following `associate-delegate-to-resource` command adds a delegate to a resource.

```
aws workmail associate-delegate-to-resource \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --resource-id r-68bf2d3b1c0244aab7264c24b9217443 \  
  --entity-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

This command produces no output.

- For API details, see [AssociateDelegateToResource](#) in *AWS CLI Command Reference*.

associate-member-to-group

The following code example shows how to use `associate-member-to-group`.

AWS CLI

To add a member to a group

The following `associate-member-to-group` command adds the specified member to a group.

```
aws workmail associate-member-to-group \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --group-id S-1-1-11-1122222222-2222233333-3333334444-4444 \  
  --member-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

This command produces no output.

- For API details, see [AssociateMemberToGroup](#) in *AWS CLI Command Reference*.

create-alias

The following code example shows how to use `create-alias`.

AWS CLI

To create an alias

The following `create-alias` command creates an alias for the specified entity (user or group).

```
aws workmail create-alias \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --entity-id S-1-1-11-1122222222-2222233333-3333334444-4444 \  
  --alias exampleAlias@site.awsapps.com
```

This command produces no output.

- For API details, see [CreateAlias](#) in *AWS CLI Command Reference*.

create-group

The following code example shows how to use `create-group`.

AWS CLI

To create a new group

The following `create-group` command creates a new group for the specified organization.

```
aws workmail create-group \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --group-name my-new-group
```

```
--name exampleGroup1
```

Output:

```
{
  "GroupId": "S-1-1-11-1122222222-2222233333-3333334444-4444"
}
```

- For API details, see [CreateGroup](#) in *AWS CLI Command Reference*.

create-resource

The following code example shows how to use `create-resource`.

AWS CLI

To create a new resource

The following `create-resource` command creates a new resource (meeting room) for the specified organization.

```
aws workmail create-resource \
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
  --name exampleRoom1 \
  --type ROOM
```

Output:

```
{
  "ResourceId": "r-7afe0efbade843a58cdc10251fce992c"
}
```

- For API details, see [CreateResource](#) in *AWS CLI Command Reference*.

create-user

The following code example shows how to use `create-user`.

AWS CLI

To create a new user

The following `create-user` command creates a new user.

```
aws workmail create-user \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --name exampleName \  
  --display-name exampleDisplayName \  
  --password examplePa$$w0rd
```

Output:

```
{  
  "UserId": "S-1-1-11-1111111111-2222222222-3333333333-3333"  
}
```

- For API details, see [CreateUser](#) in *AWS CLI Command Reference*.

delete-access-control-rule

The following code example shows how to use `delete-access-control-rule`.

AWS CLI

To delete an access control rule

The following `delete-access-control-rule` example deletes the specified access control rule from the specified Amazon WorkMail organization.

```
aws workmail delete-access-control-rule \  
  --organization-id m-n1pq2345678r901st2u3vx45x6789yza \  
  --name "myRule"
```

This command produces no output.

For more information, see [Working with Access Control Rules](#) in the *Amazon WorkMail Administrator Guide*.

- For API details, see [DeleteAccessControlRule](#) in *AWS CLI Command Reference*.

delete-alias

The following code example shows how to use `delete-alias`.

AWS CLI

To delete an alias

The following `delete-alias` command deletes the alias for the specified entity (user or group).

```
aws workmail delete-alias \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --entity-id S-1-1-11-1122222222-2222233333-3333334444-4444 \  
  --alias exampleAlias@site.awsapps.com
```

This command produces no output.

- For API details, see [DeleteAlias](#) in *AWS CLI Command Reference*.

delete-group

The following code example shows how to use `delete-group`.

AWS CLI

To delete an existing group

The following `delete-group` command deletes an existing group from Amazon WorkMail.

```
aws workmail delete-group \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --group-id S-1-1-11-1122222222-2222233333-3333334444-4444
```

This command produces no output.

- For API details, see [DeleteGroup](#) in *AWS CLI Command Reference*.

delete-mailbox-permissions

The following code example shows how to use `delete-mailbox-permissions`.

AWS CLI

To delete mailbox permissions

The following `delete-mailbox-permissions` command deletes mailbox permissions that were previously granted to a user or group. The entity represents the user that owns the mailbox, and the grantee represents the user or group for whom to delete permissions.

```
aws workmail delete-mailbox-permissions \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --entity-id S-1-1-11-1122222222-2222233333-3333334444-4444 \  
  --grantee-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

This command produces no output.

- For API details, see [DeleteMailboxPermissions](#) in *AWS CLI Command Reference*.

delete-resource

The following code example shows how to use `delete-resource`.

AWS CLI

To delete an existing resource

The following `delete-resource` command deletes an existing resource from Amazon WorkMail.

```
aws workmail delete-resource \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --resource-id r-7afe0efbade843a58cdc10251fce992c
```

This command produces no output.

- For API details, see [DeleteResource](#) in *AWS CLI Command Reference*.

delete-user

The following code example shows how to use `delete-user`.

AWS CLI

To delete a user

The following `delete-user` command deletes the specified user from Amazon WorkMail and all subsequent systems.

```
aws workmail delete-user \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --user-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

This command produces no output.

- For API details, see [DeleteUser](#) in *AWS CLI Command Reference*.

deregister-from-work-mail

The following code example shows how to use `deregister-from-work-mail`.

AWS CLI

To disable an existing entity

The following `deregister-from-work-mail` command disables an existing entity (user, group, or resource) from using Amazon WorkMail.

```
aws workmail deregister-from-work-mail \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --entity-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

This command produces no output.

- For API details, see [DeregisterFromWorkMail](#) in *AWS CLI Command Reference*.

describe-group

The following code example shows how to use `describe-group`.

AWS CLI

To retrieve information for a group

The following `describe-group` command retrieves information about the specified group.

```
aws workmail describe-group \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --group-id S-1-1-11-1122222222-2222233333-3333334444-4444
```

Output:

```
{
  "GroupId": "S-1-1-11-1122222222-2222233333-3333334444-4444",
  "Name": "exampleGroup1",
  "State": "ENABLED"
}
```

- For API details, see [DescribeGroup](#) in *AWS CLI Command Reference*.

describe-organization

The following code example shows how to use `describe-organization`.

AWS CLI**To retrieve information for an organization**

The following `describe-organization` command retrieves information for the specified Amazon WorkMail organization.

```
aws workmail describe-organization \
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27
```

Output:

```
{
  "OrganizationId": "m-d281d0a2fd824be5b6cd3d3ce909fd27",
  "Alias": "alias",
  "State": "Active",
  "DirectoryId": "d-926726012c",
  "DirectoryType": "VpcDirectory",
  "DefaultMailDomain": "site.awsapps.com",
  "CompletedDate": 1522693605.468,
  "ARN": "arn:aws:workmail:us-west-2:111122223333:organization/m-
n1pq2345678r901st2u3vx45x6789yza"
}
```

For more information, see [Working with Organizations](#) in the *Amazon WorkMail Administrator Guide*.

- For API details, see [DescribeOrganization](#) in *AWS CLI Command Reference*.

describe-resource

The following code example shows how to use `describe-resource`.

AWS CLI

To retrieve information for a resource

The following `describe-resource` command retrieves information about the specified resource.

```
aws workmail describe-resource \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --resource-id r-7afe0efbade843a58cdc10251fce992c
```

Output:

```
{  
  "ResourceId": "r-7afe0efbade843a58cdc10251fce992c",  
  "Name": "exampleRoom1",  
  "Type": "ROOM",  
  "BookingOptions": {  
    "AutoAcceptRequests": true,  
    "AutoDeclineRecurringRequests": false,  
    "AutoDeclineConflictingRequests": true  
  },  
  "State": "ENABLED"  
}
```

- For API details, see [DescribeResource](#) in *AWS CLI Command Reference*.

describe-user

The following code example shows how to use `describe-user`.

AWS CLI

To retrieve user information

The following `describe-user` command retrieves information about the specified user.

```
aws workmail describe-user \  
  --user-id u-7afe0efbade843a58cdc10251fce992c
```

```
--organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
--user-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

Output:

```
{  
  "UserId": "S-1-1-11-1111111111-2222222222-3333333333-3333",  
  "Name": "exampleUser1",  
  "Email": "exampleUser1@site.awsapps.com",  
  "DisplayName": "",  
  "State": "ENABLED",  
  "UserRole": "USER",  
  "EnabledDate": 1532459261.827  
}
```

- For API details, see [DescribeUser](#) in *AWS CLI Command Reference*.

disassociate-delegate-from-resource

The following code example shows how to use `disassociate-delegate-from-resource`.

AWS CLI

To remove a member from a resource

The following `disassociate-delegate-from-resource` command removes the specified member from a resource.

```
aws workmail disassociate-delegate-from-resource \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --resource-id r-68bf2d3b1c0244aab7264c24b9217443 \  
  --entity-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

This command produces no output.

- For API details, see [DisassociateDelegateFromResource](#) in *AWS CLI Command Reference*.

disassociate-member-from-group

The following code example shows how to use `disassociate-member-from-group`.

AWS CLI

To remove a member from a group

The following `disassociate-member-from-group` command removes the specified member from a group.

```
aws workmail disassociate-member-from-group \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --group-id S-1-1-11-1122222222-2222233333-3333334444-4444 \  
  --member-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

This command produces no output.

- For API details, see [DisassociateMemberFromGroup](#) in *AWS CLI Command Reference*.

get-access-control-effect

The following code example shows how to use `get-access-control-effect`.

AWS CLI

To get the effect of access control rules

The following `get-access-control-effect` example retrieves the effect of the specified Amazon WorkMail organization's access control rules for the specified IP address, access protocol action, and user ID.

```
aws workmail get-access-control-effect \  
  --organization-id m-n1pq2345678r901st2u3vx45x6789yza \  
  --ip-address "192.0.2.0" \  
  --action "WindowsOutlook" \  
  --user-id "S-1-1-11-1111111111-2222222222-3333333333-3333"
```

Output:

```
{  
  "Effect": "DENY",  
  "MatchedRules": [  
    "myRule"  
  ]  
}
```



```
}
```

For more information, see [Working with Access Control Rules](#) in the *Amazon WorkMail Administrator Guide*.

- For API details, see [GetAccessControlEffect](#) in *AWS CLI Command Reference*.

get-mailbox-details

The following code example shows how to use `get-mailbox-details`.

AWS CLI

To get a user's mailbox details

The following `get-mailbox-details` command retrieves details about the specified user's mailbox.

```
aws workmail get-mailbox-details \  
  --organization-id m-n1pq2345678r901st2u3vx45x6789yza \  
  --user-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

Output:

```
{  
  "MailboxQuota": 51200,  
  "MailboxSize": 0.03890800476074219  
}
```

For more information, see [Managing User Accounts](#) in the *Amazon WorkMail Administrator Guide*.

- For API details, see [GetMailboxDetails](#) in *AWS CLI Command Reference*.

list-access-control-rules

The following code example shows how to use `list-access-control-rules`.

AWS CLI

To list access control rules

The following `list-access-control-rules` example lists the access control rules for the specified Amazon WorkMail organization.

```
aws workmail list-access-control-rules \  
  --organization-id m-n1pq2345678r901st2u3vx45x6789yza
```

Output:

```
{  
  "Rules": [  
    {  
      "Name": "default",  
      "Effect": "ALLOW",  
      "Description": "Default WorkMail Rule",  
      "DateCreated": 0.0,  
      "DateModified": 0.0  
    },  
    {  
      "Name": "myRule",  
      "Effect": "DENY",  
      "Description": "my rule",  
      "UserIds": [  
        "S-1-1-11-1111111111-2222222222-3333333333-3333"  
      ],  
      "DateCreated": 1581635628.0,  
      "DateModified": 1581635628.0  
    }  
  ]  
}
```

For more information, see [Working with Access Control Rules](#) in the *Amazon WorkMail Administrator Guide*.

- For API details, see [ListAccessControlRules](#) in *AWS CLI Command Reference*.

list-aliases

The following code example shows how to use `list-aliases`.

AWS CLI

To list aliases for a member

The following `list-aliases` command lists aliases for the specified member (user or group).

```
aws workmail list-aliases \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --entity-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

Output:

```
{  
  "Aliases": [  
    "exampleAlias@site.awsapps.com",  
    "exampleAlias1@site.awsapps.com"  
  ]  
}
```

- For API details, see [ListAliases](#) in *AWS CLI Command Reference*.

`list-group-members`

The following code example shows how to use `list-group-members`.

AWS CLI

To list group members

The following `list-group-members` command lists the members of the specified group.

```
aws workmail list-group-members \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --group-id S-1-1-11-1122222222-2222233333-3333334444-4444
```

Output:

```
{  
  "Members": [  
    {  
      "Id": "S-1-1-11-1111111111-2222222222-3333333333-3333",  
      "Name": "exampleUser1",  
      "Type": "USER",  
      "State": "ENABLED",  
      "EnabledDate": 1532459261.827  
    }  
  ]  
}
```

```
]
}
```

- For API details, see [ListGroupMembers](#) in *AWS CLI Command Reference*.

list-groups

The following code example shows how to use `list-groups`.

AWS CLI

To retrieve a list of groups

The following `list-groups` command retrieves summaries of the groups in the specified organization.

```
aws workmail list-groups \
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27
```

Output:

```
{
  "Groups": [
    {
      "Id": "S-1-1-11-1122222222-2222233333-3333334444-4444",
      "Name": "exampleGroup1",
      "State": "DISABLED"
    },
    {
      "Id": "S-4-4-44-1122222222-2222233333-3333334444-4444",
      "Name": "exampleGroup2",
      "State": "ENABLED"
    }
  ]
}
```

- For API details, see [ListGroups](#) in *AWS CLI Command Reference*.

list-mailbox-permissions

The following code example shows how to use `list-mailbox-permissions`.

AWS CLI

To retrieve mailbox permissions

The following `list-mailbox-permissions` command retrieves the mailbox permissions associated with the specified entity's mailbox.

```
aws workmail list-mailbox-permissions \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --entity-id S-1-1-11-1111111111-2222222222-3333333333-3333
```

Output:

```
{  
  "Permissions": [  
    {  
      "GranteeId": "S-1-1-11-1122222222-2222233333-3333334444-4444",  
      "GranteeType": "USER",  
      "PermissionValues": [  
        "FULL_ACCESS"  
      ]  
    }  
  ]  
}
```

- For API details, see [ListMailboxPermissions](#) in *AWS CLI Command Reference*.

list-organizations

The following code example shows how to use `list-organizations`.

AWS CLI

To retrieve a list of organizations

The following `list-organizations` command retrieves summaries of non-deleted organizations.

```
aws workmail list-organizations
```

Output:

```
{
  "OrganizationSummaries": [
    {
      "OrganizationId": "m-d281d0a2fd824be5b6cd3d3ce909fd27",
      "Alias": "exampleAlias",
      "State": "Active"
    }
  ]
}
```

- For API details, see [ListOrganizations](#) in *AWS CLI Command Reference*.

list-resource-delegates

The following code example shows how to use `list-resource-delegates`.

AWS CLI

To list the delegates for a resource

The following `list-resource-delegates` command retrieves the delegates associated with the specified resource.

```
aws workmail list-resource-delegates \
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \
  --resource-id r-68bf2d3b1c0244aab7264c24b9217443
```

Output:

```
{
  "Delegates": [
    {
      "Id": "S-1-1-11-1111111111-2222222222-3333333333-3333",
      "Type": "USER"
    }
  ]
}
```

- For API details, see [ListResourceDelegates](#) in *AWS CLI Command Reference*.

list-resources

The following code example shows how to use `list-resources`.

AWS CLI

To retrieve a list of resources

The following `list-resources` command retrieves summaries of the resources for the specified organization.

```
aws workmail list-resources \
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27
```

Output:

```
{
  "Resources": [
    {
      "Id": "r-7afe0efbade843a58cdc10251fce992c",
      "Name": "exampleRoom1",
      "Type": "ROOM",
      "State": "ENABLED"
    }
  ]
}
```

- For API details, see [ListResources](#) in *AWS CLI Command Reference*.

list-tags-for-resource

The following code example shows how to use `list-tags-for-resource`.

AWS CLI

To list the tags for a resource

The following `list-tags-for-resource` example lists the tags for the specified Amazon WorkMail organization.

```
aws workmail list-tags-for-resource \
```

```
--resource-arn arn:aws:workmail:us-west-2:111122223333:organization/m-
n1pq2345678r901st2u3vx45x6789yza
```

Output:

```
{
  "Tags": [
    {
      "Key": "priority",
      "Value": "1"
    }
  ]
}
```

For more information, see [Tagging an Organization](#) in the *Amazon WorkMail Administrator Guide*.

- For API details, see [ListTagsForResource](#) in *AWS CLI Command Reference*.

list-users

The following code example shows how to use `list-users`.

AWS CLI

To retrieve a list of users

The following `list-users` command retrieves summaries of the users in the specified organization.

```
aws workmail list-users \
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27
```

Output:

```
{
  "Users": [
    {
      "Id": "S-1-1-11-1111111111-2222222222-3333333333-3333",
      "Email": "exampleUser1@site.awsapps.com",
      "Name": "exampleUser1",
      "State": "ENABLED",
    }
  ]
}
```



```

        "UserRole": "USER",
        "EnabledDate": 1532459261.827
    },
    {
        "Id": "S-1-1-11-1122222222-2222233333-3333334444-4444",
        "Name": "exampleGuestUser",
        "State": "DISABLED",
        "UserRole": "SYSTEM_USER"
    }
]
}

```

- For API details, see [ListUsers](#) in *AWS CLI Command Reference*.

put-access-control-rule

The following code example shows how to use `put-access-control-rule`.

AWS CLI

To put a new access control rule

The following `put-access-control-rule` example denies the specified user access to the specified Amazon WorkMail organization.

```

aws workmail put-access-control-rule \
  --name "myRule" \
  --effect "DENY" \
  --description "my rule" \
  --user-ids "S-1-1-11-1111111111-2222222222-3333333333-3333" \
  --organization-id m-n1pq2345678r901st2u3vx45x6789yza

```

This command produces no output.

For more information, see [Working with Access Control Rules](#) in the *Amazon WorkMail Administrator Guide*.

- For API details, see [PutAccessControlRule](#) in *AWS CLI Command Reference*.

put-mailbox-permissions

The following code example shows how to use `put-mailbox-permissions`.

AWS CLI

To set mailbox permissions

The following `put-mailbox-permissions` command sets full access permissions for the specified grantee (user or group). The entity represents the owner of the mailbox.

```
aws workmail put-mailbox-permissions \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --entity-id S-1-1-11-1111111111-2222222222-3333333333-3333 \  
  --grantee-id S-1-1-11-1122222222-2222233333-3333334444-4444 \  
  --permission-values FULL_ACCESS
```

This command produces no output.

- For API details, see [PutMailboxPermissions](#) in *AWS CLI Command Reference*.

register-to-work-mail

The following code example shows how to use `register-to-work-mail`.

AWS CLI

To register an existing or disabled entity

The following `register-to-work-mail` command enables the specified existing entity (user, group, or resource) to use Amazon WorkMail.

```
aws workmail register-to-work-mail \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --entity-id S-1-1-11-1122222222-2222233333-3333334444-4444 \  
  --email exampleGroup1@site.awsapps.com
```

This command produces no output.

- For API details, see [RegisterToWorkMail](#) in *AWS CLI Command Reference*.

reset-password

The following code example shows how to use `reset-password`.

AWS CLI

To reset a user's password

The following `reset-password` command resets the password for the specified user.

```
aws workmail reset-password \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --user-id S-1-1-11-1111111111-2222222222-3333333333-3333 \  
  --password examplePa$$w0rd
```

This command produces no output.

- For API details, see [ResetPassword](#) in *AWS CLI Command Reference*.

tag-resource

The following code example shows how to use `tag-resource`.

AWS CLI

To apply a tag to a resource

The following `tag-resource` example applies a tag with key "priority" and value "1" to the specified Amazon WorkMail organization.

```
aws workmail tag-resource \  
  --resource-arn arn:aws:workmail:us-west-2:111122223333:organization/m-  
n1pq2345678r901st2u3vx45x6789yza \  
  --tags "Key=priority,Value=1"
```

This command produces no output.

For more information, see [Tagging an Organization](#) in the *Amazon WorkMail Administrator Guide*.

- For API details, see [TagResource](#) in *AWS CLI Command Reference*.

untag-resource

The following code example shows how to use `untag-resource`.

AWS CLI

To untag a resource

The following `untag-resource` example removes the specified tag from the specified Amazon WorkMail organization.

```
aws workmail untag-resource \  
  --resource-arn arn:aws:workmail:us-west-2:111122223333:organization/m-  
n1pq2345678r901st2u3vx45x6789yza \  
  --tag-keys "priority"
```

This command produces no output.

For more information, see [Tagging an Organization](#) in the *Amazon WorkMail Administrator Guide*.

- For API details, see [UntagResource](#) in *AWS CLI Command Reference*.

update-mailbox-quota

The following code example shows how to use `update-mailbox-quota`.

AWS CLI

To update a user's mailbox quota

The following `update-mailbox-quota` command changes the specified user's mailbox quota.

```
aws workmail update-mailbox-quota \  
  --organization-id m-n1pq2345678r901st2u3vx45x6789yza \  
  --user-id S-1-1-11-1111111111-2222222222-3333333333-3333 \  
  --mailbox-quota 40000
```

This command produces no output.

For more information, see [Managing User Accounts](#) in the *Amazon WorkMail Administrator Guide*.

- For API details, see [UpdateMailboxQuota](#) in *AWS CLI Command Reference*.

update-primary-email-address

The following code example shows how to use `update-primary-email-address`.

AWS CLI

To update a primary email address

The following `update-primary-email-address` command updates the primary email address of the specified entity (user, group, or resource).

```
aws workmail update-primary-email-address \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --entity-id S-1-1-11-1111111111-2222222222-3333333333-3333 \  
  --email exampleUser2@site.awsapps.com
```

This command produces no output.

- For API details, see [UpdatePrimaryEmailAddress](#) in *AWS CLI Command Reference*.

update-resource

The following code example shows how to use `update-resource`.

AWS CLI

To update a resource

The following `update-resource` command updates the name of the specified resource.

```
aws workmail update-resource \  
  --organization-id m-d281d0a2fd824be5b6cd3d3ce909fd27 \  
  --resource-id r-7afe0efbade843a58cdc10251f992c \  
  --name exampleRoom2
```

This command produces no output.

- For API details, see [UpdateResource](#) in *AWS CLI Command Reference*.

Amazon WorkMail Message Flow examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Amazon WorkMail Message Flow.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

get-raw-message-content

The following code example shows how to use `get-raw-message-content`.

AWS CLI

To get the raw content of an email message

The following `get-raw-message-content` example gets the raw content of an in-transit email message and sends it to a text file named `test`.

```
aws workmailmessageflow get-raw-message-content \  
  --message-id a1b2cd34-ef5g-6h7j-k18m-npq9012345rs \  
  test
```

Contents of file `test` after command runs:

```
Subject: Hello World  
From: =?UTF-8?Q?marymajor_marymajor?= <marymajor@example.com>  
To: =?UTF-8?Q?mateojackson=40example=2Enet?= <mateojackson@example.net>  
Date: Thu, 7 Nov 2019 19:22:46 +0000  
Mime-Version: 1.0  
Content-Type: multipart/alternative;  
  boundary="=_EXAMPLE+"  
References: <mail.1ab23c45.5de6.7f890g123hj45678@storage.wm.amazon.com>
```

```
X-Priority: 3 (Normal)
X-Mailer: Amazon WorkMail
Thread-Index: EXAMPLE
Thread-Topic: Hello World
Message-Id: <mail.1ab23c45.5de6.7f890g123hj45678@storage.wm.amazon.com>

This is a multi-part message in MIME format. Your mail reader does not
understand MIME message format.
--=_EXAMPLE+
Content-Type: text/plain; charset=UTF-8
Content-Transfer-Encoding: 7bit

hello world

--=_EXAMPLE+
Content-Type: text/html; charset=utf-8
Content-Transfer-Encoding: quoted-printable

<!DOCTYPE HTML><html>
<head>
<meta name=3D"Generator" content=3D"Amazon WorkMail v3.0-4510">
<meta http-equiv=3D"Content-Type" content=3D"text/html; charset=3Dutf-8">=

<title>testing</title>
</head>
<body>
<p style=3D"margin: 0px; font-family: Arial, Tahoma, Helvetica, sans-seri=
f; font-size: small;">hello world</p>
</body>
</html>
--=_EXAMPLE+--
```

For more information, see [Retrieving Message Content with AWS Lambda](#) in the *Amazon WorkMail Administrator Guide*.

- For API details, see [GetRawMessageContent](#) in *AWS CLI Command Reference*.

WorkSpaces examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with WorkSpaces.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

create-tags

The following code example shows how to use `create-tags`.

AWS CLI

To add tags to a Workspace

The following `create-tags` example adds the specified tags to the specified Workspace.

```
aws workspaces create-tags \  
  --resource-id ws-dk1xzt417 \  
  --tags Key=Department,Value=Finance
```

This command produces no output.

For more information, see [Tag WorkSpaces resources](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [CreateTags](#) in *AWS CLI Command Reference*.

create-workspaces

The following code example shows how to use `create-workspaces`.

AWS CLI

Example 1: To create an AlwaysOn WorkSpace

The following `create-workspaces` example creates an AlwaysOn WorkSpace for the specified user, using the specified directory and bundle.

```
aws workspaces create-workspaces \
  --workspaces DirectoryId=d-926722edaf,UserName=Mateo,BundleId=wsb-0zsvgp8fc
```

Output:

```
{
  "FailedRequests": [],
  "PendingRequests": [
    {
      "WorkspaceId": "ws-kcqms853t",
      "DirectoryId": "d-926722edaf",
      "UserName": "Mateo",
      "State": "PENDING",
      "BundleId": "wsb-0zsvgp8fc"
    }
  ]
}
```

Example 2: To create an AutoStop WorkSpace

The following `create-workspaces` example creates an AutoStop WorkSpace for the specified user, using the specified directory and bundle.

```
aws workspaces create-workspaces \
  --workspaces
  DirectoryId=d-926722edaf,UserName=Mary,BundleId=wsb-0zsvgp8fc,WorkspaceProperties={RunningM
```

Output:

```
{
  "FailedRequests": [],
  "PendingRequests": [
    {
      "WorkspaceId": "ws-dk1x zr417",
```

```

        "DirectoryId": "d-926722edaf",
        "UserName": "Mary",
        "State": "PENDING",
        "BundleId": "wsb-0zsvgp8fc"
    }
]
}

```

Example 3: To create a user-decoupled WorkSpace

The following `create-workspaces` example creates a user-decoupled WorkSpace by setting the username to `[UNDEFINED]`, and specifying a WorkSpace name, directory ID, and bundle ID.

```

aws workspaces create-workspaces \
  --workspaces
  DirectoryId=d-926722edaf,UserName='"[UNDEFINED]"',WorkspaceName=MaryWorkspace1,BundleId=wsb-0zsvgp8fc

```

Output:

```

{
  "FailedRequests": [],
  "PendingRequests": [
    {
      "WorkspaceId": "ws-abcd1234",
      "DirectoryId": "d-926722edaf",
      "UserName": "[UNDEFINED]",
      "State": "PENDING",
      "BundleId": "wsb-0zsvgp8fc",
      "WorkspaceName": "MaryWorkspace1"
    }
  ]
}

```

For more information, see [Launch a virtual desktop](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [CreateWorkspaces](#) in *AWS CLI Command Reference*.

delete-tags

The following code example shows how to use `delete-tags`.

AWS CLI

To delete a tag from a WorkSpace

The following `delete-tags` example deletes the specified tag from the specified WorkSpace.

```
aws workspaces delete-tags \  
  --resource-id ws-dk1x zr417 \  
  --tag-keys Department
```

This command produces no output.

For more information, see [Tag WorkSpaces resources](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [DeleteTags](#) in *AWS CLI Command Reference*.

deregister-workspace-directory

The following code example shows how to use `deregister-workspace-directory`.

AWS CLI

To deregister a directory

The following `deregister-workspace-directory` example deregisters the specified directory.

```
aws workspaces deregister-workspace-directory \  
  --directory-id d-926722edaf
```

This command produces no output.

For more information, see [Register a directory with WorkSpaces](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [DeregisterWorkspaceDirectory](#) in *AWS CLI Command Reference*.

describe-tags

The following code example shows how to use `describe-tags`.

AWS CLI

To describe the tags for a Workspace

The following `describe-tags` example describes the tags for the specified Workspace.

```
aws workspaces describe-tags \  
  --resource-id ws-dk1x zr417
```

Output:

```
{  
  "TagList": [  
    {  
      "Key": "Department",  
      "Value": "Finance"  
    }  
  ]  
}
```

For more information, see [Tag WorkSpaces resources](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [DescribeTags](#) in *AWS CLI Command Reference*.

describe-workspace-bundles

The following code example shows how to use `describe-workspace-bundles`.

AWS CLI

To list the bundles provided by Amazon

The following `describe-workspace-bundles` example lists the names and IDs of the bundles provided by Amazon, in table format and sorted by name.

```
aws workspaces describe-workspace-bundles \  
  --owner AMAZON \  
  --query "Bundles[*].[Name, BundleId]"
```

Output:

```
[
  [
    "Standard with Amazon Linux 2",
    "wsb-clj85qzj1"
  ],
  [
    "Performance with Windows 10 (Server 2016 based)",
    "wsb-gm4d5tx2v"
  ],
  [
    "PowerPro with Windows 7",
    "wsb-1pzkp0bx4"
  ],
  [
    "Power with Amazon Linux 2",
    "wsb-2bs6k5lgn"
  ],
  [
    "Graphics with Windows 10 (Server 2019 based)",
    "wsb-03gyjnfyy"
  ],
  ...
]
```

For more information, see [WorkSpaces bundles and images](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [DescribeWorkspaceBundles](#) in *AWS CLI Command Reference*.

describe-workspace-directories

The following code example shows how to use `describe-workspace-directories`.

AWS CLI

To describe a registered directory

The following `describe-workspace-directories` example describes the specified registered directory.

```
aws workspaces describe-workspace-directories \
  --directory-ids d-926722edaf
```

Output:

```
{
  "Directories": [
    {
      "DirectoryId": "d-926722edaf",
      "Alias": "d-926722edaf",
      "DirectoryName": "example.com",
      "RegistrationCode": "WSpdx+9RJ8JT",
      "SubnetIds": [
        "subnet-9d19c4c6",
        "subnet-500d5819"
      ],
      "DnsIpAddresses": [
        "172.16.1.140",
        "172.16.0.30"
      ],
      "CustomerUserName": "Administrator",
      "IamRoleId": "arn:aws:iam::123456789012:role/workspaces_DefaultRole",
      "DirectoryType": "SIMPLE_AD",
      "WorkspaceSecurityGroupId": "sg-0d89e927e5645d7c5",
      "State": "REGISTERED",
      "WorkspaceCreationProperties": {
        "EnableWorkDocs": false,
        "EnableInternetAccess": false,
        "UserEnabledAsLocalAdministrator": true,
        "EnableMaintenanceMode": true
      },
      "WorkspaceAccessProperties": {
        "DeviceTypeWindows": "ALLOW",
        "DeviceTypeOsx": "ALLOW",
        "DeviceTypeWeb": "DENY",
        "DeviceTypeIos": "ALLOW",
        "DeviceTypeAndroid": "ALLOW",
        "DeviceTypeChromeOs": "ALLOW",
        "DeviceTypeZeroClient": "ALLOW",
        "DeviceTypeLinux": "DENY"
      },
      "Tenancy": "SHARED",
      "SelfservicePermissions": {
        "RestartWorkspace": "ENABLED",
        "IncreaseVolumeSize": "DISABLED",
        "ChangeComputeType": "DISABLED",
        "SwitchRunningMode": "DISABLED",

```

```
        "RebuildWorkspace": "DISABLED"
      }
    }
  ]
}
```

For more information, see [Manage directories for WorkSpaces](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [DescribeWorkspaceDirectories](#) in *AWS CLI Command Reference*.

describe-workspaces-connection-status

The following code example shows how to use `describe-workspaces-connection-status`.

AWS CLI

To describe the connection status of a Workspace

The following `describe-workspaces-connection-status` example describes the connection status of the specified Workspace.

```
aws workspaces describe-workspaces-connection-status \
  --workspace-ids ws-dk1x zr417
```

Output:

```
{
  "WorkspacesConnectionStatus": [
    {
      "WorkspaceId": "ws-dk1x zr417",
      "ConnectionState": "CONNECTED",
      "ConnectionStateCheckTimestamp": 1662526214.744
    }
  ]
}
```

For more information, see [Administer your WorkSpaces](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [DescribeWorkspacesConnectionStatus](#) in *AWS CLI Command Reference*.

describe-workspaces

The following code example shows how to use describe-workspaces.

AWS CLI

To describe a Workspace

The following describe-workspaces example describes the specified Workspace.

```
aws workspaces describe-workspaces \  
  --workspace-ids ws-dk1x zr417
```

Output:

```
{  
  "Workspaces": [  
    {  
      "WorkspaceId": "ws-dk1x zr417",  
      "DirectoryId": "d-926722edaf",  
      "UserName": "Mary",  
      "IpAddress": "172.16.0.175",  
      "State": "STOPPED",  
      "BundleId": "wsb-0zsvgp8fc",  
      "SubnetId": "subnet-500d5819",  
      "ComputerName": "WSAMZN-RBSLTDD9",  
      "WorkspaceProperties": {  
        "RunningMode": "AUTO_STOP",  
        "RunningModeAutoStopTimeoutInMinutes": 60,  
        "RootVolumeSizeGib": 80,  
        "UserVolumeSizeGib": 10,  
        "ComputeTypeName": "VALUE"  
      },  
      "ModificationStates": []  
    }  
  ]  
}
```

For more information, see [Administer your WorkSpaces](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [DescribeWorkspaces](#) in *AWS CLI Command Reference*.

migrate-workspace

The following code example shows how to use `migrate-workspace`.

AWS CLI

To migrate a Workspace

The following `migrate-workspace` example migrates the specified Workspace to the specified bundle.

```
aws workspaces migrate-workspace \  
  --source-workspace-id ws-dk1x zr417 \  
  --bundle-id wsb-j4d ky1gs4
```

Output:

```
{  
  "SourceWorkspaceId": "ws-dk1x zr417",  
  "TargetWorkspaceId": "ws-x5h11b kp5"  
}
```

For more information, see [Migrate a Workspace](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [MigrateWorkspace](#) in *AWS CLI Command Reference*.

modify-workspace-creation-properties

The following code example shows how to use `modify-workspace-creation-properties`.

AWS CLI

To modify a Workspace creation property of a directory

The following `modify-workspace-creation-properties` example enables the `EnableInternetAccess` property for the specified directory. This enables automatic assignment of public IP addresses for the WorkSpaces created for the directory.

```
aws workspaces modify-workspace-creation-properties \  
  --resource-id d-926722edaf \  
  --enable-internet-access
```

```
--workspace-creation-properties EnableInternetAccess=true
```

This command produces no output.

For more information, see [Update directory details for your WorkSpaces](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [ModifyWorkspaceCreationProperties](#) in *AWS CLI Command Reference*.

modify-workspace-properties

The following code example shows how to use `modify-workspace-properties`.

AWS CLI

To modify the running mode of a Workspace

The following `modify-workspace-properties` example sets the running mode of the specified Workspace to `AUTO_STOP`.

```
aws workspaces modify-workspace-properties \  
  --workspace-id ws-dk1x zr417 \  
  --workspace-properties RunningMode=AUTO_STOP
```

This command produces no output.

For more information, see [Modify a Workspace](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [ModifyWorkspaceProperties](#) in *AWS CLI Command Reference*.

modify-workspace-state

The following code example shows how to use `modify-workspace-state`.

AWS CLI

To modify the state of a Workspace

The following `modify-workspace-state` example sets the state of the specified Workspace to `ADMIN_MAINTENANCE`.

```
aws workspaces modify-workspace-state \  
  --workspace-id ws-dk1x zr417 \  
  --workspace-state ADMIN_MAINTENANCE
```

```
--workspace-id ws-dk1xzir417 \  
--workspace-state ADMIN_MAINTENANCE
```

This command produces no output.

For more information, see [Workspace maintenance](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [ModifyWorkspaceState](#) in *AWS CLI Command Reference*.

reboot-workspaces

The following code example shows how to use `reboot-workspaces`.

AWS CLI

To reboot a WorkSpace

The following `reboot-workspaces` example reboots the specified WorkSpace.

```
aws workspaces reboot-workspaces \  
--reboot-workspace-requests ws-dk1xzir417
```

Output:

```
{  
  "FailedRequests": []  
}
```

For more information, see [Reboot a WorkSpace](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [RebootWorkspaces](#) in *AWS CLI Command Reference*.

rebuild-workspaces

The following code example shows how to use `rebuild-workspaces`.

AWS CLI

To rebuild a WorkSpace

The following `rebuild-workspaces` example rebuilds the specified `WorkSpace`.

```
aws workspaces rebuild-workspaces \  
  --rebuild-workspace-requests ws-dk1x zr417
```

Output:

```
{  
  "FailedRequests": []  
}
```

For more information, see [Rebuild a WorkSpace](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [RebuildWorkspaces](#) in *AWS CLI Command Reference*.

register-workspace-directory

The following code example shows how to use `register-workspace-directory`.

AWS CLI

To register a directory

The following `register-workspace-directory` example registers the specified directory for use with Amazon WorkSpaces.

```
aws workspaces register-workspace-directory \  
  --directory-id d-926722edaf \  
  --no-enable-work-docs
```

This command produces no output.

For more information, see [Register a directory with WorkSpaces](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [RegisterWorkspaceDirectory](#) in *AWS CLI Command Reference*.

restore-workspace

The following code example shows how to use `restore-workspace`.

AWS CLI

To restore a Workspace

The following `restore-workspace` example restores the specified Workspace.

```
aws workspaces restore-workspace \  
  --workspace-id ws-dk1x zr417
```

This command produces no output.

For more information, see [Restore a Workspace](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [RestoreWorkspace](#) in *AWS CLI Command Reference*.

start-workspaces

The following code example shows how to use `start-workspaces`.

AWS CLI

To start an AutoStop Workspace

The following `start-workspaces` example starts the specified Workspace. The Workspace must have a running mode of AutoStop.

```
aws workspaces start-workspaces \  
  --start-workspace-requests WorkspaceId=ws-dk1x zr417
```

Output:

```
{  
  "FailedRequests": []  
}
```

For more information, see [Stop and start an AutoStop Workspace](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [StartWorkspaces](#) in *AWS CLI Command Reference*.

stop-workspaces

The following code example shows how to use stop-workspaces.

AWS CLI

To stop an AutoStop Workspace

The following stop-workspaces example stops the specified Workspace. The Workspace must have a running mode of AutoStop.

```
aws workspaces stop-workspaces \  
  --stop-workspace-requests WorkspaceId=ws-dk1x zr417
```

Output:

```
{  
  "FailedRequests": []  
}
```

For more information, see [Stop and start an AutoStop Workspace](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [StopWorkspaces](#) in *AWS CLI Command Reference*.

terminate-workspaces

The following code example shows how to use terminate-workspaces.

AWS CLI

To terminate a Workspace

The following terminate-workspaces example terminates the specified workspace.

```
aws workspaces terminate-workspaces \  
  --terminate-workspace-requests ws-dk1x zr417
```

Output:

```
{
```

```
"FailedRequests": []  
}
```

For more information, see [Delete a Workspace](#) in the *Amazon WorkSpaces Administration Guide*.

- For API details, see [TerminateWorkspaces](#) in *AWS CLI Command Reference*.

X-Ray examples using AWS CLI

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with X-Ray.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

batch-traces-get

The following code example shows how to use `batch-traces-get`.

AWS CLI

To get a list of traces

The following `batch-get-traces` example retrieves a list of traces specified by an ID. The full trace includes a document for each segment, compiled from all of the segment documents received with the same trace ID.

```
aws xray batch-get-traces \
```

```
--trace-ids 1-5d82881a-0a9126e92a73e971eed891b9
```

Output:

```
{
  "Traces": [
    {
      "Id": "1-5d82881a-0a9126e92a73e971eed891b9",
      "Duration": 0.232,
      "Segments": [
        {
          "Id": "54aff5735b12dd28",
          "Document": "{\"id\":\"54aff5735b12dd28\",\"name\":
\\\"Scorekeep\\\",\\\"start_time\\\":1.568835610432E9,\\\"end_time\\\":1.568835610664E9,
\\\"http\\\":{\\\"request\\\":{\\\"url\\\":\\\"http://scorekeep-env-1.m4fg2pfzpv.us-
east-2.elasticbeanstalk.com/api/user\\\",\\\"method\\\":\\\"POST\\\",\\\"user_agent\\\":
\\\"curl/7.59.0\\\",\\\"client_ip\\\":\\\"52.95.4.28\\\",\\\"x_forwarded_for\\\":true},
\\\"response\\\":{\\\"status\\\":200}},\\\"aws\\\":{\\\"elastic_beanstalk\\\":{\\\"version_label
\\\":\\\"Sample Application-1\\\",\\\"deployment_id\\\":3,\\\"environment_name\\\":\\\"Scorekeep-
env-1\\\"},\\\"ec2\\\":{\\\"availability_zone\\\":\\\"us-east-2b\\\",\\\"instance_id\\\":
\\\"i-0e3cf4d2de0f3f37a\\\"},\\\"xray\\\":{\\\"sdk_version\\\":\\\"1.1.0\\\",\\\"sdk\\\":\\\"X-Ray for
Java\\\"}},\\\"service\\\":{\\\"runtime\\\":\\\"OpenJDK 64-Bit Server VM\\\",\\\"runtime_version
\\\":\\\"1.8.0_222\\\"},\\\"trace_id\\\":\\\"1-5d82881a-0a9126e92a73e971eed891b9\\\",
\\\"origin\\\":\\\"AWS::ElasticBeanstalk::Environment\\\",\\\"subsegments\\\":[{\\\"id\\\":
\\\"2d6900034ccfe558\\\",\\\"name\\\":\\\"DynamoDB\\\",\\\"start_time\\\":1.568835610658E9,
\\\"end_time\\\":1.568835610664E9,\\\"http\\\":{\\\"response\\\":{\\\"status\\\":200,
\\\"content_length\\\":61}},\\\"aws\\\":{\\\"table_name\\\":\\\"scorekeep-user\\\",\\\"operation\\\":
\\\"UpdateItem\\\",\\\"request_id\\\":\\\"TPEIDNDUROMLPOV17U4A79555NVV4KQNS05AEMVJF66Q9ASUAAJG
\\\",\\\"resource_names\\\":[\\\"scorekeep-user\\\"]},\\\"namespace\\\":\\\"aws\\\"}]}"
        },
        {
          "Id": "0f278b6334c34e6b",
          "Document": "{\"id\":\"0f278b6334c34e6b\",\"name\":
\\\"DynamoDB\\\",\\\"start_time\\\":1.568835610658E9,\\\"end_time\\\":1.568835610664E9,
\\\"parent_id\\\":\\\"2d6900034ccfe558\\\",\\\"inferred\\\":true,\\\"http\\\":{\\\"response
\\\":{\\\"status\\\":200,\\\"content_length\\\":61}},\\\"aws\\\":{\\\"table_name
\\\":\\\"scorekeep-user\\\",\\\"operation\\\":\\\"UpdateItem\\\",\\\"request_id\\\":
\\\"TPEIDNDUROMLPOV17U4A79555NVV4KQNS05AEMVJF66Q9ASUAAJG\\\",\\\"resource_names\\\":
[\\\"scorekeep-user\\\"]},\\\"trace_id\\\":\\\"1-5d82881a-0a9126e92a73e971eed891b9\\\",\\\"origin
\\\":\\\"AWS::DynamoDB::Table\\\"}"
        }
      ]
    }
  ]
}
```



```
    ],  
    "UnprocessedTraceIds": []  
  }  
}
```

For more information, see [Using the AWS X-Ray API with the AWS CLI](#) in the *AWS X-Ray Developer Guide*.

- For API details, see [BatchTracesGet](#) in *AWS CLI Command Reference*.

create-group

The following code example shows how to use create-group.

AWS CLI

To create a group

The following create-group example creates a group resource named AdminGroup. The group gets a filter expression that defines the criteria of the group as a segment related to a specific service causing a fault or an error.

```
aws xray create-group \  
  --group-name "AdminGroup" \  
  --filter-expression "service(\"mydomain.com\") {fault OR error}"
```

Output:

```
{  
  "GroupName": "AdminGroup",  
  "GroupARN": "arn:aws:xray:us-west-2:123456789012:group/AdminGroup/123456789",  
  "FilterExpression": "service(\"mydomain.com\") {fault OR error}"  
}
```

For more information, see [Configuring Sampling, Groups, and Encryption Settings with the AWS X-Ray API](#) in the *AWS X-Ray Developer Guide*.

- For API details, see [CreateGroup](#) in *AWS CLI Command Reference*.

create-sampling-rule

The following code example shows how to use create-sampling-rule.

AWS CLI

To create a sampling rule

The following `create-sampling-rule` example creates a rule to control sampling behavior for instrumented applications. The rules are provided by a JSON file. The majority of the sampling rule fields are required to create the rule.

```
aws xray create-sampling-rule \  
  --cli-input-json file://9000-base-scorekeep.json
```

Contents of `9000-base-scorekeep.json`:

```
{  
  "SamplingRule": {  
    "RuleName": "base-scorekeep",  
    "ResourceARN": "*",  
    "Priority": 9000,  
    "FixedRate": 0.1,  
    "ReservoirSize": 5,  
    "ServiceName": "Scorekeep",  
    "ServiceType": "*",  
    "Host": "*",  
    "HTTPMethod": "*",  
    "URLPath": "*",  
    "Version": 1  
  }  
}
```

Output:

```
{  
  "SamplingRuleRecord": {  
    "SamplingRule": {  
      "RuleName": "base-scorekeep",  
      "RuleARN": "arn:aws:xray:us-west-2:123456789012:sampling-rule/base-scorekeep",  
      "ResourceARN": "*",  
      "Priority": 9000,  
      "FixedRate": 0.1,  
      "ReservoirSize": 5,  
      "ServiceName": "Scorekeep",
```

```
        "ServiceType": "*",
        "Host": "*",
        "HTTPMethod": "*",
        "URLPath": "*",
        "Version": 1,
        "Attributes": {}
    },
    "CreatedAt": 1530574410.0,
    "ModifiedAt": 1530574410.0
}
}
```

For more information, see [Configuring Sampling, Groups, and Encryption Settings with the AWS X-Ray API](#) in the *AWS X-Ray Developer Guide*.

- For API details, see [CreateSamplingRule](#) in *AWS CLI Command Reference*.

delete-group

The following code example shows how to use `delete-group`.

AWS CLI

To delete a group

The following `delete-group` example deletes the specified group resource.

```
aws xray delete-group \
  --group-name "AdminGroup" \
  --group-arn "arn:aws:xray:us-east-2:123456789012:group/AdminGroup/123456789"
```

This command produces no output.

For more information, see [Configuring Sampling, Groups, and Encryption Settings with the AWS X-Ray API](#) in the *AWS X-Ray Developer Guide*.

- For API details, see [DeleteGroup](#) in *AWS CLI Command Reference*.

delete-sampling-rule

The following code example shows how to use `delete-sampling-rule`.

AWS CLI

To delete a sampling rule

The following `delete-sampling-rule` example deletes the specified sampling rule. You can specify the group by using either the group name or group ARN.

```
aws xray delete-sampling-rule \  
  --rule-name polling-scorekeep
```

Output:

```
{  
  "SamplingRuleRecord": {  
    "SamplingRule": {  
      "RuleName": "polling-scorekeep",  
      "RuleARN": "arn:aws:xray:us-west-2:123456789012:sampling-rule/polling-  
scorekeep",  
      "ResourceARN": "*",  
      "Priority": 5000,  
      "FixedRate": 0.003,  
      "ReservoirSize": 0,  
      "ServiceName": "Scorekeep",  
      "ServiceType": "*",  
      "Host": "*",  
      "HTTPMethod": "GET",  
      "URLPath": "/api/state/*",  
      "Version": 1,  
      "Attributes": {}  
    },  
    "CreatedAt": 1530574399.0,  
    "ModifiedAt": 1530574399.0  
  }  
}
```

For more information, see [Configuring Sampling, Groups, and Encryption Settings with the AWS X-Ray API](#) in the *AWS X-Ray Developer Guide*.

- For API details, see [DeleteSamplingRule](#) in *AWS CLI Command Reference*.

get-encryption-config

The following code example shows how to use `get-encryption-config`.

AWS CLI

To retrieve the encryption configuration

The following `get-encryption-config` example retrieves the current encryption configuration for your AWS X-Ray data.

```
aws xray get-encryption-config
```

Output:

```
{
  "EncryptionConfig": {
    "KeyId": "ae4aa6d49-a4d8-9df9-a475-4ff6d7898456",
    "Status": "ACTIVE",
    "Type": "NONE"
  }
}
```

For more information, see [Configuring Sampling, Groups, and Encryption Settings with the AWS X-Ray API](#) in the *AWS X-Ray Developer Guide*.

- For API details, see [GetEncryptionConfig](#) in *AWS CLI Command Reference*.

get-group

The following code example shows how to use `get-group`.

AWS CLI

To retrieve a group

The following `get-group` example displays details for the specified group resource. The details include the group name, the group ARN, and the filter expression that defines the criteria for that group. Groups can also be retrieved by ARN.

```
aws xray get-group \
  --group-name "AdminGroup"
```

Output:

```
{
```

```
"Group": [  
  {  
    "GroupName": "AdminGroup",  
    "GroupARN": "arn:aws:xray:us-west-2:123456789012:group/  
AdminGroup/123456789",  
    "FilterExpression": "service(\"mydomain.com\") {fault OR error}"  
  }  
]
```

For more information, see [Configuring Sampling, Groups, and Encryption Settings with the AWS X-Ray API](#) in the *AWS X-Ray Developer Guide*.

- For API details, see [GetGroup](#) in *AWS CLI Command Reference*.

get-groups

The following code example shows how to use `get-groups`.

AWS CLI

To retrieve all groups

The following example displays details for all active group.

```
aws xray get-groups
```

Output:

```
{  
  "Groups": [  
    {  
      "GroupName": "AdminGroup",  
      "GroupARN": "arn:aws:xray:us-west-2:123456789012:group/  
AdminGroup/123456789",  
      "FilterExpression": "service(\"example.com\") {fault OR error}"  
    },  
    {  
      "GroupName": "SDETGroup",  
      "GroupARN": "arn:aws:xray:us-west-2:123456789012:group/  
SDETGroup/987654321",  
      "FilterExpression": "responsetime > 2"  
    }  
  ]  
}
```

```
]
}
```

For more information, see [Configuring Sampling, Groups, and Encryption Settings with the AWS X-Ray API](#) in the *AWS X-Ray Developer Guide*.

- For API details, see [GetGroups](#) in *AWS CLI Command Reference*.

get-sampling-rules

The following code example shows how to use `get-sampling-rules`.

AWS CLI

To retrieve all sampling rules

The following `get-sampling-rules` example displays details for all available sampling rules.:

```
aws xray get-sampling-rules
```

Output:

```
{
  "SamplingRuleRecords": [
    {
      "SamplingRule": {
        "RuleName": "Default",
        "RuleARN": "arn:aws:xray:us-east-1::sampling-rule/Default",
        "ResourceARN": "*",
        "Priority": 10000,
        "FixedRate": 0.01,
        "ReservoirSize": 0,
        "ServiceName": "*",
        "ServiceType": "*",
        "Host": "*",
        "HTTPMethod": "*",
        "URLPath": "*",
        "Version": 1,
        "Attributes": {}
      },
      "CreatedAt": 0.0,
      "ModifiedAt": 1530558121.0
    },
  ],
}
```

```
{
  "SamplingRule": {
    "RuleName": "base-scorekeep",
    "RuleARN": "arn:aws:xray:us-east-1::sampling-rule/base-scorekeep",
    "ResourceARN": "*",
    "Priority": 9000,
    "FixedRate": 0.1,
    "ReservoirSize": 2,
    "ServiceName": "Scorekeep",
    "ServiceType": "*",
    "Host": "*",
    "HTTPMethod": "*",
    "URLPath": "*",
    "Version": 1,
    "Attributes": {}
  },
  "CreatedAt": 1530573954.0,
  "ModifiedAt": 1530920505.0
},
{
  "SamplingRule": {
    "RuleName": "polling-scorekeep",
    "RuleARN": "arn:aws:xray:us-east-1::sampling-rule/polling-
scorekeep",
    "ResourceARN": "*",
    "Priority": 5000,
    "FixedRate": 0.003,
    "ReservoirSize": 0,
    "ServiceName": "Scorekeep",
    "ServiceType": "*",
    "Host": "*",
    "HTTPMethod": "GET",
    "URLPath": "/api/state/*",
    "Version": 1,
    "Attributes": {}
  },
  "CreatedAt": 1530918163.0,
  "ModifiedAt": 1530918163.0
}
]
```


For more information, see [Using Sampling Rules with the X-Ray API](#) in the *AWS X-Ray Developer Guide*.

- For API details, see [GetSamplingRules](#) in *AWS CLI Command Reference*.

get-sampling-targets

The following code example shows how to use `get-sampling-targets`.

AWS CLI

To request a sampling quota

The following `get-sampling-targets` example requests a sampling quota for rules that the service is using to sample requests. The response from AWS X-Ray includes a quota that can be used instead of borrowing from the reservoir.

```
aws xray get-sampling-targets \  
  --sampling-statistics-documents '[ { "RuleName": "base-scorekeep", "ClientID":  
  "ABCDEF1234567890ABCDEF10", "Timestamp": "2018-07-07T00:20:06", "RequestCount": 110,  
  "SampledCount": 20, "BorrowCount": 10 }, { "RuleName": "polling-scorekeep", 31,  
  "BorrowCount": 0 } ]'
```

Output:

```
{  
  "SamplingTargetDocuments": [  
    {  
      "RuleName": "base-scorekeep",  
      "FixedRate": 0.1,  
      "ReservoirQuota": 2,  
      "ReservoirQuotaTTL": 1530923107.0,  
      "Interval": 10  
    },  
    {  
      "RuleName": "polling-scorekeep",  
      "FixedRate": 0.003,  
      "ReservoirQuota": 0,  
      "ReservoirQuotaTTL": 1530923107.0,  
      "Interval": 10  
    }  
  ],  
  "LastRuleModification": 1530920505.0,  
}
```

```
"UnprocessedStatistics": []
}
```

For more information, see [Using Sampling Rules with the X-Ray API](#) in the *AWS X-Ray Developer Guide*.

- For API details, see [GetSamplingTargets](#) in *AWS CLI Command Reference*.

get-service-graph

The following code example shows how to use `get-service-graph`.

AWS CLI

To get a service graph

The following example displays a document within a specified time period that describes services processing incoming requests, and the downstream services that they call as a result.:

```
aws xray get-service-graph \
  --start-time 1568835392.0
  --end-time 1568835446.0
```

Output:

```
{
  "Services": [
    {
      "ReferenceId": 0,
      "Name": "Scorekeep",
      "Names": [
        "Scorekeep"
      ],
      "Root": true,
      "Type": "AWS::ElasticBeanstalk::Environment",
      "State": "active",
      "StartTime": 1568835392.0,
      "EndTime": 1568835446.0,
      "Edges": [
        {
          "ReferenceId": 1,
          "StartTime": 1568835392.0,
          "EndTime": 1568835446.0,
```

```
"SummaryStatistics": {
  "OkCount": 14,
  "ErrorStatistics": {
    "ThrottleCount": 0,
    "OtherCount": 0,
    "TotalCount": 0
  },
  "FaultStatistics": {
    "OtherCount": 0,
    "TotalCount": 0
  },
  "TotalCount": 14,
  "TotalResponseTime": 0.13
},
"ResponseTimeHistogram": [
  {
    "Value": 0.008,
    "Count": 1
  },
  {
    "Value": 0.005,
    "Count": 7
  },
  {
    "Value": 0.009,
    "Count": 1
  },
  {
    "Value": 0.021,
    "Count": 1
  },
  {
    "Value": 0.038,
    "Count": 1
  },
  {
    "Value": 0.007,
    "Count": 1
  },
  {
    "Value": 0.006,
    "Count": 2
  }
],
```

```

        "Aliases": []
      },
      ... TRUNCATED FOR BREVITY ...
    ]
  }
],
"StartTime": 1568835392.0,
"EndTime": 1568835446.0,
"ContainsOldGroupVersions": false
}

```

For more information, see [Using the AWS X-Ray API with the AWS CLI](#) in the *AWS X-Ray Developer Guide*.

- For API details, see [GetServiceGraph](#) in *AWS CLI Command Reference*.

get-trace-summaries

The following code example shows how to use `get-trace-summaries`.

AWS CLI

To get a trace summary

The following `get-trace-summaries` example retrieves IDs and metadata for traces available within a specified time frame.

```

aws xray get-trace-summaries \
  --start-time 1568835392.0 \
  --end-time 1568835446.0

```

Output:

```

[
  "http://scorekeep-env-1.123456789.us-east-2.elasticbeanstalk.com/api/move/
VSAE93HF/GSSD2NTB/DP0PCC09",
  "http://scorekeep-env-1.123456789.us-east-2.elasticbeanstalk.com/api/move/
GCQ2B35P/FREELDFT/4LRE643M",
  "http://scorekeep-env-1.123456789.us-east-2.elasticbeanstalk.com/api/game/
VSAE93HF/GSSD2NTB/starttime/1568835513",

```

```
"http://scorekeep-env-1.123456789.us-east-2.elasticbeanstalk.com/api/
move/4MQNA5NN/L99KK2RF/null"
]
```

For more information, see [Using the AWS X-Ray API with the AWS CLI](#) in the *AWS X-Ray Developer Guide*.

- For API details, see [GetTraceSummaries](#) in *AWS CLI Command Reference*.

put-encryption-config

The following code example shows how to use `put-encryption-config`.

AWS CLI

To update the encryption configuration

The following `put-encryption-config` example updates the encryption configuration for AWS X-Ray data to use the default AWS managed KMS key `aws/xray`.

```
aws xray put-encryption-config \
  --type KMS \
  --key-id alias/aws/xray
```

Output:

```
{
  "EncryptionConfig": {
    "KeyId": "arn:aws:kms:us-west-2:123456789012:key/c234g4e8-39e9-4gb0-84e2-
b0ea215cbba5",
    "Status": "UPDATING",
    "Type": "KMS"
  }
}
```

For more information, see [Configuring Sampling, Groups, and Encryption Settings with the AWS X-Ray API](#) in the *AWS X-Ray Developer Guide*.

- For API details, see [PutEncryptionConfig](#) in *AWS CLI Command Reference*.

put-trace-segments

The following code example shows how to use `put-trace-segments`.

AWS CLI

To upload a segment

The following `put-trace-segments` example uploads segment documents to AWS X-Ray. The segment document is consumed as a list of JSON segment documents.

```
aws xray put-trace-segments \
  --trace-segment-documents "{\"id\":\"20312a0e2b8809f4\",\"name
  \": \"DynamoDB\", \"trace_id\": \"1-5832862d-a43aafded3334a971fe312db\",
  \"start_time\": 1.479706157195E9, \"end_time\": 1.479706157202E9, \"parent_id\":
  \"79736b962fe3239e\", \"http\": {\"response\": {\"content_length\": 60, \"status
  \": 200}}, \"inferred\": true, \"aws\": {\"consistent_read\": false, \"table_name
  \": \"scorekeep-session-xray\", \"operation\": \"GetItem\", \"request_id\":
  \"SCAU230M6M8F038UASGC7785ARVV4KQNS05AEMVJF66Q9ASUAAJG\", \"resource_names\":
  [\"scorekeep-session-xray\"]}, \"origin\": \"AWS::DynamoDB::Table\"}"
```

Output:

```
{
  "UnprocessedTraceSegments": []
}
```

For more information, see [Sending Trace Data to AWS X-Ray](#) in the *AWS X-Ray Developer Guide*.

- For API details, see [PutTraceSegments](#) in *AWS CLI Command Reference*.

update-group

The following code example shows how to use `update-group`.

AWS CLI

To update a group

The following `update-group` example updates the criteria by which to accept traces into the group named `AdminGroup`. You can specify the desired group by using either the group name or group ARN.

```
aws xray update-group \  
  --group-name "AdminGroup" \  
  --group-arn "arn:aws:xray:us-west-2:123456789012:group/AdminGroup/123456789" \  
  --filter-expression "service(\"mydomain.com\") {fault}"
```

Output:

```
{  
  "GroupName": "AdminGroup",  
  "GroupARN": "arn:aws:xray:us-east-2:123456789012:group/AdminGroup/123456789",  
  "FilterExpression": "service(\"mydomain.com\") {fault}"  
}
```

For more information, see [Configuring Sampling, Groups, and Encryption Settings with the AWS X-Ray API](#) in the *AWS X-Ray Developer Guide*.

- For API details, see [UpdateGroup](#) in *AWS CLI Command Reference*.

update-sampling-rule

The following code example shows how to use `update-sampling-rule`.

AWS CLI

To update a sampling rule

The following `update-sampling-rule` example modifies a sampling rule's configuration. The rules are consumed from a JSON file. Only the fields being updated are required.

```
aws xray update-sampling-rule \  
  --cli-input-json file://1000-default.json
```

Contents of `1000-default.json`:

```
{  
  "SamplingRuleUpdate": {  
    "RuleName": "Default",  
    "FixedRate": 0.01,  
    "ReservoirSize": 0  
  }  
}
```

```
}
```

Output:

```
{
  "SamplingRuleRecords": [
    {
      "SamplingRule": {
        "RuleName": "Default",
        "RuleARN": "arn:aws:xray:us-west-2:123456789012:sampling-rule/
Default",
        "ResourceARN": "*",
        "Priority": 10000,
        "FixedRate": 0.01,
        "ReservoirSize": 0,
        "ServiceName": "*",
        "ServiceType": "*",
        "Host": "*",
        "HTTPMethod": "*",
        "URLPath": "*",
        "Version": 1,
        "Attributes": {}
      },
      "CreatedAt": 0.0,
      "ModifiedAt": 1529959993.0
    }
  ]
}
```

For more information, see [Configuring Sampling, Groups, and Encryption Settings with the AWS X-Ray API](#) in the *AWS X-Ray Developer Guide*.

- For API details, see [UpdateSamplingRule](#) in *AWS CLI Command Reference*.

AWS CLI with Bash script code examples

The code examples in this topic show you how to use the AWS Command Line Interface with Bash script with AWS.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Cross-service examples are sample applications that work across multiple AWS services.

Examples

- [Actions and scenarios using AWS CLI with Bash script](#)

Actions and scenarios using AWS CLI with Bash script

The following code examples show how to perform actions and implement common scenarios by using the AWS Command Line Interface with Bash script with AWS services.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Services

- [DynamoDB examples using AWS CLI with Bash script](#)
- [Amazon EC2 examples using AWS CLI with Bash script](#)
- [HealthImaging examples using AWS CLI with Bash script](#)
- [IAM examples using AWS CLI with Bash script](#)
- [Amazon S3 examples using AWS CLI with Bash script](#)
- [AWS STS examples using AWS CLI with Bash script](#)

DynamoDB examples using AWS CLI with Bash script

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Bash script with DynamoDB.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)
- [Scenarios](#)

Actions

BatchGetItem

The following code example shows how to use BatchGetItem.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function dynamodb_batch_get_item
#
# This function gets a batch of items from a DynamoDB table.
#
# Parameters:
#     -i item -- Path to json file containing the keys of the items to get.
#
# Returns:
#     The items as json output.
#
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function dynamodb_batch_get_item() {
    local item response
    local option OPTARG # Required to use getopt command in a function.
```

```
#####
# Function usage explanation
#####
function usage() {
    echo "function dynamodb_batch_get_item"
    echo "Get a batch of items from a DynamoDB table."
    echo " -i item -- Path to json file containing the keys of the items to get."
    echo ""
}

while getopts "i:h" option; do
    case "${option}" in
        i) item="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

if [[ -z "$item" ]]; then
    errecho "ERROR: You must provide an item with the -i parameter."
    usage
    return 1
fi

response=$(aws dynamodb batch-get-item \
    --request-items file://"${item}")
local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports batch-get-item operation failed.$response"
    return 1
fi

echo "$response"
```

```

    return 0
}

```

The utility functions used in this example.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-return-codes.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then

```

```

    errecho " The system environment or configuration was invalid."
elif [ "$err_code" == 254 ]; then
    errecho " The service returned an error."
elif [ "$err_code" == 255 ]; then
    errecho " 255 is a catch-all error."
fi

return 0
}

```

- For API details, see [BatchGetItem](#) in *AWS CLI Command Reference*.

BatchWriteItem

The following code example shows how to use BatchWriteItem.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```

#####
# function dynamodb_batch_write_item
#
# This function writes a batch of items into a DynamoDB table.
#
# Parameters:
#     -i item -- Path to json file containing the items to write.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function dynamodb_batch_write_item() {
    local item response
    local option OPTARG # Required to use getopt command in a function.

#####

```

```
# Function usage explanation
#####
function usage() {
    echo "function dynamodb_batch_write_item"
    echo "Write a batch of items into a DynamoDB table."
    echo " -i item -- Path to json file containing the items to write."
    echo ""
}
while getopts "i:h" option; do
    case "${option}" in
        i) item="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

if [[ -z "$item" ]]; then
    errecho "ERROR: You must provide an item with the -i parameter."
    usage
    return 1
fi

iecho "Parameters:\n"
iecho "    table_name: $table_name"
iecho "    item: $item"
iecho ""

response=$(aws dynamodb batch-write-item \
    --request-items file://"${item}")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports batch-write-item operation failed.$response"
    return 1
fi
```

```

    fi

    return 0
}

```

The utility functions used in this example.

```

#####
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#####
function iecho() {
    if [[ $VERBOSE == true ]]; then
        echo "$@"
    fi
}

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-return-codes.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#

```

```
#####
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho " One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho " Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho " Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho " Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho " The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho " The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho " 255 is a catch-all error."
  fi

  return 0
}
```

- For API details, see [BatchWriteItem](#) in *AWS CLI Command Reference*.

CreateTable

The following code example shows how to use CreateTable.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function dynamodb_create_table
#
# This function creates an Amazon DynamoDB table.
```



```

#
# Parameters:
#   -n table_name  -- The name of the table to create.
#   -a attribute_definitions -- JSON file path of a list of attributes and their
types.
#   -k key_schema  -- JSON file path of a list of attributes and their key types.
#   -p provisioned_throughput -- Provisioned throughput settings for the table.
#
# Returns:
#   0 - If successful.
#   1 - If it fails.
#####
function dynamodb_create_table() {
    local table_name attribute_definitions key_schema provisioned_throughput response
    local option OPTARG # Required to use getopt command in a function.

#####
# Function usage explanation
#####
function usage() {
    echo "function dynamodb_create_table"
    echo "Creates an Amazon DynamoDB table."
    echo " -n table_name  -- The name of the table to create."
    echo " -a attribute_definitions -- JSON file path of a list of attributes and
their types."
    echo " -k key_schema  -- JSON file path of a list of attributes and their key
types."
    echo " -p provisioned_throughput -- Provisioned throughput settings for the
table."
    echo ""
}

# Retrieve the calling parameters.
while getopt "n:a:k:p:h" option; do
    case "${option}" in
        n) table_name="${OPTARG}" ;;
        a) attribute_definitions="${OPTARG}" ;;
        k) key_schema="${OPTARG}" ;;
        p) provisioned_throughput="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)

```

```
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
done
export OPTIND=1

if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$attribute_definitions" ]]; then
    errecho "ERROR: You must provide an attribute definitions json file path the -a
parameter."
    usage
    return 1
fi

if [[ -z "$key_schema" ]]; then
    errecho "ERROR: You must provide a key schema json file path the -k parameter."
    usage
    return 1
fi

if [[ -z "$provisioned_throughput" ]]; then
    errecho "ERROR: You must provide a provisioned throughput json file path the -p
parameter."
    usage
    return 1
fi

iecho "Parameters:\n"
iecho "    table_name:  $table_name"
iecho "    attribute_definitions:  $attribute_definitions"
iecho "    key_schema:  $key_schema"
iecho "    provisioned_throughput:  $provisioned_throughput"
iecho ""

response=$(aws dynamodb create-table \
    --table-name "$table_name" \
    --attribute-definitions file://"$attribute_definitions" \
```

```

--key-schema file://"key_schema" \
--provisioned-throughput "$provisioned_throughput")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-table operation failed.$response"
    return 1
fi

return 0
}

```

The utility functions used in this example.

```

#####
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#####
function iecho() {
    if [[ $VERBOSE == true ]]; then
        echo "$@"
    fi
}

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#

```

```

# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-
return-codes.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}

```

- For API details, see [CreateTable](#) in *AWS CLI Command Reference*.

DeleteItem

The following code example shows how to use DeleteItem.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function dynamodb_delete_item
#
# This function deletes an item from a DynamoDB table.
#
# Parameters:
#     -n table_name  -- The name of the table.
#     -k keys        -- Path to json file containing the keys that identify the item to
#                       delete.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function dynamodb_delete_item() {
    local table_name keys response
    local option OPTARG # Required to use getopt command in a function.

    # #####
    # Function usage explanation
    #####
    function usage() {
        echo "function dynamodb_delete_item"
        echo "Delete an item from a DynamoDB table."
        echo " -n table_name  -- The name of the table."
        echo " -k keys        -- Path to json file containing the keys that identify the item
to delete."
        echo ""
    }
    while getopt "n:k:h" option; do
        case "${option}" in
            n) table_name="${OPTARG}" ;;
            k) keys="${OPTARG}" ;;
            h)

```

```
        usage
        return 0
        ;;
    \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
done
export OPTIND=1

if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$keys" ]]; then
    errecho "ERROR: You must provide a keys json file path the -k parameter."
    usage
    return 1
fi

iecho "Parameters:\n"
iecho "    table_name:  $table_name"
iecho "    keys:        $keys"
iecho ""

response=$(aws dynamodb delete-item \
    --table-name "$table_name" \
    --key file://" $keys")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-item operation failed.$response"
    return 1
fi

return 0
}
```

The utility functions used in this example.

```
#####
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#####
function iecho() {
    if [[ $VERBOSE == true ]]; then
        echo "$@"
    fi
}

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-return-codes.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
```

```

if [ "$err_code" == 1 ]; then
    errecho " One or more S3 transfers failed."
elif [ "$err_code" == 2 ]; then
    errecho " Command line failed to parse."
elif [ "$err_code" == 130 ]; then
    errecho " Process received SIGINT."
elif [ "$err_code" == 252 ]; then
    errecho " Command syntax invalid."
elif [ "$err_code" == 253 ]; then
    errecho " The system environment or configuration was invalid."
elif [ "$err_code" == 254 ]; then
    errecho " The service returned an error."
elif [ "$err_code" == 255 ]; then
    errecho " 255 is a catch-all error."
fi

return 0
}

```

- For API details, see [DeleteItem](#) in *AWS CLI Command Reference*.

DeleteTable

The following code example shows how to use DeleteTable.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```

#####
# function dynamodb_delete_table
#
# This function deletes a DynamoDB table.
#
# Parameters:
#     -n table_name  -- The name of the table to delete.
#

```



```

# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function dynamodb_delete_table() {
    local table_name response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function dynamodb_delete_table"
        echo "Deletes an Amazon DynamoDB table."
        echo " -n table_name -- The name of the table to delete."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "n:h" option; do
        case "${option}" in
            n) table_name="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    if [[ -z "$table_name" ]]; then
        errecho "ERROR: You must provide a table name with the -n parameter."
        usage
        return 1
    fi

    iecho "Parameters:\n"
    iecho "    table_name:  $table_name"
    iecho ""

    response=$(aws dynamodb delete-table \

```

```

    --table-name "$table_name")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-table operation failed.$response"
    return 1
fi

return 0
}

```

The utility functions used in this example.

```

#####
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#####
function iecho() {
    if [[ $VERBOSE == true ]]; then
        echo "$@"
    fi
}

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#

```

```

# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-
return-codes.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}

```

- For API details, see [DeleteTable](#) in *AWS CLI Command Reference*.

DescribeTable

The following code example shows how to use DescribeTable.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function dynamodb_describe_table
#
# This function returns the status of a DynamoDB table.
#
# Parameters:
#     -n table_name  -- The name of the table.
#
# Response:
#     - TableStatus:
#     And:
#     0 - Table is active.
#     1 - If it fails.
#####
function dynamodb_describe_table {
    local table_name
    local option OPTARG # Required to use getopt command in a function.

    #####
    # Function usage explanation
    #####
    function usage() {
        echo "function dynamodb_describe_table"
        echo "Describe the status of a DynamoDB table."
        echo "  -n table_name  -- The name of the table."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "n:h" option; do
        case "${option}" in
            n) table_name="${OPTARG}" ;;
            h)
                usage
        esac
    done
}
```

```

        return 0
        ;;
    \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
done
export OPTIND=1

if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
fi

local table_status
table_status=$(
    aws dynamodb describe-table \
        --table-name "$table_name" \
        --output text \
        --query 'Table.TableStatus'
)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log "$error_code"
    errecho "ERROR: AWS reports describe-table operation failed.$table_status"
    return 1
fi

echo "$table_status"

return 0
}

```

The utility functions used in this example.

```

#####
# function errecho

```

```

#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-return-codes.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}

```

- For API details, see [DescribeTable](#) in *AWS CLI Command Reference*.

GetItem

The following code example shows how to use GetItem.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function dynamodb_get_item
#
# This function gets an item from a DynamoDB table.
#
# Parameters:
#   -n table_name  -- The name of the table.
#   -k keys       -- Path to json file containing the keys that identify the item to
#                   get.
#   [-q query]    -- Optional JMESPath query expression.
#
# Returns:
#   The item as text output.
#
# And:
#   0 - If successful.
#   1 - If it fails.
#####
function dynamodb_get_item() {
    local table_name keys query response
    local option OPTARG # Required to use getopt command in a function.

    # #####
    # Function usage explanation
    #####
    function usage() {
        echo "function dynamodb_get_item"
```

```
    echo "Get an item from a DynamoDB table."
    echo " -n table_name -- The name of the table."
    echo " -k keys -- Path to json file containing the keys that identify the item
to get."
    echo " [-q query] -- Optional JMESPath query expression."
    echo ""
}
query=""
while getopts "n:k:q:h" option; do
    case "${option}" in
        n) table_name="${OPTARG}" ;;
        k) keys="${OPTARG}" ;;
        q) query="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$keys" ]]; then
    errecho "ERROR: You must provide a keys json file path the -k parameter."
    usage
    return 1
fi

if [[ -n "$query" ]]; then
    response=$(aws dynamodb get-item \
        --table-name "$table_name" \
        --key file://"${keys}" \
        --output text \
        --query "$query")
```



```

else
  response=$(
    aws dynamodb get-item \
      --table-name "$table_name" \
      --key file://"keys" \
      --output text
  )
fi

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports get-item operation failed.$response"
  return 1
fi

if [[ -n "$query" ]]; then
  echo "$response" | sed "/^\t/s/\t//1" # Remove initial tab that the JMSEPath
query inserts on some strings.
else
  echo "$response"
fi

return 0
}

```

The utility functions used in this example.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.

```

```

#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-
return-codes.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}

```

- For API details, see [GetItem](#) in *AWS CLI Command Reference*.

ListTables

The following code example shows how to use ListTables.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function dynamodb_list_tables
#
# This function lists all the tables in a DynamoDB.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function dynamodb_list_tables() {
    response=$(aws dynamodb list-tables \
        --output text \
        --query "TableNames")

    local error_code=${?}

    if [[ $error_code -ne 0 ]]; then
        aws_cli_error_log $error_code
        errecho "ERROR: AWS reports batch-write-item operation failed.$response"
        return 1
    fi

    echo "$response" | tr -s "[:space:]" "\n"

    return 0
}
```

The utility functions used in this example.

```
#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
```

```
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-
return-codes.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}

```

- For API details, see [ListTables](#) in *AWS CLI Command Reference*.

PutItem

The following code example shows how to use PutItem.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function dynamodb_put_item
#
# This function puts an item into a DynamoDB table.
#
# Parameters:
#     -n table_name -- The name of the table.
#     -i item -- Path to json file containing the item values.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function dynamodb_put_item() {
    local table_name item response
    local option OPTARG # Required to use getopt command in a function.

    #####
    # Function usage explanation
    #####
    function usage() {
        echo "function dynamodb_put_item"
        echo "Put an item into a DynamoDB table."
        echo " -n table_name -- The name of the table."
        echo " -i item -- Path to json file containing the item values."
        echo ""
    }

    while getopt "n:i:h" option; do
        case "${option}" in
            n) table_name="${OPTARG}" ;;

```

```
i) item="${OPTARG}" ;;
h)
    usage
    return 0
    ;;
\?)
    echo "Invalid parameter"
    usage
    return 1
    ;;
esac
done
export OPTIND=1

if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$item" ]]; then
    errecho "ERROR: You must provide an item with the -i parameter."
    usage
    return 1
fi

iecho "Parameters:\n"
iecho "    table_name:  $table_name"
iecho "    item:        $item"
iecho ""
iecho ""

response=$(aws dynamodb put-item \
    --table-name "$table_name" \
    --item file://"${item}")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports put-item operation failed.$response"
    return 1
fi
```

```
    return 0
}
```

The utility functions used in this example.

```
#####
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#####
function iecho() {
    if [[ $VERBOSE == true ]]; then
        echo "$@"
    fi
}

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-return-codes.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
```

```
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho " One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho " Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho " Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho " Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho " The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho " The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho " 255 is a catch-all error."
  fi

  return 0
}
```

- For API details, see [PutItem](#) in *AWS CLI Command Reference*.

Query

The following code example shows how to use Query.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function dynamodb_query
#
# This function queries a DynamoDB table.
#
```



```

# Parameters:
#     -n table_name -- The name of the table.
#     -k key_condition_expression -- The key condition expression.
#     -a attribute_names -- Path to JSON file containing the attribute names.
#     -v attribute_values -- Path to JSON file containing the attribute values.
#     [-p projection_expression] -- Optional projection expression.
#
# Returns:
#     The items as json output.
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function dynamodb_query() {
    local table_name key_condition_expression attribute_names attribute_values
    projection_expression response
    local option OPTARG # Required to use getopt command in a function.

    # #####
    # Function usage explanation
    #####
    function usage() {
        echo "function dynamodb_query"
        echo "Query a DynamoDB table."
        echo " -n table_name -- The name of the table."
        echo " -k key_condition_expression -- The key condition expression."
        echo " -a attribute_names -- Path to JSON file containing the attribute names."
        echo " -v attribute_values -- Path to JSON file containing the attribute
values."
        echo " [-p projection_expression] -- Optional projection expression."
        echo ""
    }

    while getopt "n:k:a:v:p:h" option; do
        case "${option}" in
            n) table_name="${OPTARG}" ;;
            k) key_condition_expression="${OPTARG}" ;;
            a) attribute_names="${OPTARG}" ;;
            v) attribute_values="${OPTARG}" ;;
            p) projection_expression="${OPTARG}" ;;
            h)
                usage
                return 0
            ;;
        esac
    done
}

```

```
\?)
    echo "Invalid parameter"
    usage
    return 1
    ;;
esac
done
export OPTIND=1

if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$key_condition_expression" ]]; then
    errecho "ERROR: You must provide a key condition expression with the -k
parameter."
    usage
    return 1
fi

if [[ -z "$attribute_names" ]]; then
    errecho "ERROR: You must provide a attribute names with the -a parameter."
    usage
    return 1
fi

if [[ -z "$attribute_values" ]]; then
    errecho "ERROR: You must provide a attribute values with the -v parameter."
    usage
    return 1
fi

if [[ -z "$projection_expression" ]]; then
    response=$(aws dynamodb query \
        --table-name "$table_name" \
        --key-condition-expression "$key_condition_expression" \
        --expression-attribute-names file://"$attribute_names" \
        --expression-attribute-values file://"$attribute_values")
else
    response=$(aws dynamodb query \
        --table-name "$table_name" \
        --key-condition-expression "$key_condition_expression" \
```

```

    --expression-attribute-names file://"$attribute_names" \
    --expression-attribute-values file://"$attribute_values" \
    --projection-expression "$projection_expression")
fi

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports query operation failed.$response"
    return 1
fi

echo "$response"

return 0
}

```

The utility functions used in this example.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-return-codes.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.

```

```
#
#####
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho " One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho " Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho " Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho " Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho " The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho " The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho " 255 is a catch-all error."
  fi

  return 0
}
```

- For API details, see [Query](#) in *AWS CLI Command Reference*.

Scan

The following code example shows how to use Scan.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function dynamodb_scan
#
```

```

# This function scans a DynamoDB table.
#
# Parameters:
#   -n table_name -- The name of the table.
#   -f filter_expression -- The filter expression.
#   -a expression_attribute_names -- Path to JSON file containing the expression
#   attribute names.
#   -v expression_attribute_values -- Path to JSON file containing the
#   expression attribute values.
#   [-p projection_expression] -- Optional projection expression.
#
# Returns:
#   The items as json output.
# And:
#   0 - If successful.
#   1 - If it fails.
#####
function dynamodb_scan() {
    local table_name filter_expression expression_attribute_names
    expression_attribute_values projection_expression response
    local option OPTARG # Required to use getopt command in a function.

    # #####
    # Function usage explanation
    # #####
    function usage() {
        echo "function dynamodb_scan"
        echo "Scan a DynamoDB table."
        echo " -n table_name -- The name of the table."
        echo " -f filter_expression -- The filter expression."
        echo " -a expression_attribute_names -- Path to JSON file containing the
        expression attribute names."
        echo " -v expression_attribute_values -- Path to JSON file containing the
        expression attribute values."
        echo " [-p projection_expression] -- Optional projection expression."
        echo ""
    }

    while getopt "n:f:a:v:p:h" option; do
        case "${option}" in
            n) table_name="${OPTARG}" ;;
            f) filter_expression="${OPTARG}" ;;
            a) expression_attribute_names="${OPTARG}" ;;
            v) expression_attribute_values="${OPTARG}" ;;

```

```
p) projection_expression="${OPTARG}" ;;
h)
    usage
    return 0
    ;;
\?)
    echo "Invalid parameter"
    usage
    return 1
    ;;
esac
done
export OPTIND=1

if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$filter_expression" ]]; then
    errecho "ERROR: You must provide a filter expression with the -f parameter."
    usage
    return 1
fi

if [[ -z "$expression_attribute_names" ]]; then
    errecho "ERROR: You must provide expression attribute names with the -a
parameter."
    usage
    return 1
fi

if [[ -z "$expression_attribute_values" ]]; then
    errecho "ERROR: You must provide expression attribute values with the -v
parameter."
    usage
    return 1
fi

if [[ -z "$projection_expression" ]]; then
    response=$(aws dynamodb scan \
        --table-name "$table_name" \
        --filter-expression "$filter_expression" \
```

```

        --expression-attribute-names file://"$expression_attribute_names" \
        --expression-attribute-values file://"$expression_attribute_values")
else
    response=$(aws dynamodb scan \
        --table-name "$table_name" \
        --filter-expression "$filter_expression" \
        --expression-attribute-names file://"$expression_attribute_names" \
        --expression-attribute-values file://"$expression_attribute_values" \
        --projection-expression "$projection_expression")
fi

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports scan operation failed.$response"
    return 1
fi

echo "$response"

return 0
}

```

The utility functions used in this example.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-return-codes.

```

```

#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}

```

- For API details, see [Scan](#) in *AWS CLI Command Reference*.

UpdateItem

The following code example shows how to use UpdateItem.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function dynamodb_update_item
#
# This function updates an item in a DynamoDB table.
#
#
# Parameters:
#   -n table_name  -- The name of the table.
#   -k keys       -- Path to json file containing the keys that identify the item to
#                   update.
#   -e update expression  -- An expression that defines one or more attributes
#                   to be updated.
#   -v values     -- Path to json file containing the update values.
#
# Returns:
#   0 - If successful.
#   1 - If it fails.
#####
function dynamodb_update_item() {
    local table_name keys update_expression values response
    local option OPTARG # Required to use getopt command in a function.

    #####
    # Function usage explanation
    #####
    function usage() {
        echo "function dynamodb_update_item"
        echo "Update an item in a DynamoDB table."
        echo " -n table_name  -- The name of the table."
        echo " -k keys       -- Path to json file containing the keys that identify the item
to update."
        echo " -e update expression  -- An expression that defines one or more
attributes to be updated."
        echo " -v values     -- Path to json file containing the update values."
    }
}
```

```
    echo ""
}

while getopts "n:k:e:v:h" option; do
    case "${option}" in
        n) table_name="${OPTARG}" ;;
        k) keys="${OPTARG}" ;;
        e) update_expression="${OPTARG}" ;;
        v) values="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$keys" ]]; then
    errecho "ERROR: You must provide a keys json file path the -k parameter."
    usage
    return 1
fi

if [[ -z "$update_expression" ]]; then
    errecho "ERROR: You must provide an update expression with the -e parameter."
    usage
    return 1
fi

if [[ -z "$values" ]]; then
    errecho "ERROR: You must provide a values json file path the -v parameter."
    usage
    return 1
fi
```

```

iecho "Parameters:\n"
iecho "  table_name:  $table_name"
iecho "  keys:  $keys"
iecho "  update_expression:  $update_expression"
iecho "  values:  $values"

response=$(aws dynamodb update-item \
  --table-name "$table_name" \
  --key file://" $keys" \
  --update-expression "$update_expression" \
  --expression-attribute-values file://" $values")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports update-item operation failed.$response"
  return 1
fi

return 0
}

```

The utility functions used in this example.

```

#####
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#####
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

#####
# function errecho
#

```

```

# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-return-codes.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}

```

- For API details, see [UpdateItem](#) in *AWS CLI Command Reference*.

Scenarios

Get started with tables, items, and queries

The following code example shows how to:

- Create a table that can hold movie data.
- Put, get, and update a single movie in the table.
- Write movie data to the table from a sample JSON file.
- Query for movies that were released in a given year.
- Scan for movies that were released in a range of years.
- Delete a movie from the table, then delete the table.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

The DynamoDB getting started scenario.

```
#####
# function dynamodb_getting_started_movies
#
# Scenario to create an Amazon DynamoDB table and perform a series of operations on
# the table.
#
# Returns:
#     0 - If successful.
#     1 - If an error occurred.
#####
function dynamodb_getting_started_movies() {

    source ./dynamodb_operations.sh
```

```
key_schema_json_file="dynamodb_key_schema.json"
attribute_definitions_json_file="dynamodb_attr_def.json"
item_json_file="movie_item.json"
key_json_file="movie_key.json"
batch_json_file="batch.json"
attribute_names_json_file="attribute_names.json"
attributes_values_json_file="attribute_values.json"

echo_repeat "*" 88
echo
echo "Welcome to the Amazon DynamoDB getting started demo."
echo
echo_repeat "*" 88
echo

local table_name
echo -n "Enter a name for a new DynamoDB table: "
get_input
table_name=$get_input_result

local provisioned_throughput="ReadCapacityUnits=5,WriteCapacityUnits=5"

echo '['
{"AttributeName": "year", "KeyType": "HASH"},
 {"AttributeName": "title", "KeyType": "RANGE"}
]' >"$key_schema_json_file"

echo '['
{"AttributeName": "year", "AttributeType": "N"},
 {"AttributeName": "title", "AttributeType": "S"}
]' >"$attribute_definitions_json_file"

if dynamodb_create_table -n "$table_name" -a "$attribute_definitions_json_file" \
  -k "$key_schema_json_file" -p "$provisioned_throughput" 1>/dev/null; then
  echo "Created a DynamoDB table named $table_name"
else
  errecho "The table failed to create. This demo will exit."
  clean_up
  return 1
fi

echo "Waiting for the table to become active...."

if dynamodb_wait_table_active -n "$table_name"; then
```

```
    echo "The table is now active."
else
    errecho "The table failed to become active. This demo will exit."
    cleanup "$table_name"
    return 1
fi

echo
echo_repeat "*" 88
echo

echo -n "Enter the title of a movie you want to add to the table: "
get_input
local added_title
added_title=$get_input_result

local added_year
get_int_input "What year was it released? "
added_year=$get_input_result

local rating
get_float_input "On a scale of 1 - 10, how do you rate it? " "1" "10"
rating=$get_input_result

local plot
echo -n "Summarize the plot for me: "
get_input
plot=$get_input_result

echo '{
  "year": {"N" : ""$added_year""},
  "title": {"S" : ""$added_title""},
  "info": {"M" : {"plot": {"S" : ""$plot""}, "rating": {"N" : ""$rating""} } }
}' >"$item_json_file"

if dynamodb_put_item -n "$table_name" -i "$item_json_file"; then
    echo "The movie '$added_title' was successfully added to the table
'$table_name'."
else
    errecho "Put item failed. This demo will exit."
    clean_up "$table_name"
    return 1
fi
```

```
echo
echo_repeat "*" 88
echo

echo "Let's update your movie '$added_title'."
get_float_input "You rated it $rating, what new rating would you give it? " "1"
"10"
rating=$get_input_result

echo -n "You summarized the plot as '$plot'."
echo "What would you say now? "
get_input
plot=$get_input_result

echo '{
  "year": {"N" : ""$added_year""},
  "title": {"S" : ""$added_title""}
}' >"$key_json_file"

echo '{
  "r": {"N" : ""$rating""},
  "p": {"S" : ""$plot""}
}' >"$item_json_file"

local update_expression="SET info.rating = :r, info.plot = :p"

if dynamodb_update_item -n "$table_name" -k "$key_json_file" -e
"$update_expression" -v "$item_json_file"; then
  echo "Updated '$added_title' with new attributes."
else
  errecho "Update item failed. This demo will exit."
  clean_up "$table_name"
  return 1
fi

echo
echo_repeat "*" 88
echo

echo "We will now use batch write to upload 150 movie entries into the table."

local batch_json
for batch_json in movie_files/movies_*.json; do
  echo "{ \"$table_name\" : $(<"$batch_json") }" >"$batch_json_file"
```



```

    if dynamodb_batch_write_item -i "$batch_json_file" 1>/dev/null; then
        echo "Entries in $batch_json added to table."
    else
        errecho "Batch write failed. This demo will exit."
        clean_up "$table_name"
        return 1
    fi
done

local title="The Lord of the Rings: The Fellowship of the Ring"
local year="2001"

if get_yes_no_input "Let's move on...do you want to get info about '$title'? (y/n)
"; then
    echo '{
"year": {"N" : ""$year""},
"title": {"S" : ""$title""}
}' >"$key_json_file"
    local info
    info=$(dynamodb_get_item -n "$table_name" -k "$key_json_file")

    # shellcheck disable=SC2181
    if [[ ${?} -ne 0 ]]; then
        errecho "Get item failed. This demo will exit."
        clean_up "$table_name"
        return 1
    fi

    echo "Here is what I found:"
    echo "$info"
fi

local ask_for_year=true
while [[ "$ask_for_year" == true ]]; do
    echo "Let's get a list of movies released in a given year."
    get_int_input "Enter a year between 1972 and 2018: " "1972" "2018"
    year=$get_input_result
    echo '{
"#n": "year"
}' >"$attribute_names_json_file"

    echo '{
":v": {"N" : ""$year""}
}' >"$attributes_values_json_file"

```

```
response=$(dynamodb_query -n "$table_name" -k "#n=:v" -a
"$attribute_names_json_file" -v "$attributes_values_json_file")

# shellcheck disable=SC2181
if [[ ${?} -ne 0 ]]; then
    errecho "Query table failed. This demo will exit."
    clean_up "$table_name"
    return 1
fi

echo "Here is what I found:"
echo "$response"

if ! get_yes_no_input "Try another year? (y/n) "; then
    ask_for_year=false
fi
done

echo "Now let's scan for movies released in a range of years. Enter a year: "
get_int_input "Enter a year between 1972 and 2018: " "1972" "2018"
local start=$get_input_result

get_int_input "Enter another year: " "1972" "2018"
local end=$get_input_result

echo '{
  "#n": "year"
}' >"$attribute_names_json_file"

echo '{
  ":v1": {"N" : ""$start""},
  ":v2": {"N" : ""$end""}
}' >"$attributes_values_json_file"

response=$(dynamodb_scan -n "$table_name" -f "#n BETWEEN :v1 AND :v2" -a
"$attribute_names_json_file" -v "$attributes_values_json_file")

# shellcheck disable=SC2181
if [[ ${?} -ne 0 ]]; then
    errecho "Scan table failed. This demo will exit."
    clean_up "$table_name"
    return 1
fi
```

```

echo "Here is what I found:"
echo "$response"

echo
echo_repeat "*" 88
echo

echo "Let's remove your movie '$added_title' from the table."

if get_yes_no_input "Do you want to remove '$added_title'? (y/n) "; then
  echo '{
"year": {"N" : ""$added_year""},
"title": {"S" : ""$added_title""}
}' >"$key_json_file"

  if ! dynamodb_delete_item -n "$table_name" -k "$key_json_file"; then
    errecho "Delete item failed. This demo will exit."
    clean_up "$table_name"
    return 1
  fi
fi

if get_yes_no_input "Do you want to delete the table '$table_name'? (y/n) "; then
  if ! clean_up "$table_name"; then
    return 1
  fi
else
  if ! clean_up; then
    return 1
  fi
fi

return 0
}

```

The DynamoDB functions used in this scenario.

```

#####
# function dynamodb_create_table
#
# This function creates an Amazon DynamoDB table.

```

```

#
# Parameters:
#   -n table_name -- The name of the table to create.
#   -a attribute_definitions -- JSON file path of a list of attributes and their
types.
#   -k key_schema -- JSON file path of a list of attributes and their key types.
#   -p provisioned_throughput -- Provisioned throughput settings for the table.
#
# Returns:
#   0 - If successful.
#   1 - If it fails.
#####
function dynamodb_create_table() {
    local table_name attribute_definitions key_schema provisioned_throughput response
    local option OPTARG # Required to use getopt command in a function.

#####
# Function usage explanation
#####
function usage() {
    echo "function dynamodb_create_table"
    echo "Creates an Amazon DynamoDB table."
    echo " -n table_name -- The name of the table to create."
    echo " -a attribute_definitions -- JSON file path of a list of attributes and
their types."
    echo " -k key_schema -- JSON file path of a list of attributes and their key
types."
    echo " -p provisioned_throughput -- Provisioned throughput settings for the
table."
    echo ""
}

# Retrieve the calling parameters.
while getopt "n:a:k:p:h" option; do
    case "${option}" in
        n) table_name="${OPTARG}" ;;
        a) attribute_definitions="${OPTARG}" ;;
        k) key_schema="${OPTARG}" ;;
        p) provisioned_throughput="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)

```

```
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
done
export OPTIND=1

if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$attribute_definitions" ]]; then
    errecho "ERROR: You must provide an attribute definitions json file path the -a
parameter."
    usage
    return 1
fi

if [[ -z "$key_schema" ]]; then
    errecho "ERROR: You must provide a key schema json file path the -k parameter."
    usage
    return 1
fi

if [[ -z "$provisioned_throughput" ]]; then
    errecho "ERROR: You must provide a provisioned throughput json file path the -p
parameter."
    usage
    return 1
fi

iecho "Parameters:\n"
iecho "    table_name:  $table_name"
iecho "    attribute_definitions:  $attribute_definitions"
iecho "    key_schema:  $key_schema"
iecho "    provisioned_throughput:  $provisioned_throughput"
iecho ""

response=$(aws dynamodb create-table \
    --table-name "$table_name" \
    --attribute-definitions file://"$attribute_definitions" \
```

```

--key-schema file://"key_schema" \
--provisioned-throughput "$provisioned_throughput")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-table operation failed.$response"
    return 1
fi

return 0
}

#####
# function dynamodb_describe_table
#
# This function returns the status of a DynamoDB table.
#
# Parameters:
#     -n table_name  -- The name of the table.
#
# Response:
#     - TableStatus:
#     And:
#     0 - Table is active.
#     1 - If it fails.
#####
function dynamodb_describe_table {
    local table_name
    local option OPTARG # Required to use getopt command in a function.

    #####
    # Function usage explanation
    #####
    function usage() {
        echo "function dynamodb_describe_table"
        echo "Describe the status of a DynamoDB table."
        echo " -n table_name  -- The name of the table."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "n:h" option; do

```

```

case "${option}" in
  n) table_name="${OPTARG}" ;;
  h)
    usage
    return 0
    ;;
  \?)
    echo "Invalid parameter"
    usage
    return 1
    ;;
esac
done
export OPTIND=1

if [[ -z "$table_name" ]]; then
  errecho "ERROR: You must provide a table name with the -n parameter."
  usage
  return 1
fi

local table_status
table_status=$(
  aws dynamodb describe-table \
    --table-name "$table_name" \
    --output text \
    --query 'Table.TableStatus'
)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log "$error_code"
  errecho "ERROR: AWS reports describe-table operation failed.$table_status"
  return 1
fi

echo "$table_status"

return 0
}

#####
# function dynamodb_put_item

```

```

#
# This function puts an item into a DynamoDB table.
#
# Parameters:
#     -n table_name  -- The name of the table.
#     -i item        -- Path to json file containing the item values.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function dynamodb_put_item() {
    local table_name item response
    local option OPTARG # Required to use getopt command in a function.

    #####
    # Function usage explanation
    #####
    function usage() {
        echo "function dynamodb_put_item"
        echo "Put an item into a DynamoDB table."
        echo " -n table_name  -- The name of the table."
        echo " -i item        -- Path to json file containing the item values."
        echo ""
    }

    while getopt "n:i:h" option; do
        case "${option}" in
            n) table_name="${OPTARG}" ;;
            i) item="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    if [[ -z "$table_name" ]]; then

```



```

    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$item" ]]; then
    errecho "ERROR: You must provide an item with the -i parameter."
    usage
    return 1
fi

iecho "Parameters:\n"
iecho "    table_name:  $table_name"
iecho "    item:        $item"
iecho ""
iecho ""

response=$(aws dynamodb put-item \
    --table-name "$table_name" \
    --item file://" $item")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports put-item operation failed.$response"
    return 1
fi

return 0
}

#####
# function dynamodb_update_item
#
# This function updates an item in a DynamoDB table.
#
# Parameters:
#     -n table_name  -- The name of the table.
#     -k keys        -- Path to json file containing the keys that identify the item to
# update.

```

```

#     -e update expression  -- An expression that defines one or more attributes
to be updated.
#     -v values  -- Path to json file containing the update values.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function dynamodb_update_item() {
    local table_name keys update_expression values response
    local option OPTARG # Required to use getopt command in a function.

#####
# Function usage explanation
#####
function usage() {
    echo "function dynamodb_update_item"
    echo "Update an item in a DynamoDB table."
    echo " -n table_name  -- The name of the table."
    echo " -k keys  -- Path to json file containing the keys that identify the item
to update."
    echo " -e update expression  -- An expression that defines one or more
attributes to be updated."
    echo " -v values  -- Path to json file containing the update values."
    echo ""
}

while getopt "n:k:e:v:h" option; do
    case "${option}" in
        n) table_name="${OPTARG}" ;;
        k) keys="${OPTARG}" ;;
        e) update_expression="${OPTARG}" ;;
        v) values="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done

```

```
export OPTIND=1

if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$keys" ]]; then
    errecho "ERROR: You must provide a keys json file path the -k parameter."
    usage
    return 1
fi

if [[ -z "$update_expression" ]]; then
    errecho "ERROR: You must provide an update expression with the -e parameter."
    usage
    return 1
fi

if [[ -z "$values" ]]; then
    errecho "ERROR: You must provide a values json file path the -v parameter."
    usage
    return 1
fi

iecho "Parameters:\n"
iecho "  table_name:  $table_name"
iecho "  keys:        $keys"
iecho "  update_expression:  $update_expression"
iecho "  values:      $values"

response=$(aws dynamodb update-item \
  --table-name "$table_name" \
  --key file://" $keys" \
  --update-expression "$update_expression" \
  --expression-attribute-values file://" $values")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports update-item operation failed.$response"
    return 1
fi
```

```

return 0

}

#####
# function dynamodb_batch_write_item
#
# This function writes a batch of items into a DynamoDB table.
#
# Parameters:
#     -i item  -- Path to json file containing the items to write.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function dynamodb_batch_write_item() {
    local item response
    local option OPTARG # Required to use getopt command in a function.

    #####
    # Function usage explanation
    #####
    function usage() {
        echo "function dynamodb_batch_write_item"
        echo "Write a batch of items into a DynamoDB table."
        echo " -i item  -- Path to json file containing the items to write."
        echo ""
    }
    while getopt "i:h" option; do
        case "${option}" in
            i) item="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done

```

```

export OPTIND=1

if [[ -z "$item" ]]; then
    errecho "ERROR: You must provide an item with the -i parameter."
    usage
    return 1
fi

iecho "Parameters:\n"
iecho "    table_name:  $table_name"
iecho "    item:        $item"
iecho ""

response=$(aws dynamodb batch-write-item \
    --request-items file://"$item")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports batch-write-item operation failed.$response"
    return 1
fi

return 0
}

#####
# function dynamodb_get_item
#
# This function gets an item from a DynamoDB table.
#
# Parameters:
#     -n table_name -- The name of the table.
#     -k keys      -- Path to json file containing the keys that identify the item to
# get.
#     [-q query]  -- Optional JMESPath query expression.
#
# Returns:
#     The item as text output.
#
# And:
#     0 - If successful.
#     1 - If it fails.
#####

```

```

function dynamodb_get_item() {
  local table_name keys query response
  local option OPTARG # Required to use getopt command in a function.

  # #####
  # Function usage explanation
  # #####
  function usage() {
    echo "function dynamodb_get_item"
    echo "Get an item from a DynamoDB table."
    echo " -n table_name -- The name of the table."
    echo " -k keys -- Path to json file containing the keys that identify the item
to get."
    echo " [-q query] -- Optional JMESPath query expression."
    echo ""
  }
  query=""
  while getopt "n:k:q:h" option; do
    case "${option}" in
      n) table_name="${OPTARG}" ;;
      k) keys="${OPTARG}" ;;
      q) query="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$keys" ]]; then
    errecho "ERROR: You must provide a keys json file path the -k parameter."
    usage
  fi
}

```

```

    return 1
fi

if [[ -n "$query" ]]; then
    response=$(aws dynamodb get-item \
        --table-name "$table_name" \
        --key file://" $keys" \
        --output text \
        --query "$query")
else
    response=$(
        aws dynamodb get-item \
            --table-name "$table_name" \
            --key file://" $keys" \
            --output text
    )
fi

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports get-item operation failed.$response"
    return 1
fi

if [[ -n "$query" ]]; then
    echo "$response" | sed "/^\t/s/\t//1" # Remove initial tab that the JMSEPath
query inserts on some strings.
else
    echo "$response"
fi

return 0
}

#####
# function dynamodb_query
#
# This function queries a DynamoDB table.
#
# Parameters:
#     -n table_name -- The name of the table.
#     -k key_condition_expression -- The key condition expression.

```

```

#     -a attribute_names -- Path to JSON file containing the attribute names.
#     -v attribute_values -- Path to JSON file containing the attribute values.
#     [-p projection_expression] -- Optional projection expression.
#
# Returns:
#     The items as json output.
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function dynamodb_query() {
    local table_name key_condition_expression attribute_names attribute_values
    projection_expression response
    local option OPTARG # Required to use getopt command in a function.

    # #####
    # Function usage explanation
    # #####
    function usage() {
        echo "function dynamodb_query"
        echo "Query a DynamoDB table."
        echo " -n table_name -- The name of the table."
        echo " -k key_condition_expression -- The key condition expression."
        echo " -a attribute_names -- Path to JSON file containing the attribute names."
        echo " -v attribute_values -- Path to JSON file containing the attribute
values."
        echo " [-p projection_expression] -- Optional projection expression."
        echo ""
    }

    while getopt "n:k:a:v:p:h" option; do
        case "${option}" in
            n) table_name="${OPTARG}" ;;
            k) key_condition_expression="${OPTARG}" ;;
            a) attribute_names="${OPTARG}" ;;
            v) attribute_values="${OPTARG}" ;;
            p) projection_expression="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage

```



```
        return 1
        ;;
    esac
done
export OPTIND=1

if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$key_condition_expression" ]]; then
    errecho "ERROR: You must provide a key condition expression with the -k
parameter."
    usage
    return 1
fi

if [[ -z "$attribute_names" ]]; then
    errecho "ERROR: You must provide a attribute names with the -a parameter."
    usage
    return 1
fi

if [[ -z "$attribute_values" ]]; then
    errecho "ERROR: You must provide a attribute values with the -v parameter."
    usage
    return 1
fi

if [[ -z "$projection_expression" ]]; then
    response=$(aws dynamodb query \
        --table-name "$table_name" \
        --key-condition-expression "$key_condition_expression" \
        --expression-attribute-names file://"$attribute_names" \
        --expression-attribute-values file://"$attribute_values")
else
    response=$(aws dynamodb query \
        --table-name "$table_name" \
        --key-condition-expression "$key_condition_expression" \
        --expression-attribute-names file://"$attribute_names" \
        --expression-attribute-values file://"$attribute_values" \
        --projection-expression "$projection_expression")
```

```

fi

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports query operation failed.$response"
    return 1
fi

echo "$response"

return 0
}

#####
# function dynamodb_scan
#
# This function scans a DynamoDB table.
#
# Parameters:
#     -n table_name -- The name of the table.
#     -f filter_expression -- The filter expression.
#     -a expression_attribute_names -- Path to JSON file containing the expression
attribute names.
#     -v expression_attribute_values -- Path to JSON file containing the
expression attribute values.
#     [-p projection_expression] -- Optional projection expression.
#
# Returns:
#     The items as json output.
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function dynamodb_scan() {
    local table_name filter_expression expression_attribute_names
expression_attribute_values projection_expression response
    local option OPTARG # Required to use getopt command in a function.

    # #####
    # Function usage explanation
    #####
    function usage() {

```

```
    echo "function dynamodb_scan"
    echo "Scan a DynamoDB table."
    echo " -n table_name -- The name of the table."
    echo " -f filter_expression -- The filter expression."
    echo " -a expression_attribute_names -- Path to JSON file containing the
expression attribute names."
    echo " -v expression_attribute_values -- Path to JSON file containing the
expression attribute values."
    echo " [-p projection_expression] -- Optional projection expression."
    echo ""
}

while getopts "n:f:a:v:p:h" option; do
    case "${option}" in
        n) table_name="${OPTARG}" ;;
        f) filter_expression="${OPTARG}" ;;
        a) expression_attribute_names="${OPTARG}" ;;
        v) expression_attribute_values="${OPTARG}" ;;
        p) projection_expression="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$filter_expression" ]]; then
    errecho "ERROR: You must provide a filter expression with the -f parameter."
    usage
    return 1
fi
```

```
if [[ -z "$expression_attribute_names" ]]; then
    errecho "ERROR: You must provide expression attribute names with the -a
parameter."
    usage
    return 1
fi

if [[ -z "$expression_attribute_values" ]]; then
    errecho "ERROR: You must provide expression attribute values with the -v
parameter."
    usage
    return 1
fi

if [[ -z "$projection_expression" ]]; then
    response=$(aws dynamodb scan \
        --table-name "$table_name" \
        --filter-expression "$filter_expression" \
        --expression-attribute-names file://"$expression_attribute_names" \
        --expression-attribute-values file://"$expression_attribute_values")
else
    response=$(aws dynamodb scan \
        --table-name "$table_name" \
        --filter-expression "$filter_expression" \
        --expression-attribute-names file://"$expression_attribute_names" \
        --expression-attribute-values file://"$expression_attribute_values" \
        --projection-expression "$projection_expression")
fi

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports scan operation failed.$response"
    return 1
fi

echo "$response"

return 0
}

#####
# function dynamodb_delete_item
```

```

#
# This function deletes an item from a DynamoDB table.
#
# Parameters:
#     -n table_name  -- The name of the table.
#     -k keys       -- Path to json file containing the keys that identify the item to
#                    delete.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function dynamodb_delete_item() {
    local table_name keys response
    local option OPTARG # Required to use getopt command in a function.

    # #####
    # Function usage explanation
    # #####
    function usage() {
        echo "function dynamodb_delete_item"
        echo "Delete an item from a DynamoDB table."
        echo " -n table_name  -- The name of the table."
        echo " -k keys       -- Path to json file containing the keys that identify the item
to delete."
        echo ""
    }
    while getopt "n:k:h" option; do
        case "${option}" in
            n) table_name="${OPTARG}" ;;
            k) keys="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

```

```

if [[ -z "$table_name" ]]; then
    errecho "ERROR: You must provide a table name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$keys" ]]; then
    errecho "ERROR: You must provide a keys json file path the -k parameter."
    usage
    return 1
fi

iecho "Parameters:\n"
iecho "    table_name:  $table_name"
iecho "    keys:        $keys"
iecho ""

response=$(aws dynamodb delete-item \
    --table-name "$table_name" \
    --key file://" $keys")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-item operation failed.$response"
    return 1
fi

return 0
}

#####
# function dynamodb_delete_table
#
# This function deletes a DynamoDB table.
#
# Parameters:
#     -n table_name  -- The name of the table to delete.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.

```

```
#####
function dynamodb_delete_table() {
    local table_name response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function dynamodb_delete_table"
        echo "Deletes an Amazon DynamoDB table."
        echo " -n table_name -- The name of the table to delete."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "n:h" option; do
        case "${option}" in
            n) table_name="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    if [[ -z "$table_name" ]]; then
        errecho "ERROR: You must provide a table name with the -n parameter."
        usage
        return 1
    fi

    iecho "Parameters:\n"
    iecho "    table_name:  $table_name"
    iecho ""

    response=$(aws dynamodb delete-table \
        --table-name "$table_name")

    local error_code=${?}
}
```

```

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-table operation failed.$response"
    return 1
fi

return 0
}

```

The utility functions used in this scenario.

```

#####
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#####
function iecho() {
    if [[ $VERBOSE == true ]]; then
        echo "$@"
    fi
}

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# See https://docs.aws.amazon.com/cli/latest/topic/return-codes.html#cli-aws-help-return-codes.
#
# The function expects the following argument:

```



```
#      $1 - The error code returned by the AWS CLI.
#
# Returns:
#      0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}
```

- For API details, see the following topics in *AWS CLI Command Reference*.
 - [BatchWriteItem](#)
 - [CreateTable](#)
 - [DeleteItem](#)
 - [DeleteTable](#)
 - [DescribeTable](#)
 - [GetItem](#)
 - [PutItem](#)
 - [Query](#)
 - [Scan](#)

- [UpdateItem](#)

Amazon EC2 examples using AWS CLI with Bash script

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Bash script with Amazon EC2.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)
- [Scenarios](#)

Actions

AllocateAddress

The following code example shows how to use AllocateAddress.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####  
# function ec2_allocate_address  
#
```

```

# This function allocates an Elastic IP address for use with Amazon Elastic Compute
  Cloud (Amazon EC2) instances in a specific AWS Region.
#
# Parameters:
#     -d domain - The domain for the Elastic IP address (either 'vpc' or
  'standard').
#
# Returns:
#     The allocated Elastic IP address, or an error message if the operation
  fails.
# And:
#     0 - If successful.
#     1 - If it fails.
#
#####
function ec2_allocate_address() {
    local domain response

    # Function to display usage information
    function usage() {
        echo "function ec2_allocate_address"
        echo "Allocates an Elastic IP address for use with Amazon Elastic Compute Cloud
  (Amazon EC2) instances in a specific AWS Region."
        echo "  -d domain - The domain for the Elastic IP address (either 'vpc' or
  'standard')."
        echo ""
    }

    # Parse the command-line arguments
    while getopts "d:h" option; do
        case "${option}" in
            d) domain="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

```

```

# Validate the input parameters
if [[ -z "$domain" ]]; then
    errecho "ERROR: You must provide a domain with the -d parameter (either 'vpc' or
'standard')."
    return 1
fi

if [[ "$domain" != "vpc" && "$domain" != "standard" ]]; then
    errecho "ERROR: Invalid domain value. Must be either 'vpc' or 'standard'."
    return 1
fi

# Allocate the Elastic IP address
response=$(aws ec2 allocate-address \
    --domain "$domain" \
    --query "[PublicIp,AllocationId]" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports allocate-address operation failed."
    errecho "$response"
    return 1
}

echo "$response"
return 0
}

```

The utility functions used in this example.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#

```

```
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}
```

- For API details, see [AllocateAddress](#) in *AWS CLI Command Reference*.

AssociateAddress

The following code example shows how to use AssociateAddress.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function ec2_associate_address
#
# This function associates an Elastic IP address with an Amazon Elastic Compute
# Cloud (Amazon EC2) instance.
#
# Parameters:
#   -a allocation_id - The allocation ID of the Elastic IP address to associate.
#   -i instance_id - The ID of the EC2 instance to associate the Elastic IP
# address with.
#
# Returns:
#   0 - If successful.
#   1 - If it fails.
#
#####
function ec2_associate_address() {
    local allocation_id instance_id response

    # Function to display usage information
    function usage() {
        echo "function ec2_associate_address"
        echo "Associates an Elastic IP address with an Amazon Elastic Compute Cloud
(Amazon EC2) instance."
        echo "  -a allocation_id - The allocation ID of the Elastic IP address to
associate."
        echo "  -i instance_id - The ID of the EC2 instance to associate the Elastic IP
address with."
        echo ""
    }

    # Parse the command-line arguments
    while getopts "a:i:h" option; do
        case "${option}" in
```

```
a) allocation_id="${OPTARG}" ;;
i) instance_id="${OPTARG}" ;;
h)
    usage
    return 0
    ;;
\?)
    echo "Invalid parameter"
    usage
    return 1
    ;;
esac
done
export OPTIND=1

# Validate the input parameters
if [[ -z "$allocation_id" ]]; then
    errecho "ERROR: You must provide an allocation ID with the -a parameter."
    return 1
fi

if [[ -z "$instance_id" ]]; then
    errecho "ERROR: You must provide an instance ID with the -i parameter."
    return 1
fi

# Associate the Elastic IP address
response=$(aws ec2 associate-address \
    --allocation-id "$allocation_id" \
    --instance-id "$instance_id" \
    --query "AssociationId" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports associate-address operation failed."
    errecho "$response"
    return 1
}

echo "$response"
return 0
}
```

The utility functions used in this example.

```
#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}

```


- For API details, see [AssociateAddress](#) in *AWS CLI Command Reference*.

AuthorizeSecurityGroupIngress

The following code example shows how to use AuthorizeSecurityGroupIngress.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function ec2_authorize_security_group_ingress
#
# This function authorizes an ingress rule for an Amazon Elastic Compute Cloud
# (Amazon EC2) security group.
#
# Parameters:
#   -g security_group_id - The ID of the security group.
#   -i ip_address - The IP address or CIDR block to authorize.
#   -p protocol - The protocol to authorize (e.g., tcp, udp, icmp).
#   -f from_port - The start of the port range to authorize.
#   -t to_port - The end of the port range to authorize.
#
# And:
#   0 - If successful.
#   1 - If it fails.
#####
function ec2_authorize_security_group_ingress() {
    local security_group_id ip_address protocol from_port to_port response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_authorize_security_group_ingress"
        echo "Authorizes an ingress rule for an Amazon Elastic Compute Cloud (Amazon
        EC2) security group."
    }
}
```

```
    echo " -g security_group_id - The ID of the security group."
    echo " -i ip_address - The IP address or CIDR block to authorize."
    echo " -p protocol - The protocol to authorize (e.g., tcp, udp, icmp)."
    echo " -f from_port - The start of the port range to authorize."
    echo " -t to_port - The end of the port range to authorize."
    echo ""
}

# Retrieve the calling parameters.
while getopts "g:i:p:f:t:h" option; do
    case "${option}" in
        g) security_group_id="${OPTARG}" ;;
        i) ip_address="${OPTARG}" ;;
        p) protocol="${OPTARG}" ;;
        f) from_port="${OPTARG}" ;;
        t) to_port="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -g parameter."
    usage
    return 1
fi

if [[ -z "$ip_address" ]]; then
    errecho "ERROR: You must provide an IP address or CIDR block with the -i
parameter."
    usage
    return 1
fi

if [[ -z "$protocol" ]]; then
    errecho "ERROR: You must provide a protocol with the -p parameter."
```

```

usage
return 1
fi

if [[ -z "$from_port" ]]; then
    errecho "ERROR: You must provide a start port with the -f parameter."
    usage
    return 1
fi

if [[ -z "$to_port" ]]; then
    errecho "ERROR: You must provide an end port with the -t parameter."
    usage
    return 1
fi

response=$(aws ec2 authorize-security-group-ingress \
    --group-id "$security_group_id" \
    --cidr "${ip_address}/32" \
    --protocol "$protocol" \
    --port "$from_port-$to_port" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports authorize-security-group-ingress operation failed.
$response"
    return 1
}

return 0
}

```

The utility functions used in this example.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

```

```
#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}

```

- For API details, see [AuthorizeSecurityGroupIngress](#) in *AWS CLI Command Reference*.

CreateKeyPair

The following code example shows how to use CreateKeyPair.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function ec2_create_keypair
#
# This function creates an Amazon Elastic Compute Cloud (Amazon EC2) ED25519 or
# 2048-bit RSA key pair
# and writes it to a file.
#
# Parameters:
#     -n key_pair_name - A key pair name.
#     -f file_path - File to store the key pair.
#
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function ec2_create_keypair() {
    local key_pair_name file_path response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_create_keypair"
        echo "Creates an Amazon Elastic Compute Cloud (Amazon EC2) ED25519 or 2048-bit
RSA key pair"
        echo " and writes it to a file."
        echo "  -n key_pair_name - A key pair name."
        echo "  -f file_path - File to store the key pair."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "n:f:h" option; do
        case "${option}" in
            n) key_pair_name="${OPTARG}" ;;

```

```
f) file_path="${OPTARG}" ;;
h)
    usage
    return 0
    ;;
\?)
    echo "Invalid parameter"
    usage
    return 1
    ;;
esac
done
export OPTIND=1

if [[ -z "$key_pair_name" ]]; then
    errecho "ERROR: You must provide a key name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$file_path" ]]; then
    errecho "ERROR: You must provide a file path with the -f parameter."
    usage
    return 1
fi

response=$(aws ec2 create-key-pair \
    --key-name "$key_pair_name" \
    --query 'KeyMaterial' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
}

if [[ -n "$file_path" ]]; then
    echo "$response" >"$file_path"
fi

return 0
}
```

The utility functions used in this example.

```
#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}
```

- For API details, see [CreateKeyPair](#) in *AWS CLI Command Reference*.

CreateSecurityGroup

The following code example shows how to use CreateSecurityGroup.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function ec2_create_security_group
#
# This function creates an Amazon Elastic Compute Cloud (Amazon EC2) security group.
#
# Parameters:
#     -n security_group_name - The name of the security group.
#     -d security_group_description - The description of the security group.
#
# Returns:
#     The ID of the created security group, or an error message if the operation
#     fails.
# And:
#     0 - If successful.
#     1 - If it fails.
#
#####
function ec2_create_security_group() {
    local security_group_name security_group_description response

    # Function to display usage information
    function usage() {
        echo "function ec2_create_security_group"
        echo "Creates an Amazon Elastic Compute Cloud (Amazon EC2) security group."
        echo "  -n security_group_name - The name of the security group."
        echo "  -d security_group_description - The description of the security group."
    }
}
```



```
    echo ""
}

# Parse the command-line arguments
while getopts "n:d:h" option; do
    case "${option}" in
        n) security_group_name="${OPTARG}" ;;
        d) security_group_description="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

# Validate the input parameters
if [[ -z "$security_group_name" ]]; then
    errecho "ERROR: You must provide a security group name with the -n parameter."
    return 1
fi

if [[ -z "$security_group_description" ]]; then
    errecho "ERROR: You must provide a security group description with the -d
parameter."
    return 1
fi

# Create the security group
response=$(aws ec2 create-security-group \
    --group-name "$security_group_name" \
    --description "$security_group_description" \
    --query "GroupId" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports create-security-group operation failed."
    errecho "$response"
    return 1
}
```

```

echo "$response"
return 0
}

```

The utility functions used in this example.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then

```

```

    errecho " The service returned an error."
elif [ "$err_code" == 255 ]; then
    errecho " 255 is a catch-all error."
fi

return 0
}

```

- For API details, see [CreateSecurityGroup](#) in *AWS CLI Command Reference*.

DeleteKeyPair

The following code example shows how to use DeleteKeyPair.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```

#####
# function ec2_delete_keypair
#
# This function deletes an Amazon EC2 ED25519 or 2048-bit RSA key pair.
#
# Parameters:
#     -n key_pair_name - A key pair name.
#
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function ec2_delete_keypair() {
    local key_pair_name response

    local option OPTARG # Required to use getopt command in a function.
    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_delete_keypair"
    }
}

```

```

    echo "Deletes an Amazon EC2 ED25519 or 2048-bit RSA key pair."
    echo "  -n key_pair_name - A key pair name."
    echo ""
}

# Retrieve the calling parameters.
while getopts "n:h" option; do
    case "${option}" in
        n) key_pair_name="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

if [[ -z "$key_pair_name" ]]; then
    errecho "ERROR: You must provide a key pair name with the -n parameter."
    usage
    return 1
fi

response=$(aws ec2 delete-key-pair \
    --key-name "$key_pair_name") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports delete-key-pair operation failed.$response"
    return 1
}

return 0
}

```

The utility functions used in this example.

```

#####
# function errecho

```

```

#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}

```

- For API details, see [DeleteKeyPair](#) in *AWS CLI Command Reference*.

DeleteSecurityGroup

The following code example shows how to use DeleteSecurityGroup.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function ec2_delete_security_group
#
# This function deletes an Amazon Elastic Compute Cloud (Amazon EC2) security group.
#
# Parameters:
#     -i security_group_id - The ID of the security group to delete.
#
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function ec2_delete_security_group() {
    local security_group_id response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_delete_security_group"
        echo "Deletes an Amazon Elastic Compute Cloud (Amazon EC2) security group."
        echo "  -i security_group_id - The ID of the security group to delete."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "i:h" option; do
        case "${option}" in
            i) security_group_id="${OPTARG}" ;;
            h)
                usage
                return 0
        esac
    done
}
```

```

        ;;
    \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
done
export OPTIND=1

if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -i parameter."
    usage
    return 1
fi

response=$(aws ec2 delete-security-group --group-id "$security_group_id" --output
text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports delete-security-group operation failed.$response"
    return 1
}

return 0
}

```

The utility functions used in this example.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#

```

```

# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}

```

- For API details, see [DeleteSecurityGroup](#) in *AWS CLI Command Reference*.

DescribeImages

The following code example shows how to use DescribeImages.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).


```
#####
# function ec2_describe_images
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
images.
#
# Parameters:
#     -i image_ids - A space-separated list of image IDs (optional).
#     -h - Display help.
#
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function ec2_describe_images() {
    local image_ids response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_describe_images"
        echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2) images."
        echo "  -i image_ids - A space-separated list of image IDs (optional)."
        echo "  -h - Display help."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "i:h" option; do
        case "${option}" in
            i) image_ids="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1
}
```

```

local aws_cli_args=()

if [[ -n "$image_ids" ]]; then
    # shellcheck disable=SC2206
    aws_cli_args+=("--image-ids" $image_ids)
fi

response=$(aws ec2 describe-images \
    "${aws_cli_args[@]}" \
    --query 'Images[*].[Description,Architecture,ImageId]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-images operation failed.$response"
    return 1
}

echo "$response"

return 0
}

```

The utility functions used in this example.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:

```

```
#          0: - Success.
#
#####
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho " One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho " Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho " Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho " Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho " The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho " The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho " 255 is a catch-all error."
  fi

  return 0
}
```

- For API details, see [DescribeImages](#) in *AWS CLI Command Reference*.

DescribeInstanceTypes

The following code example shows how to use DescribeInstanceTypes.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# ec2_describe_instance_types
```

```

#
# This function describes EC2 instance types filtered by processor architecture
# and optionally by instance type. It takes the following arguments:
#
# -a, --architecture ARCHITECTURE   Specify the processor architecture (e.g., x86_64)
# -t, --type INSTANCE_TYPE           Comma-separated list of instance types (e.g.,
#   t2.micro)
# -h, --help                         Show the usage help
#
# The function prints the instance type and supported architecture for each
# matching instance type.
#####
function ec2_describe_instance_types() {
    local architecture=""
    local instance_types=""

    # bashsupport disable=BP5008
    function usage() {
        echo "Usage: ec2_describe_instance_types [-a|--architecture ARCHITECTURE] [-t|--
type INSTANCE_TYPE] [-h|--help]"
        echo "  -a, --architecture ARCHITECTURE   Specify the processor architecture
(e.g., x86_64)"
        echo "  -t, --type INSTANCE_TYPE           Comma-separated list of instance types
(e.g., t2.micro)"
        echo "  -h, --help                         Show this help message"
    }

    while [[ $# -gt 0 ]]; do
        case "$1" in
            -a | --architecture)
                architecture="$2"
                shift 2
                ;;
            -t | --type)
                instance_types="$2"
                shift 2
                ;;
            -h | --help)
                usage
                return 0
                ;;
            *)
                echo "Unknown argument: $1"
                return 1
        esac
    done
}

```

```
        ;;
    esac
done

if [[ -z "$architecture" ]]; then
    errecho "Error: Architecture not specified."
    usage
    return 1
fi

if [[ -z "$instance_types" ]]; then
    errecho "Error: Instance type not specified."
    usage
    return 1
fi

local tmp_json_file="temp_ec2.json"
echo -n '['
{
    "Name": "processor-info.supported-architecture",
    "Values": [' >"$tmp_json_file"

local items
IFS=',' read -ra items <<<"$architecture"
local array_size
array_size=${#items[@]}
for i in $(seq 0 $((array_size - 1))); do
    echo -n '""${items[$i]}""' >>"$tmp_json_file"
    if [[ $i -lt $((array_size - 1)) ]]; then
        echo -n ',' >>"$tmp_json_file"
    fi
done
echo -n ']],'
{
    "Name": "instance-type",
    "Values": [' >>"$tmp_json_file"
IFS=',' read -ra items <<<"$instance_types"
local array_size
array_size=${#items[@]}
for i in $(seq 0 $((array_size - 1))); do
    echo -n '""${items[$i]}""' >>"$tmp_json_file"
    if [[ $i -lt $((array_size - 1)) ]]; then
        echo -n ',' >>"$tmp_json_file"
    fi
fi
```

```

done

echo -n ']]]' >>"$tmp_json_file"

local response
response=$(aws ec2 describe-instance-types --filters file://"${tmp_json_file}" \
  --query 'InstanceTypes[*].[InstanceType]' --output text)

local error_code=$?

rm "$tmp_json_file"

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  echo "ERROR: AWS reports describe-instance-types operation failed."
  return 1
fi

echo "$response"
return 0
}

```

The utility functions used in this example.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:

```

```

#          0: - Success.
#
#####
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho " One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho " Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho " Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho " Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho " The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho " The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho " 255 is a catch-all error."
  fi

  return 0
}

```

- For API details, see [DescribeInstanceTypes](#) in *AWS CLI Command Reference*.

DescribeInstances

The following code example shows how to use DescribeInstances.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```

#####
# function ec2_describe_instances

```

```

#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
# instances.
#
# Parameters:
#     -i instance_id - The ID of the instance to describe (optional).
#     -q query - The query to filter the response (optional).
#     -h - Display help.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function ec2_describe_instances() {
    local instance_id query response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_describe_instances"
        echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
instances."
        echo "  -i instance_id - The ID of the instance to describe (optional).\"
        echo "  -q query - The query to filter the response (optional).\"
        echo "  -h - Display help.\"
        echo \"\"
    }

    # Retrieve the calling parameters.
    while getopt "i:q:h" option; do
        case "${option}" in
            i) instance_id="${OPTARG}" ;;
            q) query="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
}

```



```

export OPTIND=1

local aws_cli_args=()

if [[ -n "$instance_id" ]]; then
    # shellcheck disable=SC2206
    aws_cli_args+=("--instance-ids" $instance_id)
fi

local query_arg=""
if [[ -n "$query" ]]; then
    query_arg="--query '$query'"
else
    query_arg="--query Reservations[*].Instances[*].
[InstanceId,ImageId,InstanceType,KeyName,VpcId,PublicIpAddress,State.Name]"
fi

# shellcheck disable=SC2086
response=$(aws ec2 describe-instances \
    "${aws_cli_args[@]}" \
    $query_arg \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-instances operation failed.$response"
    return 1
}

echo "$response"

return 0
}

```

The utility functions used in this example.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

```

```
#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}

```

- For API details, see [DescribeInstances](#) in *AWS CLI Command Reference*.

DescribeKeyPairs

The following code example shows how to use DescribeKeyPairs.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function ec2_describe_key_pairs
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2) key
# pairs.
#
# Parameters:
#     -h - Display help.
#
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function ec2_describe_key_pairs() {
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_describe_key_pairs"
        echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2) key
pairs."
        echo "  -h - Display help."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "h" option; do
        case "${option}" in
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
        esac
    done
}
```

```

        usage
        return 1
    ;;
    esac
done
export OPTIND=1

local response

response=$(aws ec2 describe-key-pairs \
  --query 'KeyPairs[*].[KeyName, KeyFingerprint]' \
  --output text) || {
  aws_cli_error_log ${?}
  errecho "ERROR: AWS reports describe-key-pairs operation failed.$response"
  return 1
}

echo "$response"

return 0
}

```

The utility functions used in this example.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:

```

```

#           0: - Success.
#
#####
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho " One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho " Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho " Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho " Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho " The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho " The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho " 255 is a catch-all error."
  fi

  return 0
}

```

- For API details, see [DescribeKeyPairs](#) in *AWS CLI Command Reference*.

DescribeSecurityGroups

The following code example shows how to use DescribeSecurityGroups.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```

#####
# function ec2_describe_security_groups

```

```

#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
# security groups.
#
# Parameters:
#     -g security_group_id - The ID of the security group to describe (optional).
#
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function ec2_describe_security_groups() {
    local security_group_id response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_describe_security_groups"
        echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2) security
groups."
        echo "  -g security_group_id - The ID of the security group to describe
(optional)."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "g:h" option; do
        case "${option}" in
            g) security_group_id="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    local query="SecurityGroups[*].[GroupName, GroupId, VpcId, IpPermissions[*].
[IpProtocol, FromPort, ToPort, IpRanges[*].CidrIp]]"

```

```

if [[ -n "$security_group_id" ]]; then
    response=$(aws ec2 describe-security-groups --group-ids "$security_group_id" --
query "${query}" --output text)
else
    response=$(aws ec2 describe-security-groups --query "${query}" --output text)
fi

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports describe-security-groups operation failed.$response"
    return 1
fi

echo "$response"

return 0
}

```

The utility functions used in this example.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.

```

```
#
#####
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho " One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho " Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho " Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho " Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho " The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho " The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho " 255 is a catch-all error."
  fi

  return 0
}
```

- For API details, see [DescribeSecurityGroups](#) in *AWS CLI Command Reference*.

DisassociateAddress

The following code example shows how to use DisassociateAddress.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function ec2_disassociate_address
#
```



```
# This function disassociates an Elastic IP address from an Amazon Elastic Compute
Cloud (Amazon EC2) instance.
#
# Parameters:
#     -a association_id - The association ID that represents the association of
the Elastic IP address with an instance.
#
# And:
#     0 - If successful.
#     1 - If it fails.
#
#####
function ec2_disassociate_address() {
    local association_id response

    # Function to display usage information
    function usage() {
        echo "function ec2_disassociate_address"
        echo "Disassociates an Elastic IP address from an Amazon Elastic Compute Cloud
(Amazon EC2) instance."
        echo " -a association_id - The association ID that represents the association
of the Elastic IP address with an instance."
        echo ""
    }

    # Parse the command-line arguments
    while getopts "a:h" option; do
        case "${option}" in
            a) association_id="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    # Validate the input parameters
    if [[ -z "$association_id" ]]; then
```

```

    errecho "ERROR: You must provide an association ID with the -a parameter."
    return 1
fi

response=$(aws ec2 disassociate-address \
  --association-id "$association_id") || {
  aws_cli_error_log ${?}
  errecho "ERROR: AWS reports disassociate-address operation failed."
  errecho "$response"
  return 1
}

return 0
}

```

The utility functions used in this example.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then

```

```

    errecho " One or more S3 transfers failed."
elif [ "$err_code" == 2 ]; then
    errecho " Command line failed to parse."
elif [ "$err_code" == 130 ]; then
    errecho " Process received SIGINT."
elif [ "$err_code" == 252 ]; then
    errecho " Command syntax invalid."
elif [ "$err_code" == 253 ]; then
    errecho " The system environment or configuration was invalid."
elif [ "$err_code" == 254 ]; then
    errecho " The service returned an error."
elif [ "$err_code" == 255 ]; then
    errecho " 255 is a catch-all error."
fi

return 0
}

```

- For API details, see [DisassociateAddress](#) in *AWS CLI Command Reference*.

ReleaseAddress

The following code example shows how to use ReleaseAddress.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```

#####
# function ec2_release_address
#
# This function releases an Elastic IP address from an Amazon Elastic Compute Cloud
# (Amazon EC2) instance.
#
# Parameters:
#     -a allocation_id - The allocation ID of the Elastic IP address to release.
#

```

```

# Returns:
#     0 - If successful.
#     1 - If it fails.
#
#####
function ec2_release_address() {
    local allocation_id response

    # Function to display usage information
    function usage() {
        echo "function ec2_release_address"
        echo "Releases an Elastic IP address from an Amazon Elastic Compute Cloud
(Amazon EC2) instance."
        echo "  -a allocation_id - The allocation ID of the Elastic IP address to
release."
        echo ""
    }

    # Parse the command-line arguments
    while getopts "a:h" option; do
        case "${option}" in
            a) allocation_id="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    # Validate the input parameters
    if [[ -z "$allocation_id" ]]; then
        errecho "ERROR: You must provide an allocation ID with the -a parameter."
        return 1
    fi

    response=$(aws ec2 release-address \
        --allocation-id "$allocation_id") || {
        aws_cli_error_log ${?}
    }
}

```

```

    errecho "ERROR: AWS reports release-address operation failed."
    errecho "$response"
    return 1
}

return 0
}

```

The utility functions used in this example.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    }
}

```

```

elif [ "$err_code" == 253 ]; then
    errecho " The system environment or configuration was invalid."
elif [ "$err_code" == 254 ]; then
    errecho " The service returned an error."
elif [ "$err_code" == 255 ]; then
    errecho " 255 is a catch-all error."
fi

return 0
}

```

- For API details, see [ReleaseAddress](#) in *AWS CLI Command Reference*.

RunInstances

The following code example shows how to use RunInstances.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```

#####
# function ec2_run_instances
#
# This function launches one or more Amazon Elastic Compute Cloud (Amazon EC2)
# instances.
#
# Parameters:
#   -i image_id - The ID of the Amazon Machine Image (AMI) to use.
#   -t instance_type - The instance type to use (e.g., t2.micro).
#   -k key_pair_name - The name of the key pair to use.
#   -s security_group_id - The ID of the security group to use.
#   -c count - The number of instances to launch (default: 1).
#   -h - Display help.
#
# Returns:
#   0 - If successful.

```

```

# 1 - If it fails.
#####
function ec2_run_instances() {
    local image_id instance_type key_pair_name security_group_id count response
    local option OPTARG # Required to use getopt command in a function.

# bashsupport disable=BP5008
function usage() {
    echo "function ec2_run_instances"
    echo "Launches one or more Amazon Elastic Compute Cloud (Amazon EC2) instances."
    echo " -i image_id - The ID of the Amazon Machine Image (AMI) to use."
    echo " -t instance_type - The instance type to use (e.g., t2.micro)."
    echo " -k key_pair_name - The name of the key pair to use."
    echo " -s security_group_id - The ID of the security group to use."
    echo " -c count - The number of instances to launch (default: 1)."
    echo " -h - Display help."
    echo ""
}

# Retrieve the calling parameters.
while getopt "i:t:k:s:c:h" option; do
    case "${option}" in
        i) image_id="${OPTARG}" ;;
        t) instance_type="${OPTARG}" ;;
        k) key_pair_name="${OPTARG}" ;;
        s) security_group_id="${OPTARG}" ;;
        c) count="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

if [[ -z "$image_id" ]]; then
    errecho "ERROR: You must provide an Amazon Machine Image (AMI) ID with the -i
parameter."
    usage

```

```
    return 1
fi

if [[ -z "$instance_type" ]]; then
    errecho "ERROR: You must provide an instance type with the -t parameter."
    usage
    return 1
fi

if [[ -z "$key_pair_name" ]]; then
    errecho "ERROR: You must provide a key pair name with the -k parameter."
    usage
    return 1
fi

if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -s parameter."
    usage
    return 1
fi

if [[ -z "$count" ]]; then
    count=1
fi

response=$(aws ec2 run-instances \
    --image-id "$image_id" \
    --instance-type "$instance_type" \
    --key-name "$key_pair_name" \
    --security-group-ids "$security_group_id" \
    --count "$count" \
    --query 'Instances[*].[InstanceId]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports run-instances operation failed.$response"
    return 1
}

echo "$response"

return 0
}
```


The utility functions used in this example.

```
#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}

```

- For API details, see [RunInstances](#) in *AWS CLI Command Reference*.

StartInstances

The following code example shows how to use StartInstances.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function ec2_start_instances
#
# This function starts one or more Amazon Elastic Compute Cloud (Amazon EC2)
# instances.
#
# Parameters:
#   -i instance_id - The ID(s) of the instance(s) to start (comma-separated).
#   -h - Display help.
#
# Returns:
#   0 - If successful.
#   1 - If it fails.
#####
function ec2_start_instances() {
    local instance_ids
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_start_instances"
        echo "Starts one or more Amazon Elastic Compute Cloud (Amazon EC2) instances."
        echo "  -i instance_id - The ID(s) of the instance(s) to start (comma-separated)."
        echo "  -h - Display help."
        echo ""
    }
}
```

```

}

# Retrieve the calling parameters.
while getopts "i:h" option; do
  case "${option}" in
    i) instance_ids="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$instance_ids" ]]; then
  errecho "ERROR: You must provide one or more instance IDs with the -i
parameter."
  usage
  return 1
fi

response=$(aws ec2 start-instances \
  --instance-ids "${instance_ids}") || {
  aws_cli_error_log ${?}
  errecho "ERROR: AWS reports start-instances operation failed with $response."
  return 1
}

return 0
}

```

The utility functions used in this example.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).

```

```
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}


```

- For API details, see [StartInstances](#) in *AWS CLI Command Reference*.

StopInstances

The following code example shows how to use StopInstances.

AWS CLI with Bash script

 **Note**

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function ec2_stop_instances
#
# This function stops one or more Amazon Elastic Compute Cloud (Amazon EC2)
# instances.
#
# Parameters:
#   -i instance_id - The ID(s) of the instance(s) to stop (comma-separated).
#   -h - Display help.
#
# Returns:
#   0 - If successful.
#   1 - If it fails.
#####
function ec2_stop_instances() {
    local instance_ids
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_stop_instances"
        echo "Stops one or more Amazon Elastic Compute Cloud (Amazon EC2) instances."
        echo "  -i instance_id - The ID(s) of the instance(s) to stop (comma-
separated)."
        echo "  -h - Display help."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "i:h" option; do
        case "${option}" in
```

```

        i) instance_ids="${OPTARG}" ;;
    h)
        usage
        return 0
        ;;
    \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
esac
done
export OPTIND=1

if [[ -z "$instance_ids" ]]; then
    errecho "ERROR: You must provide one or more instance IDs with the -i
parameter."
    usage
    return 1
fi

response=$(aws ec2 stop-instances \
--instance-ids "${instance_ids}") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports stop-instances operation failed with $response."
    return 1
}

return 0
}

```

The utility functions used in this example.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

```

```
#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}

```

- For API details, see [StopInstances](#) in *AWS CLI Command Reference*.

TerminateInstances

The following code example shows how to use `TerminateInstances`.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function ec2_terminate_instances
#
# This function terminates one or more Amazon Elastic Compute Cloud (Amazon EC2)
# instances using the AWS CLI.
#
# Parameters:
#     -i instance_ids - A space-separated list of instance IDs.
#     -h - Display help.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function ec2_terminate_instances() {
    local instance_ids response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_terminate_instances"
        echo "Terminates one or more Amazon Elastic Compute Cloud (Amazon EC2)
instances."
        echo "  -i instance_ids - A space-separated list of instance IDs."
        echo "  -h - Display help."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "i:h" option; do
        case "${option}" in
            i) instance_ids="${OPTARG}" ;;
            h)
                usage

```



```

        return 0
        ;;
    \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
done
export OPTIND=1

# Check if instance ID is provided
if [[ -z "${instance_ids}" ]]; then
    echo "Error: Missing required instance IDs parameter."
    usage
    return 1
fi

# shellcheck disable=SC2086
response=$(aws ec2 terminate-instances \
    "--instance-ids" $instance_ids \
    "--query 'TerminatingInstances[*].[InstanceId,CurrentState.Name]' \
    "--output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports terminate-instances operation failed.$response"
    return 1
}

return 0
}

```

The utility functions used in this example.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

```

```
#####
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}

```

- For API details, see [TerminateInstances](#) in *AWS CLI Command Reference*.

Scenarios

Get started with instances

The following code example shows how to:

- Create a key pair and security group.
- Select an Amazon Machine Image (AMI) and compatible instance type, then create an instance.
- Stop and restart the instance.
- Associate an Elastic IP address with your instance.
- Connect to your instance with SSH, then clean up resources.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

Run an interactive scenario at a command prompt.

```
#####
# function get_started_with_ec2_instances
#
# Runs an interactive scenario that shows how to get started using EC2 instances.
#
# "EC2 access" permissions are needed to run this code.
#
# Returns:
# 0 - If successful.
# 1 - If an error occurred.
#####
function get_started_with_ec2_instances() {
    # Requires version 4 for mapfile.
    local required_version=4.0

    # Get the current Bash version
    # Check if BASH_VERSION is set
    local current_version
    if [[ -n "$BASH_VERSION" ]]; then
        # Convert BASH_VERSION to a number for comparison
        current_version=$BASH_VERSION
    else
        # Get the current Bash version using the bash command
        current_version=$(bash --version | head -n 1 | awk '{ print $4 }')
```

```
fi

# Convert version strings to numbers for comparison
local required_version_num current_version_num
required_version_num=$(echo "$required_version" | awk -F. '{ print ($1 * 10000) +
($2 * 100) + $3 }')
current_version_num=$(echo "$current_version" | awk -F. '{ print ($1 * 10000) +
($2 * 100) + $3 }')

# Compare versions
if ((current_version_num < required_version_num)); then
    echo "Error: This script requires Bash version $required_version or higher."
    echo "Your current Bash version is number is $current_version."
    exit 1
fi

{
    if [ "$EC2_OPERATIONS_SOURCED" != "True" ]; then

        source ./ec2_operations.sh
    fi
}

echo_repeat "*" 88
echo "Welcome to the Amazon Elastic Compute Cloud (Amazon EC2) get started with
instances demo."
echo_repeat "*" 88
echo

echo "Let's create an RSA key pair that you can be use to securely connect to "
echo "your EC2 instance."

echo -n "Enter a unique name for your key: "
get_input
local key_name
key_name=$get_input_result

local temp_dir
temp_dir=$(mktemp -d)
local key_file_name="$temp_dir/${key_name}.pem"

if ec2_create_keypair -n "${key_name}" -f "${key_file_name}"; then
    echo "Created a key pair $key_name and saved the private key to $key_file_name"
    echo
```

```
else
  errecho "The key pair failed to create. This demo will exit."
  return 1
fi

chmod 400 "${key_file_name}"

if yes_no_input "Do you want to list some of your key pairs? (y/n) "; then
  local keys_and_fingerprints
  keys_and_fingerprints="$(ec2_describe_key_pairs)" && {
    local image_name_and_id
    while IFS=$'\n' read -r image_name_and_id; do
      local entries
      IFS=$'\t' read -ra entries <<<"$image_name_and_id"
      echo "Found rsa key ${entries[0]} with fingerprint:"
      echo "    ${entries[1]}"
    done <<<"$keys_and_fingerprints"
  }
fi

echo_repeat "*" 88
echo_repeat "*" 88

echo "Let's create a security group to manage access to your instance."
echo -n "Enter a unique name for your security group: "
get_input
local security_group_name
security_group_name=$get_input_result
local security_group_id
security_group_id=$(ec2_create_security_group -n "$security_group_name" \
  -d "Security group for EC2 instance") || {
  errecho "The security failed to create. This demo will exit."
  clean_up "$key_name" "$key_file_name"
  return 1
}

echo "Security group created with ID $security_group_id"
echo

local public_ip
public_ip=$(curl -s http://checkip.amazonaws.com)

echo "Let's add a rule to allow SSH only from your current IP address."
```

```

echo "Your public IP address is $public_ip"
echo -n "press return to add this rule to your security group."
get_input

if ! ec2_authorize_security_group_ingress -g "$security_group_id" -i "$public_ip"
-p tcp -f 22 -t 22; then
  errecho "The security group rules failed to update. This demo will exit."
  clean_up "$key_name" "$key_file_name" "$security_group_id"
  return 1
fi

echo "Security group rules updated"

local security_group_description
security_group_description="$(ec2_describe_security_groups -g
"${security_group_id}")" || {
  errecho "Failed to describe security groups. This demo will exit."
  clean_up "$key_name" "$key_file_name" "$security_group_id"
  return 1
}

mapfile -t parameters <<<"$security_group_description"
IFS=$'\t' read -ra entries <<<"${parameters[0]}"
echo "Security group: ${entries[0]}"
echo "  ID: ${entries[1]}"
echo "  VPC: ${entries[2]}"
echo "Inbound permissions:"
IFS=$'\t' read -ra entries <<<"${parameters[1]}"
echo "  IpProtocol: ${entries[0]}"
echo "  FromPort: ${entries[1]}"
echo "  ToPort: ${entries[2]}"
echo "  CidrIp: ${parameters[2]}"

local parameters
parameters="$(ssm_get_parameters_by_path -p "/aws/service/ami-amazon-linux-
latest")" || {
  errecho "Failed to get parameters. This demo will exit."
  clean_up "$key_name" "$key_file_name" "$security_group_id"
  return 1
}

local image_ids=""
mapfile -t parameters <<<"$parameters"

```

```

for image_name_and_id in "${parameters[@]}"; do
    IFS=$'\t' read -ra values <<<"$image_name_and_id"
    if [[ "${values[0]}" == *"amzn2"* ]]; then
        image_ids+="${values[1]} "
    fi
done

local images
images="$(ec2_describe_images -i "$image_ids")" || {
    errecho "Failed to describe images. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
}

new_line_and_tab_to_list "$images"
local images=("${list_result[@]}")

# Get the size of the array
local images_count=${#images[@]}

if ((images_count == 0)); then
    errecho "No images found. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
fi

echo_repeat "*" 88
echo_repeat "*" 88

echo "Let's create an instance from an Amazon Linux 2 AMI. Here are some options:"
for ((i = 0; i < images_count; i += 3)); do
    echo "$(((i / 3) + 1)) - ${images[$i]}"
done

integer_input "Please enter the number of the AMI you want to use: " 1
"$((images_count / 3))"
local choice=$get_input_result
choice=$((choice - 1) * 3)

echo "Great choice."
echo

local architecture=${images[$((choice + 1))]}

```

```

local image_id=${images[$((choice + 2))]}
echo "Here are some instance types that support the ${architecture} architecture
of the image:"
response="$(ec2_describe_instance_types -a "${architecture}" -t
"*micro,*small")" || {
    errecho "Failed to describe instance types. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
}

local instance_types
mapfile -t instance_types <<<"$response"

# Get the size of the array
local instance_types_count=${#instance_types[@]}

echo "Here are some options:"
for ((i = 0; i < instance_types_count; i++)); do
    echo "$((i + 1)) - ${instance_types[$i]}"
done

integer_input "Which one do you want to use? " 1 "${#instance_types[@]}"
"
choice=$get_input_result
local instance_type=${instance_types[$((choice - 1))]}
echo "Another great choice."
echo

echo "Creating your instance and waiting for it to start..."
local instance_id
instance_id=$(ec2_run_instances -i "$image_id" -t "$instance_type" -k "$key_name"
-s "$security_group_id") || {
    errecho "Failed to run instance. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
}

ec2_wait_for_instance_running -i "$instance_id"
echo "Your instance is ready:"
echo

local instance_details
instance_details="$(ec2_describe_instances -i "${instance_id}")"

```



```
echo
print_instance_details "${instance_details}"

local public_ip
public_ip=$(echo "${instance_details}" | awk '{print $6}')
echo
echo "You can use SSH to connect to your instance"
echo "If the connection attempt times out, you might have to manually update the
SSH ingress rule"
echo "for your IP address in the AWS Management Console."
connect_to_instance "$key_file_name" "$public_ip"

echo -n "Press Enter when you're ready to continue the demo: "
get_input

echo_repeat "*" 88
echo_repeat "*" 88

echo "Let's stop and start your instance to see what changes."
echo "Stopping your instance and waiting until it's stopped..."
ec2_stop_instances -i "$instance_id"
ec2_wait_for_instance_stopped -i "$instance_id"

echo "Your instance is stopped. Restarting..."

ec2_start_instances -i "$instance_id"
ec2_wait_for_instance_running -i "$instance_id"

echo "Your instance is running again."
local instance_details
instance_details="$(ec2_describe_instances -i "${instance_id}")"

print_instance_details "${instance_details}"

public_ip=$(echo "${instance_details}" | awk '{print $6}')

echo "Every time your instance is restarted, its public IP address changes"
connect_to_instance "$key_file_name" "$public_ip"

echo -n "Press Enter when you're ready to continue the demo: "
get_input

echo_repeat "*" 88
echo_repeat "*" 88
```

```
echo "You can allocate an Elastic IP address and associate it with your instance"
echo "to keep a consistent IP address even when your instance restarts."

local result
result=$(ec2_allocate_address -d vpc) || {
    errecho "Failed to allocate an address. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id" "$instance_id"
    return 1
}

local elastic_ip allocation_id
elastic_ip=$(echo "$result" | awk '{print $1}')
allocation_id=$(echo "$result" | awk '{print $2}')

echo "Allocated static Elastic IP address: $elastic_ip"

local association_id
association_id=$(ec2_associate_address -i "$instance_id" -a "$allocation_id") || {
    errecho "Failed to associate an address. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id" "$instance_id"
"$allocation_id"
    return 1
}

echo "Associated your Elastic IP with your instance."
echo "You can now use SSH to connect to your instance by using the Elastic IP."
connect_to_instance "$key_file_name" "$elastic_ip"

echo -n "Press Enter when you're ready to continue the demo: "
get_input

echo_repeat "*" 88
echo_repeat "*" 88

echo "Let's stop and start your instance to see what changes."
echo "Stopping your instance and waiting until it's stopped..."
ec2_stop_instances -i "$instance_id"
ec2_wait_for_instance_stopped -i "$instance_id"

echo "Your instance is stopped. Restarting..."

ec2_start_instances -i "$instance_id"
ec2_wait_for_instance_running -i "$instance_id"
```

```
echo "Your instance is running again."
local instance_details
instance_details="$(ec2_describe_instances -i "${instance_id}")"

print_instance_details "${instance_details}"

echo "Because you have associated an Elastic IP with your instance, you can"
echo "connect by using a consistent IP address after the instance restarts."
connect_to_instance "$key_file_name" "$elastic_ip"

echo -n "Press Enter when you're ready to continue the demo: "
get_input

echo_repeat "*" 88
echo_repeat "*" 88

if yes_no_input "Do you want to delete the resources created in this demo: (y/n)
"; then
    clean_up "$key_name" "$key_file_name" "$security_group_id" "$instance_id" \
        "$allocation_id" "$association_id"
else
    echo "The following resources were not deleted."
    echo "Key pair: $key_name"
    echo "Key file: $key_file_name"
    echo "Security group: $security_group_id"
    echo "Instance: $instance_id"
    echo "Elastic IP address: $elastic_ip"
fi
}

#####
# function clean_up
#
# This function cleans up the created resources.
# $1 - The name of the ec2 key pair to delete.
# $2 - The name of the key file to delete.
# $3 - The ID of the security group to delete.
# $4 - The ID of the instance to terminate.
# $5 - The ID of the elastic IP address to release.
# $6 - The ID of the elastic IP address to disassociate.
#
# Returns:
# 0 - If successful.
```

```
#      1 - If an error occurred.
#####
function clean_up() {
    local result=0
    local key_pair_name=$1
    local key_file_name=$2
    local security_group_id=$3
    local instance_id=$4
    local allocation_id=$5
    local association_id=$6

    if [ -n "$association_id" ]; then
        # bashsupport disable=BP2002
        if (ec2_disassociate_address -a "$association_id"); then
            echo "Disassociated elastic IP address with ID $association_id"
        else
            errecho "The elastic IP address disassociation failed."
            result=1
        fi
    fi

    if [ -n "$allocation_id" ]; then
        # bashsupport disable=BP2002
        if (ec2_release_address -a "$allocation_id"); then
            echo "Released elastic IP address with ID $allocation_id"
        else
            errecho "The elastic IP address release failed."
            result=1
        fi
    fi

    if [ -n "$instance_id" ]; then
        # bashsupport disable=BP2002
        if (ec2_terminate_instances -i "$instance_id"); then
            echo "Started terminating instance with ID $instance_id"

            ec2_wait_for_instance_terminated -i "$instance_id"
        else
            errecho "The instance terminate failed."
            result=1
        fi
    fi

    if [ -n "$security_group_id" ]; then
```

```

# bashsupport disable=BP2002
if (ec2_delete_security_group -i "$security_group_id"); then
    echo "Deleted security group with ID $security_group_id"
else
    errecho "The security group delete failed."
    result=1
fi
fi

if [ -n "$key_pair_name" ]; then
    # bashsupport disable=BP2002
    if (ec2_delete_keypair -n "$key_pair_name"); then
        echo "Deleted key pair named $key_pair_name"
    else
        errecho "The key pair delete failed."
        result=1
    fi
fi

if [ -n "$key_file_name" ]; then
    rm -f "$key_file_name"
fi

return $result
}

#####
# function ssm_get_parameters_by_path
#
# This function retrieves one or more parameters from the AWS Systems Manager
# Parameter Store
# by specifying a parameter path.
#
# Parameters:
#     -p parameter_path - The path of the parameter(s) to retrieve.
#
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function ssm_get_parameters_by_path() {
    local parameter_path response
    local option OPTARG # Required to use getopt command in a function.

```

```
# bashsupport disable=BP5008
function usage() {
    echo "function ssm_get_parameters_by_path"
    echo "Retrieves one or more parameters from the AWS Systems Manager Parameter
Store by specifying a parameter path."
    echo "  -p parameter_path - The path of the parameter(s) to retrieve."
    echo ""
}

# Retrieve the calling parameters.
while getopts "p:h" option; do
    case "${option}" in
        p) parameter_path="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

if [[ -z "$parameter_path" ]]; then
    errecho "ERROR: You must provide a parameter path with the -p parameter."
    usage
    return 1
fi

response=$(aws ssm get-parameters-by-path \
    --path "$parameter_path" \
    --query "Parameters[*].[Name, Value]" \
    --output text) || {
    aws_cli_error_log $?
    errecho "ERROR: AWS reports get-parameters-by-path operation failed.$response"
    return 1
}

echo "$response"

return 0
```

```

}

#####
# function print_instance_details
#
# This function prints the details of an Amazon Elastic Compute Cloud (Amazon EC2)
# instance.
#
# Parameters:
#     instance_details - The instance details in the format "InstanceId ImageId
#     InstanceType KeyName VpcId PublicIpAddress State.Name".
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function print_instance_details() {
    local instance_details="$1"

    if [[ -z "${instance_details}" ]]; then
        echo "Error: Missing required instance details argument."
        return 1
    fi

    local instance_id image_id instance_type key_name vpc_id public_ip state
    instance_id=$(echo "${instance_details}" | awk '{print $1}')
    image_id=$(echo "${instance_details}" | awk '{print $2}')
    instance_type=$(echo "${instance_details}" | awk '{print $3}')
    key_name=$(echo "${instance_details}" | awk '{print $4}')
    vpc_id=$(echo "${instance_details}" | awk '{print $5}')
    public_ip=$(echo "${instance_details}" | awk '{print $6}')
    state=$(echo "${instance_details}" | awk '{print $7}')

    echo "    ID: ${instance_id}"
    echo "    Image ID: ${image_id}"
    echo "    Instance type: ${instance_type}"
    echo "    Key name: ${key_name}"
    echo "    VPC ID: ${vpc_id}"
    echo "    Public IP: ${public_ip}"
    echo "    State: ${state}"

    return 0
}

```

```
#####
# function connect_to_instance
#
# This function displays the public IP address of an Amazon Elastic Compute Cloud
  (Amazon EC2) instance and prompts the user to connect to the instance via SSH.
#
# Parameters:
#   $1 - The name of the key file used to connect to the instance.
#   $2 - The public IP address of the instance.
#
# Returns:
#   None
#####
function connect_to_instance() {
  local key_file_name="$1"
  local public_ip="$2"

  # Validate the input parameters
  if [[ -z "$key_file_name" ]]; then
    echo "ERROR: You must provide a key file name as the first argument." >&2
    return 1
  fi

  if [[ -z "$public_ip" ]]; then
    echo "ERROR: You must provide a public IP address as the second argument." >&2
    return 1
  fi

  # Display the public IP address and connection command
  echo "To connect, run the following command:"
  echo "  ssh -i ${key_file_name} ec2-user@${public_ip}"

  # Prompt the user to connect to the instance
  if yes_no_input "Do you want to connect now? (y/n) "; then
    echo "After you have connected, you can return to this example by typing 'exit'"
    ssh -i "${key_file_name}" ec2-user@"${public_ip}"
  fi
}

#####
# function get_input
#
# This function gets user input from the command line.
#
```



```

# Outputs:
#   User input to stdout.
#
# Returns:
#   0
#####
function get_input() {

    if [ -z "${mock_input+x}" ]; then
        read -r get_input_result
    else

        if [ "$mock_input_array_index" -lt ${#mock_input_array[@]} ]; then
            get_input_result="${mock_input_array[$mock_input_array_index]}"
            # bashsupport disable=BP2001
            # shellcheck disable=SC2206
            ((mock_input_array_index++))
            echo -n "$get_input_result"
        else
            echo "MOCK_INPUT_ARRAY has no more elements" 1>&2
            return 1
        fi
    fi

    return 0
}

#####
# function yes_no_input
#
# This function requests a yes/no answer from the user, following to a prompt.
#
# Parameters:
#   $1 - The prompt.
#
# Returns:
#   0 - If yes.
#   1 - If no.
#####
function yes_no_input() {
    if [ -z "$1" ]; then
        echo "Internal error yes_no_input"
        return 1
    fi
}

```

```

local index=0
local response="N"
while [[ $index -lt 10 ]]; do
    index=$((index + 1))
    echo -n "$1"
    if ! get_input; then
        return 1
    fi
    response=$(echo "$get_input_result" | tr '[:upper:]' '[:lower:]')
    if [ "$response" = "y" ] || [ "$response" = "n" ]; then
        break
    else
        echo -e "\nPlease enter or 'y' or 'n'."
    fi
done

echo

if [ "$response" = "y" ]; then
    return 0
else
    return 1
fi
}

#####
# function integer_input
#
# This function prompts the user to enter an integer within a specified range
# and validates the input.
#
# Parameters:
#     $1 - The prompt message to display to the user.
#     $2 - The minimum value of the accepted range.
#     $3 - The maximum value of the accepted range.
#
# Returns:
#     The valid integer input from the user.
#     If the input is invalid or out of range, the function will continue
#     prompting the user until a valid input is provided.
#####
function integer_input() {
    local prompt="$1"

```

```
local min_value="$2"
local max_value="$3"
local input=""

while true; do
  # Display the prompt message and wait for user input
  echo -n "$prompt"

  if ! get_input; then
    return 1
  fi

  input="$get_input_result"

  # Check if the input is a valid integer
  if [[ "$input" =~ ^-?[0-9]+$ ]]; then
    # Check if the input is within the specified range
    if ((input >= min_value && input <= max_value)); then
      return 0
    else
      echo "Error: Input, $input, must be between $min_value and $max_value."
    fi
  else
    echo "Error: Invalid input- $input. Please enter an integer."
  fi
done
}
#####
# function new_line_and_tab_to_list
#
# This function takes a string input containing newlines and tabs, and
# converts it into a list (array) of elements.
#
# Parameters:
#   $1 - The input string containing newlines and tabs.
#
# Returns:
#   The resulting list (array) is stored in the global variable
#   'list_result'.
#####
function new_line_and_tab_to_list() {
  local input=$1
  export list_result
```

```

list_result=()
mapfile -t lines <<<"$input"
local line
for line in "${lines[@]}"; do
    IFS=$'\t' read -ra parameters <<<"$line"
    list_result+=("${parameters[@]}")
done
}

#####
# function echo_repeat
#
# This function prints a string 'n' times to stdout.
#
# Parameters:
#     $1 - The string.
#     $2 - Number of times to print the string.
#
# Outputs:
#     String 'n' times to stdout.
#
# Returns:
#     0
#####
function echo_repeat() {
    local end=$2
    for ((i = 0; i < end; i++)); do
        echo -n "$1"
    done
    echo
}

```

The DynamoDB functions used in this scenario.

```

#####
# function ec2_create_keypair
#
# This function creates an Amazon Elastic Compute Cloud (Amazon EC2) ED25519 or
# 2048-bit RSA key pair
# and writes it to a file.
#
# Parameters:

```

```
# -n key_pair_name - A key pair name.
# -f file_path - File to store the key pair.
#
# And:
# 0 - If successful.
# 1 - If it fails.
#####
function ec2_create_keypair() {
    local key_pair_name file_path response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_create_keypair"
        echo "Creates an Amazon Elastic Compute Cloud (Amazon EC2) ED25519 or 2048-bit
RSA key pair"
        echo " and writes it to a file."
        echo " -n key_pair_name - A key pair name."
        echo " -f file_path - File to store the key pair."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "n:f:h" option; do
        case "${option}" in
            n) key_pair_name="${OPTARG}" ;;
            f) file_path="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    if [[ -z "$key_pair_name" ]]; then
        errecho "ERROR: You must provide a key name with the -n parameter."
        usage
        return 1
    fi
}
```

```

fi

if [[ -z "$file_path" ]]; then
    errecho "ERROR: You must provide a file path with the -f parameter."
    usage
    return 1
fi

response=$(aws ec2 create-key-pair \
    --key-name "$key_pair_name" \
    --query 'KeyMaterial' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
}

if [[ -n "$file_path" ]]; then
    echo "$response" >"$file_path"
fi

return 0
}

#####
# function ec2_describe_key_pairs
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2) key
# pairs.
#
# Parameters:
#     -h - Display help.
#
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function ec2_describe_key_pairs() {
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_describe_key_pairs"
    }
}

```

```
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2) key
pairs."
    echo "  -h - Display help."
    echo ""
}

# Retrieve the calling parameters.
while getopts "h" option; do
    case "${option}" in
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

local response

response=$(aws ec2 describe-key-pairs \
  --query 'KeyPairs[*].[KeyName, KeyFingerprint]' \
  --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-key-pairs operation failed.$response"
    return 1
}

echo "$response"

return 0
}

#####
# function ec2_create_security_group
#
# This function creates an Amazon Elastic Compute Cloud (Amazon EC2) security group.
#
# Parameters:
#   -n security_group_name - The name of the security group.
```

```

#       -d security_group_description - The description of the security group.
#
# Returns:
#       The ID of the created security group, or an error message if the operation
#       fails.
# And:
#       0 - If successful.
#       1 - If it fails.
#
#####
function ec2_create_security_group() {
    local security_group_name security_group_description response

    # Function to display usage information
    function usage() {
        echo "function ec2_create_security_group"
        echo "Creates an Amazon Elastic Compute Cloud (Amazon EC2) security group."
        echo "  -n security_group_name - The name of the security group."
        echo "  -d security_group_description - The description of the security group."
        echo ""
    }

    # Parse the command-line arguments
    while getopts "n:d:h" option; do
        case "${option}" in
            n) security_group_name="${OPTARG}" ;;
            d) security_group_description="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    # Validate the input parameters
    if [[ -z "$security_group_name" ]]; then
        errecho "ERROR: You must provide a security group name with the -n parameter."
        return 1
    fi
}

```



```

fi

if [[ -z "$security_group_description" ]]; then
    errecho "ERROR: You must provide a security group description with the -d
parameter."
    return 1
fi

# Create the security group
response=$(aws ec2 create-security-group \
    --group-name "$security_group_name" \
    --description "$security_group_description" \
    --query "GroupId" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports create-security-group operation failed."
    errecho "$response"
    return 1
}

echo "$response"
return 0
}

#####
# function ec2_describe_security_groups
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
# security groups.
#
# Parameters:
#     -g security_group_id - The ID of the security group to describe (optional).
#
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function ec2_describe_security_groups() {
    local security_group_id response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_describe_security_groups"
    }

```

```

    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2) security
groups."
    echo "  -g security_group_id - The ID of the security group to describe
(optional)."

```

```

    return 0
}

#####
# function ec2_authorize_security_group_ingress
#
# This function authorizes an ingress rule for an Amazon Elastic Compute Cloud
# (Amazon EC2) security group.
#
# Parameters:
#     -g security_group_id - The ID of the security group.
#     -i ip_address - The IP address or CIDR block to authorize.
#     -p protocol - The protocol to authorize (e.g., tcp, udp, icmp).
#     -f from_port - The start of the port range to authorize.
#     -t to_port - The end of the port range to authorize.
#
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function ec2_authorize_security_group_ingress() {
    local security_group_id ip_address protocol from_port to_port response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_authorize_security_group_ingress"
        echo "Authorizes an ingress rule for an Amazon Elastic Compute Cloud (Amazon
        EC2) security group."
        echo "  -g security_group_id - The ID of the security group."
        echo "  -i ip_address - The IP address or CIDR block to authorize."
        echo "  -p protocol - The protocol to authorize (e.g., tcp, udp, icmp)."
        echo "  -f from_port - The start of the port range to authorize."
        echo "  -t to_port - The end of the port range to authorize."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "g:i:p:f:t:h" option; do
        case "${option}" in
            g) security_group_id="${OPTARG}" ;;
            i) ip_address="${OPTARG}" ;;
            p) protocol="${OPTARG}" ;;
            f) from_port="${OPTARG}" ;;

```

```
t) to_port="${OPTARG}" ;;
h)
    usage
    return 0
    ;;
\?)
    echo "Invalid parameter"
    usage
    return 1
    ;;
esac
done
export OPTIND=1

if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -g parameter."
    usage
    return 1
fi

if [[ -z "$ip_address" ]]; then
    errecho "ERROR: You must provide an IP address or CIDR block with the -i
parameter."
    usage
    return 1
fi

if [[ -z "$protocol" ]]; then
    errecho "ERROR: You must provide a protocol with the -p parameter."
    usage
    return 1
fi

if [[ -z "$from_port" ]]; then
    errecho "ERROR: You must provide a start port with the -f parameter."
    usage
    return 1
fi

if [[ -z "$to_port" ]]; then
    errecho "ERROR: You must provide an end port with the -t parameter."
    usage
    return 1
fi
```

```

response=$(aws ec2 authorize-security-group-ingress \
  --group-id "$security_group_id" \
  --cidr "${ip_address}/32" \
  --protocol "$protocol" \
  --port "$from_port-$to_port" \
  --output text) || {
  aws_cli_error_log ${?}
  errecho "ERROR: AWS reports authorize-security-group-ingress operation failed.
$response"
  return 1
}

return 0
}

#####
# function ec2_describe_images
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
# images.
#
# Parameters:
#   -i image_ids - A space-separated list of image IDs (optional).
#   -h - Display help.
#
# And:
#   0 - If successful.
#   1 - If it fails.
#####
function ec2_describe_images() {
  local image_ids response
  local option OPTARG # Required to use getopt command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_images"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2) images."
    echo "  -i image_ids - A space-separated list of image IDs (optional)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.

```

```

while getopts "i:h" option; do
  case "${option}" in
    i) image_ids="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

local aws_cli_args=()

if [[ -n "$image_ids" ]]; then
  # shellcheck disable=SC2206
  aws_cli_args+=("--image-ids" $image_ids)
fi

response=$(aws ec2 describe-images \
  "${aws_cli_args[@]}" \
  --query 'Images[*].[Description,Architecture,ImageId]' \
  --output text) || {
  aws_cli_error_log ${?}
  errecho "ERROR: AWS reports describe-images operation failed.$response"
  return 1
}

echo "$response"

return 0
}

#####
# ec2_describe_instance_types
#
# This function describes EC2 instance types filtered by processor architecture
# and optionally by instance type. It takes the following arguments:
#
# -a, --architecture ARCHITECTURE Specify the processor architecture (e.g., x86_64)

```

```

# -t, --type INSTANCE_TYPE      Comma-separated list of instance types (e.g.,
                                t2.micro)
# -h, --help                    Show the usage help
#
# The function prints the instance type and supported architecture for each
# matching instance type.
#####
function ec2_describe_instance_types() {
    local architecture=""
    local instance_types=""

    # bashsupport disable=BP5008
    function usage() {
        echo "Usage: ec2_describe_instance_types [-a|--architecture ARCHITECTURE] [-t|--
type INSTANCE_TYPE] [-h|--help]"
        echo "  -a, --architecture ARCHITECTURE  Specify the processor architecture
(e.g., x86_64)"
        echo "  -t, --type INSTANCE_TYPE            Comma-separated list of instance types
(e.g., t2.micro)"
        echo "  -h, --help                          Show this help message"
    }

    while [[ $# -gt 0 ]]; do
        case "$1" in
            -a | --architecture)
                architecture="$2"
                shift 2
                ;;
            -t | --type)
                instance_types="$2"
                shift 2
                ;;
            -h | --help)
                usage
                return 0
                ;;
            *)
                echo "Unknown argument: $1"
                return 1
                ;;
        esac
    done

    if [[ -z "$architecture" ]]; then

```

```

    errecho "Error: Architecture not specified."
    usage
    return 1
fi

if [[ -z "$instance_types" ]]; then
    errecho "Error: Instance type not specified."
    usage
    return 1
fi

local tmp_json_file="temp_ec2.json"
echo -n '['
{
    "Name": "processor-info.supported-architecture",
    "Values": [' >"$tmp_json_file"

local items
IFS=', ' read -ra items <<<"$architecture"
local array_size
array_size=${#items[@]}
for i in $(seq 0 $((array_size - 1))); do
    echo -n '""${items[$i]}""' >>"$tmp_json_file"
    if [[ $i -lt $((array_size - 1)) ]]; then
        echo -n ', ' >>"$tmp_json_file"
    fi
done
echo -n ']],'
{
    "Name": "instance-type",
    "Values": [' >>"$tmp_json_file"
IFS=', ' read -ra items <<<"$instance_types"
local array_size
array_size=${#items[@]}
for i in $(seq 0 $((array_size - 1))); do
    echo -n '""${items[$i]}""' >>"$tmp_json_file"
    if [[ $i -lt $((array_size - 1)) ]]; then
        echo -n ', ' >>"$tmp_json_file"
    fi
done

echo -n ']]]' >>"$tmp_json_file"

local response

```



```

response=$(aws ec2 describe-instance-types --filters file://"${tmp_json_file}" \
  --query 'InstanceTypes[*].[InstanceType]' --output text)

local error_code=$?

rm "${tmp_json_file}"

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  echo "ERROR: AWS reports describe-instance-types operation failed."
  return 1
fi

echo "$response"
return 0
}

#####
# function ec2_run_instances
#
# This function launches one or more Amazon Elastic Compute Cloud (Amazon EC2)
# instances.
#
# Parameters:
#   -i image_id - The ID of the Amazon Machine Image (AMI) to use.
#   -t instance_type - The instance type to use (e.g., t2.micro).
#   -k key_pair_name - The name of the key pair to use.
#   -s security_group_id - The ID of the security group to use.
#   -c count - The number of instances to launch (default: 1).
#   -h - Display help.
#
# Returns:
#   0 - If successful.
#   1 - If it fails.
#####
function ec2_run_instances() {
  local image_id instance_type key_pair_name security_group_id count response
  local option OPTARG # Required to use getopt command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_run_instances"
    echo "Launches one or more Amazon Elastic Compute Cloud (Amazon EC2) instances."
    echo "  -i image_id - The ID of the Amazon Machine Image (AMI) to use."
  }
}

```

```
    echo " -t instance_type - The instance type to use (e.g., t2.micro)."  
    echo " -k key_pair_name - The name of the key pair to use."  
    echo " -s security_group_id - The ID of the security group to use."  
    echo " -c count - The number of instances to launch (default: 1)."  
    echo " -h - Display help."  
    echo ""  
}  
  
# Retrieve the calling parameters.  
while getopts "i:t:k:s:c:h" option; do  
    case "${option}" in  
        i) image_id="${OPTARG}" ;;  
        t) instance_type="${OPTARG}" ;;  
        k) key_pair_name="${OPTARG}" ;;  
        s) security_group_id="${OPTARG}" ;;  
        c) count="${OPTARG}" ;;  
        h)  
            usage  
            return 0  
            ;;  
        \?)  
            echo "Invalid parameter"  
            usage  
            return 1  
            ;;  
    esac  
done  
export OPTIND=1  
  
if [[ -z "$image_id" ]]; then  
    errecho "ERROR: You must provide an Amazon Machine Image (AMI) ID with the -i  
parameter."  
    usage  
    return 1  
fi  
  
if [[ -z "$instance_type" ]]; then  
    errecho "ERROR: You must provide an instance type with the -t parameter."  
    usage  
    return 1  
fi  
  
if [[ -z "$key_pair_name" ]]; then  
    errecho "ERROR: You must provide a key pair name with the -k parameter."
```

```

    usage
    return 1
fi

if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -s parameter."
    usage
    return 1
fi

if [[ -z "$count" ]]; then
    count=1
fi

response=$(aws ec2 run-instances \
    --image-id "$image_id" \
    --instance-type "$instance_type" \
    --key-name "$key_pair_name" \
    --security-group-ids "$security_group_id" \
    --count "$count" \
    --query 'Instances[*].[InstanceId]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports run-instances operation failed.$response"
    return 1
}

echo "$response"

return 0
}

#####
# function ec2_describe_instances
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
# instances.
#
# Parameters:
#     -i instance_id - The ID of the instance to describe (optional).
#     -q query - The query to filter the response (optional).
#     -h - Display help.
#
# Returns:

```

```

#      0 - If successful.
#      1 - If it fails.
#####
function ec2_describe_instances() {
    local instance_id query response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_describe_instances"
        echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
instances."
        echo "  -i instance_id - The ID of the instance to describe (optional)."
        echo "  -q query - The query to filter the response (optional)."
        echo "  -h - Display help."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "i:q:h" option; do
        case "${option}" in
            i) instance_id="${OPTARG}" ;;
            q) query="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    local aws_cli_args=()

    if [[ -n "$instance_id" ]]; then
        # shellcheck disable=SC2206
        aws_cli_args+=("--instance-ids" $instance_id)
    fi

    local query_arg=""

```

```

if [[ -n "$query" ]]; then
    query_arg="--query '$query'"
else
    query_arg="--query Reservations[*].Instances[*].
[InstanceId,ImageId,InstanceType,KeyName,VpcId,PublicIpAddress,State.Name]"
fi

# shellcheck disable=SC2086
response=$(aws ec2 describe-instances \
    "${aws_cli_args[@]}" \
    $query_arg \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-instances operation failed.$response"
    return 1
}

echo "$response"

return 0
}

#####
# function ec2_stop_instances
#
# This function stops one or more Amazon Elastic Compute Cloud (Amazon EC2)
# instances.
#
# Parameters:
#     -i instance_id - The ID(s) of the instance(s) to stop (comma-separated).
#     -h - Display help.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function ec2_stop_instances() {
    local instance_ids
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_stop_instances"
        echo "Stops one or more Amazon Elastic Compute Cloud (Amazon EC2) instances."
    }
}

```

```
    echo " -i instance_id - The ID(s) of the instance(s) to stop (comma-
separated)."
```

```
    echo " -h - Display help."
```

```
    echo ""
```

```
  }
```

```
  # Retrieve the calling parameters.
```

```
  while getopts "i:h" option; do
```

```
    case "${option}" in
```

```
      i) instance_ids="${OPTARG}" ;;
```

```
      h)
```

```
        usage
```

```
        return 0
```

```
        ;;
```

```
      \?)
```

```
        echo "Invalid parameter"
```

```
        usage
```

```
        return 1
```

```
        ;;
```

```
    esac
```

```
  done
```

```
  export OPTIND=1
```

```
  if [[ -z "$instance_ids" ]]; then
```

```
    errecho "ERROR: You must provide one or more instance IDs with the -i
```

```
parameter."
```

```
    usage
```

```
    return 1
```

```
  fi
```

```
  response=$(aws ec2 stop-instances \
```

```
    --instance-ids "${instance_ids}") || {
```

```
    aws_cli_error_log ${?}
```

```
    errecho "ERROR: AWS reports stop-instances operation failed with $response."
```

```
    return 1
```

```
  }
```

```
  return 0
```

```
}
```

```
#####
```

```
# function ec2_start_instances
```

```
#
```

```

# This function starts one or more Amazon Elastic Compute Cloud (Amazon EC2)
instances.
#
# Parameters:
#     -i instance_id - The ID(s) of the instance(s) to start (comma-separated).
#     -h - Display help.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function ec2_start_instances() {
    local instance_ids
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_start_instances"
        echo "Starts one or more Amazon Elastic Compute Cloud (Amazon EC2) instances."
        echo "  -i instance_id - The ID(s) of the instance(s) to start (comma-
separated)."
        echo "  -h - Display help."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "i:h" option; do
        case "${option}" in
            i) instance_ids="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    if [[ -z "$instance_ids" ]]; then

```

```

    errecho "ERROR: You must provide one or more instance IDs with the -i
parameter."
    usage
    return 1
fi

response=$(aws ec2 start-instances \
  --instance-ids "${instance_ids}") || {
  aws_cli_error_log ${?}
  errecho "ERROR: AWS reports start-instances operation failed with $response."
  return 1
}

return 0
}

#####
# function ec2_allocate_address
#
# This function allocates an Elastic IP address for use with Amazon Elastic Compute
Cloud (Amazon EC2) instances in a specific AWS Region.
#
# Parameters:
#   -d domain - The domain for the Elastic IP address (either 'vpc' or
'standard').
#
# Returns:
#   The allocated Elastic IP address, or an error message if the operation
fails.
# And:
#   0 - If successful.
#   1 - If it fails.
#
#####
function ec2_allocate_address() {
  local domain response

  # Function to display usage information
  function usage() {
    echo "function ec2_allocate_address"
    echo "Allocates an Elastic IP address for use with Amazon Elastic Compute Cloud
(Amazon EC2) instances in a specific AWS Region."
    echo "  -d domain - The domain for the Elastic IP address (either 'vpc' or
'standard')."
  }

```



```
    echo ""
}

# Parse the command-line arguments
while getopts "d:h" option; do
    case "${option}" in
        d) domain="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

# Validate the input parameters
if [[ -z "$domain" ]]; then
    errecho "ERROR: You must provide a domain with the -d parameter (either 'vpc' or
'standard')."
    return 1
fi

if [[ "$domain" != "vpc" && "$domain" != "standard" ]]; then
    errecho "ERROR: Invalid domain value. Must be either 'vpc' or 'standard'."
    return 1
fi

# Allocate the Elastic IP address
response=$(aws ec2 allocate-address \
    --domain "$domain" \
    --query "[PublicIp,AllocationId]" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports allocate-address operation failed."
    errecho "$response"
    return 1
}

echo "$response"
```

```
    return 0
}

#####
# function ec2_associate_address
#
# This function associates an Elastic IP address with an Amazon Elastic Compute
# Cloud (Amazon EC2) instance.
#
# Parameters:
#     -a allocation_id - The allocation ID of the Elastic IP address to associate.
#     -i instance_id - The ID of the EC2 instance to associate the Elastic IP
# address with.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#
#####
function ec2_associate_address() {
    local allocation_id instance_id response

    # Function to display usage information
    function usage() {
        echo "function ec2_associate_address"
        echo "Associates an Elastic IP address with an Amazon Elastic Compute Cloud
(Amazon EC2) instance."
        echo "  -a allocation_id - The allocation ID of the Elastic IP address to
associate."
        echo "  -i instance_id - The ID of the EC2 instance to associate the Elastic IP
address with."
        echo ""
    }

    # Parse the command-line arguments
    while getopts "a:i:h" option; do
        case "${option}" in
            a) allocation_id="${OPTARG}" ;;
            i) instance_id="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)

```

```

        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
done
export OPTIND=1

# Validate the input parameters
if [[ -z "$allocation_id" ]]; then
    errecho "ERROR: You must provide an allocation ID with the -a parameter."
    return 1
fi

if [[ -z "$instance_id" ]]; then
    errecho "ERROR: You must provide an instance ID with the -i parameter."
    return 1
fi

# Associate the Elastic IP address
response=$(aws ec2 associate-address \
    --allocation-id "$allocation_id" \
    --instance-id "$instance_id" \
    --query "AssociationId" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports associate-address operation failed."
    errecho "$response"
    return 1
}

echo "$response"
return 0
}

#####
# function ec2_disassociate_address
#
# This function disassociates an Elastic IP address from an Amazon Elastic Compute
# Cloud (Amazon EC2) instance.
#
# Parameters:
#     -a association_id - The association ID that represents the association of
#     the Elastic IP address with an instance.

```

```

#
# And:
#     0 - If successful.
#     1 - If it fails.
#
#####
function ec2_disassociate_address() {
    local association_id response

    # Function to display usage information
    function usage() {
        echo "function ec2_disassociate_address"
        echo "Disassociates an Elastic IP address from an Amazon Elastic Compute Cloud
(Amazon EC2) instance."
        echo "  -a association_id - The association ID that represents the association
of the Elastic IP address with an instance."
        echo ""
    }

    # Parse the command-line arguments
    while getopts "a:h" option; do
        case "${option}" in
            a) association_id="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    # Validate the input parameters
    if [[ -z "$association_id" ]]; then
        errecho "ERROR: You must provide an association ID with the -a parameter."
        return 1
    fi

    response=$(aws ec2 disassociate-address \
        --association-id "$association_id") || {

```

```

    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports disassociate-address operation failed."
    errecho "$response"
    return 1
}

return 0
}

#####
# function ec2_release_address
#
# This function releases an Elastic IP address from an Amazon Elastic Compute Cloud
# (Amazon EC2) instance.
#
# Parameters:
#     -a allocation_id - The allocation ID of the Elastic IP address to release.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#
#####
function ec2_release_address() {
    local allocation_id response

    # Function to display usage information
    function usage() {
        echo "function ec2_release_address"
        echo "Releases an Elastic IP address from an Amazon Elastic Compute Cloud
        (Amazon EC2) instance."
        echo "  -a allocation_id - The allocation ID of the Elastic IP address to
        release."
        echo ""
    }

    # Parse the command-line arguments
    while getopts "a:h" option; do
        case "${option}" in
            a) allocation_id="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
        esac
    done
}

```

```

    \?)
    echo "Invalid parameter"
    usage
    return 1
    ;;
  esac
done
export OPTIND=1

# Validate the input parameters
if [[ -z "$allocation_id" ]]; then
  errecho "ERROR: You must provide an allocation ID with the -a parameter."
  return 1
fi

response=$(aws ec2 release-address \
  --allocation-id "$allocation_id") || {
  aws_cli_error_log ${?}
  errecho "ERROR: AWS reports release-address operation failed."
  errecho "$response"
  return 1
}

return 0
}

#####
# function ec2_terminate_instances
#
# This function terminates one or more Amazon Elastic Compute Cloud (Amazon EC2)
# instances using the AWS CLI.
#
# Parameters:
#   -i instance_ids - A space-separated list of instance IDs.
#   -h - Display help.
#
# Returns:
#   0 - If successful.
#   1 - If it fails.
#####
function ec2_terminate_instances() {
  local instance_ids response
  local option OPTARG # Required to use getopt command in a function.

```

```
# bashsupport disable=BP5008
function usage() {
    echo "function ec2_terminate_instances"
    echo "Terminates one or more Amazon Elastic Compute Cloud (Amazon EC2)
instances."
    echo "  -i instance_ids - A space-separated list of instance IDs."
    echo "  -h - Display help."
    echo ""
}

# Retrieve the calling parameters.
while getopts "i:h" option; do
    case "${option}" in
        i) instance_ids="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

# Check if instance ID is provided
if [[ -z "${instance_ids}" ]]; then
    echo "Error: Missing required instance IDs parameter."
    usage
    return 1
fi

# shellcheck disable=SC2086
response=$(aws ec2 terminate-instances \
    "--instance-ids" $instance_ids \
    "--query 'TerminatingInstances[*].[InstanceId,CurrentState.Name]' \
    "--output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports terminate-instances operation failed.$response"
    return 1
}
```

```
    return 0
}

#####
# function ec2_delete_security_group
#
# This function deletes an Amazon Elastic Compute Cloud (Amazon EC2) security group.
#
# Parameters:
#     -i security_group_id - The ID of the security group to delete.
#
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function ec2_delete_security_group() {
    local security_group_id response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_delete_security_group"
        echo "Deletes an Amazon Elastic Compute Cloud (Amazon EC2) security group."
        echo "  -i security_group_id - The ID of the security group to delete."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "i:h" option; do
        case "${option}" in
            i) security_group_id="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1
```



```

if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -i parameter."
    usage
    return 1
fi

response=$(aws ec2 delete-security-group --group-id "$security_group_id" --output
text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports delete-security-group operation failed.$response"
    return 1
}

return 0
}

#####
# function ec2_delete_keypair
#
# This function deletes an Amazon EC2 ED25519 or 2048-bit RSA key pair.
#
# Parameters:
#     -n key_pair_name - A key pair name.
#
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function ec2_delete_keypair() {
    local key_pair_name response

    local option OPTARG # Required to use getopt command in a function.
    # bashsupport disable=BP5008
    function usage() {
        echo "function ec2_delete_keypair"
        echo "Deletes an Amazon EC2 ED25519 or 2048-bit RSA key pair."
        echo "  -n key_pair_name - A key pair name."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "n:h" option; do
        case "${option}" in
            n) key_pair_name="${OPTARG}" ;;

```

```

        h)
        usage
        return 0
        ;;
    \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
done
export OPTIND=1

if [[ -z "$key_pair_name" ]]; then
    errecho "ERROR: You must provide a key pair name with the -n parameter."
    usage
    return 1
fi

response=$(aws ec2 delete-key-pair \
    --key-name "$key_pair_name") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports delete-key-pair operation failed.$response"
    return 1
}

return 0
}

```

The utility functions used in this scenario.

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function aws_cli_error_log()

```

```

#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#     $1 - The error code returned by the AWS CLI.
#
# Returns:
#     0: - Success.
#
#####
function aws_cli_error_log() {
    local err_code=$1
    errecho "Error code : $err_code"
    if [ "$err_code" == 1 ]; then
        errecho " One or more S3 transfers failed."
    elif [ "$err_code" == 2 ]; then
        errecho " Command line failed to parse."
    elif [ "$err_code" == 130 ]; then
        errecho " Process received SIGINT."
    elif [ "$err_code" == 252 ]; then
        errecho " Command syntax invalid."
    elif [ "$err_code" == 253 ]; then
        errecho " The system environment or configuration was invalid."
    elif [ "$err_code" == 254 ]; then
        errecho " The service returned an error."
    elif [ "$err_code" == 255 ]; then
        errecho " 255 is a catch-all error."
    fi

    return 0
}

```

- For API details, see the following topics in *AWS CLI Command Reference*.
 - [AllocateAddress](#)
 - [AssociateAddress](#)
 - [AuthorizeSecurityGroupIngress](#)
 - [CreateKeyPair](#)
 - [CreateSecurityGroup](#)
 - [DeleteKeyPair](#)

- [DeleteSecurityGroup](#)
- [DescribeImages](#)
- [DescribeInstanceTypes](#)
- [DescribeInstances](#)
- [DescribeKeyPairs](#)
- [DescribeSecurityGroups](#)
- [DisassociateAddress](#)
- [ReleaseAddress](#)
- [RunInstances](#)
- [StartInstances](#)
- [StopInstances](#)
- [TerminateInstances](#)
- [UnmonitorInstances](#)

HealthImaging examples using AWS CLI with Bash script

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Bash script with HealthImaging.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

CreateDatastore

The following code example shows how to use CreateDatastore.

AWS CLI with Bash script

```
#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function imaging_create_datastore
#
# This function creates an AWS HealthImaging data store for importing DICOM P10
# files.
#
# Parameters:
#     -n data_store_name - The name of the data store.
#
# Returns:
#     The datastore ID.
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function imaging_create_datastore() {
    local datastore_name response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function imaging_create_datastore"
        echo "Creates an AWS HealthImaging data store for importing DICOM P10 files."
        echo "  -n data_store_name - The name of the data store."
        echo ""
    }
}
```

```
# Retrieve the calling parameters.
while getopts "n:h" option; do
  case "${option}" in
    n) datastore_name="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$datastore_name" ]]; then
  errecho "ERROR: You must provide a data store name with the -n parameter."
  usage
  return 1
fi

response=$(aws medical-imaging create-datastore \
  --datastore-name "$datastore_name" \
  --output text \
  --query 'datastoreId')

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports medical-imaging create-datastore operation failed.
$response"
  return 1
fi

echo "$response"

return 0
}
```

- For API details, see [CreateDatastore](#) in *AWS CLI Command Reference*.

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

DeleteDatastore

The following code example shows how to use DeleteDatastore.

AWS CLI with Bash script

```
#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function imaging_delete_datastore
#
# This function deletes an AWS HealthImaging data store.
#
# Parameters:
#     -i datastore_id - The ID of the data store.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function imaging_delete_datastore() {
    local datastore_id response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function imaging_delete_datastore"
        echo "Deletes an AWS HealthImaging data store."
        echo "  -i datastore_id - The ID of the data store."
        echo ""
    }
}
```

```
}

# Retrieve the calling parameters.
while getopts "i:h" option; do
  case "${option}" in
    i) datastore_id="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$datastore_id" ]]; then
  errecho "ERROR: You must provide a data store ID with the -i parameter."
  usage
  return 1
fi

response=$(aws medical-imaging delete-datastore \
  --datastore-id "$datastore_id")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports medical-imaging delete-datastore operation failed.
$response"
  return 1
fi

return 0
}
```

- For API details, see [DeleteDatastore](#) in *AWS CLI Command Reference*.

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

GetDatastore

The following code example shows how to use GetDatastore.

AWS CLI with Bash script

```
#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function imaging_get_datastore
#
# Get a data store's properties.
#
# Parameters:
#     -i data_store_id - The ID of the data store.
#
# Returns:
#     [datastore_name, datastore_id, datastore_status, datastore_arn, created_at,
#     updated_at]
#     And:
#     0 - If successful.
#     1 - If it fails.
#####
function imaging_get_datastore() {
    local datastore_id option OPTARG # Required to use getopt command in a function.
    local error_code
    # bashsupport disable=BP5008
    function usage() {
        echo "function imaging_get_datastore"
        echo "Gets a data store's properties."
    }
}
```

```
    echo " -i datastore_id - The ID of the data store."
    echo ""
}

# Retrieve the calling parameters.
while getopts "i:h" option; do
    case "${option}" in
        i) datastore_id="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

if [[ -z "$datastore_id" ]]; then
    errecho "ERROR: You must provide a data store ID with the -i parameter."
    usage
    return 1
fi

local response

response=$(
    aws medical-imaging get-datastore \
        --datastore-id "$datastore_id" \
        --output text \
        --query "[ datastoreProperties.datastoreName,
datastoreProperties.datastoreId, datastoreProperties.datastoreStatus,
datastoreProperties.datastoreArn, datastoreProperties.createdAt,
datastoreProperties.updatedAt]"
)
error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-datastores operation failed.$response"
    return 1
fi
```

```

fi

echo "$response"

return 0
}

```

- For API details, see [GetDatastore](#) in *AWS CLI Command Reference*.

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

ListDatastores

The following code example shows how to use `ListDatastores`.

AWS CLI with Bash script

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function imaging_list_datastores
#
# List the HealthImaging data stores in the account.
#
# Returns:
#     [[datastore_name, datastore_id, datastore_status]]
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function imaging_list_datastores() {

```

```
local option OPTARG # Required to use getopt command in a function.
local error_code
# bashsupport disable=BP5008
function usage() {
    echo "function imaging_list_datastores"
    echo "Lists the AWS HealthImaging data stores in the account."
    echo ""
}

# Retrieve the calling parameters.
while getopt "h" option; do
    case "${option}" in
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

local response
response=$(aws medical-imaging list-datastores \
    --output text \
    --query "datastoreSummaries[*][datastoreName, datastoreId, datastoreStatus]")
error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-datastores operation failed.$response"
    return 1
fi

echo "$response"

return 0
}
```

- For API details, see [ListDatastores](#) in *AWS CLI Command Reference*.

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

IAM examples using AWS CLI with Bash script

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Bash script with IAM.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)
- [Scenarios](#)

Actions

AttachRolePolicy

The following code example shows how to use `AttachRolePolicy`.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a role.
#
# Parameters:
#     -n role_name -- The name of the IAM role.
#     -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function iam_attach_role_policy() {
    local role_name policy_arn response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_attach_role_policy"
        echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
role."
        echo " -n role_name    The name of the IAM role."
        echo " -p policy_ARN -- The IAM policy document ARN."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "n:p:h" option; do
        case "${option}" in
            n) role_name="${OPTARG}" ;;
            p) policy_arn="${OPTARG}" ;;
            h)
                usage

```

```
        return 0
        ;;
    \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
fi

response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
fi

echo "$response"

return 0
}
```

- For API details, see [AttachRolePolicy](#) in *AWS CLI Command Reference*.

CreateAccessKey

The following code example shows how to use CreateAccessKey.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#     -u user_name -- The name of the IAM user.
#     [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#     [access_key_id access_key_secret]
#     And:
#     0 - If successful.
#     1 - If it fails.
#####
function iam_create_user_access_key() {
    local user_name file_name response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_create_user_access_key"
```



```
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name  The name of the IAM user."
    echo "  [-f file_name]  Optional file name for the access key output."
    echo ""
}

# Retrieve the calling parameters.
while getopts "u:f:h" option; do
    case "${option}" in
        u) user_name="${OPTARG}" ;;
        f) file_name="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
fi

response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
fi

if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
```

```

fi

local key_id key_secret
# shellcheck disable=SC2086
key_id=$(echo $response | cut -f 2 -d ' ')
# shellcheck disable=SC2086
key_secret=$(echo $response | cut -f 4 -d ' ')

echo "$key_id $key_secret"

return 0
}

```

- For API details, see [CreateAccessKey](#) in *AWS CLI Command Reference*.

CreatePolicy

The following code example shows how to use CreatePolicy.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function iam_create_policy
#
# This function creates an IAM policy.
#

```

```

# Parameters:
#     -n policy_name -- The name of the IAM policy.
#     -p policy_json -- The policy document.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function iam_create_policy() {
    local policy_name policy_document response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_create_policy"
        echo "Creates an AWS Identity and Access Management (IAM) policy."
        echo "  -n policy_name  The name of the IAM policy."
        echo "  -p policy_json -- The policy document."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "n:p:h" option; do
        case "${option}" in
            n) policy_name="${OPTARG}" ;;
            p) policy_document="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    if [[ -z "$policy_name" ]]; then
        errecho "ERROR: You must provide a policy name with the -n parameter."
        usage
        return 1
    fi
}

```

```

if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
fi

response=$(aws iam create-policy \
    --policy-name "$policy_name" \
    --policy-document "$policy_document" \
    --output text \
    --query Policy.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
fi

echo "$response"
}

```

- For API details, see [CreatePolicy](#) in *AWS CLI Command Reference*.

CreateRole

The following code example shows how to use CreateRole.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```

#####
# function errecho
#

```

```

# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#     -n role_name -- The name of the IAM role.
#     -p policy_json -- The assume role policy document.
#
# Returns:
#     The ARN of the role.
#     And:
#     0 - If successful.
#     1 - If it fails.
#####
function iam_create_role() {
    local role_name policy_document response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_create_user_access_key"
        echo "Creates an AWS Identity and Access Management (IAM) role."
        echo "  -n role_name  The name of the IAM role."
        echo "  -p policy_json -- The assume role policy document."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "n:p:h" option; do
        case "${option}" in
            n) role_name="${OPTARG}" ;;
            p) policy_document="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)

```

```
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
fi

response=$(aws iam create-role \
    --role-name "$role_name" \
    --assume-role-policy-document "$policy_document" \
    --output text \
    --query Role.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
fi

echo "$response"

return 0
}
```

- For API details, see [CreateRole](#) in *AWS CLI Command Reference*.

CreateUser

The following code example shows how to use CreateUser.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#####
function iecho() {
    if [[ $VERBOSE == true ]]; then
        echo "$@"
    fi
}

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#     -u user_name -- The name of the user to create.
#
# Returns:
```

```

# The ARN of the user.
# And:
# 0 - If successful.
# 1 - If it fails.
#####
function iam_create_user() {
    local user_name response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_create_user"
        echo "Creates an WS Identity and Access Management (IAM) user. You must supply a
username:"
        echo " -u user_name The name of the user. It must be unique within the
account."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "u:h" option; do
        case "${option}" in
            u) user_name="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    if [[ -z "$user_name" ]]; then
        errecho "ERROR: You must provide a username with the -u parameter."
        usage
        return 1
    fi

    iecho "Parameters:\n"
    iecho " User name: $user_name"

```



```

iecho ""

# If the user already exists, we don't want to try to create it.
if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
fi

response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
fi

echo "$response"

return 0
}

```

- For API details, see [CreateUser](#) in *AWS CLI Command Reference*.

DeleteAccessKey

The following code example shows how to use DeleteAccessKey.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```

#####
# function errecho

```

```

#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#     -u user_name -- The name of the user.
#     -k access_key -- The access key to delete.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function iam_delete_access_key() {
    local user_name access_key response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_delete_access_key"
        echo "Deletes an WS Identity and Access Management (IAM) access key for the
specified IAM user"
        echo "  -u user_name    The name of the user."
        echo "  -k access_key    The access key to delete."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "u:k:h" option; do
        case "${option}" in
            u) user_name="${OPTARG}" ;;
            k) access_key="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)

```

```
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
fi

if [[ -z "$access_key" ]]; then
    errecho "ERROR: You must provide an access key with the -k parameter."
    usage
    return 1
fi

iecho "Parameters:\n"
iecho "    Username:  $user_name"
iecho "    Access key: $access_key"
iecho ""

response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
fi

iecho "delete-access-key response:$response"
iecho

return 0
}
```

- For API details, see [DeleteAccessKey](#) in *AWS CLI Command Reference*.

DeletePolicy

The following code example shows how to use DeletePolicy.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#####
function iecho() {
    if [[ $VERBOSE == true ]]; then
        echo "$@"
    fi
}

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#     -n policy_arn -- The name of the IAM policy arn.
```

```

#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function iam_delete_policy() {
    local policy_arn response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_delete_policy"
        echo "Deletes an WS Identity and Access Management (IAM) policy"
        echo "  -n policy_arn -- The name of the IAM policy arn."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "n:h" option; do
        case "${option}" in
            n) policy_arn="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    if [[ -z "$policy_arn" ]]; then
        errecho "ERROR: You must provide a policy arn with the -n parameter."
        usage
        return 1
    fi

    iecho "Parameters:\n"
    iecho "  Policy arn: $policy_arn"
    iecho ""

```

```

response=$(aws iam delete-policy \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
  return 1
fi

iecho "delete-policy response:$response"
iecho

return 0
}

```

- For API details, see [DeletePolicy](#) in *AWS CLI Command Reference*.

DeleteRole

The following code example shows how to use DeleteRole.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```

#####
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#####
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

```

```

}

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#     -n role_name -- The name of the IAM role.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function iam_delete_role() {
    local role_name response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_delete_role"
        echo "Deletes an WS Identity and Access Management (IAM) role"
        echo "  -n role_name -- The name of the IAM role."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "n:h" option; do
        case "${option}" in
            n) role_name="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)

```

```
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
done
export OPTIND=1

echo "role_name:$role_name"
if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
fi

iecho "Parameters:\n"
iecho "    Role name:  $role_name"
iecho ""

response=$(aws iam delete-role \
    --role-name "$role_name")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
fi

iecho "delete-role response:$response"
iecho

return 0
}
```

- For API details, see [DeleteRole](#) in *AWS CLI Command Reference*.

DeleteUser

The following code example shows how to use DeleteUser.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#####
function iecho() {
    if [[ $VERBOSE == true ]]; then
        echo "$@"
    fi
}

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#     -u user_name -- The name of the user to create.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function iam_delete_user() {
```

```
local user_name response
local option OPTARG # Required to use getopt command in a function.

# bashsupport disable=BP5008
function usage() {
    echo "function iam_delete_user"
    echo "Deletes an WS Identity and Access Management (IAM) user. You must supply a
username:"
    echo "  -u user_name    The name of the user."
    echo ""
}

# Retrieve the calling parameters.
while getopt "u:h" option; do
    case "${option}" in
        u) user_name="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
fi

iecho "Parameters:\n"
iecho "  User name:  $user_name"
iecho ""

# If the user does not exist, we don't want to try to delete it.
if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
    return 1
fi
```

```

response=$(aws iam delete-user \
  --user-name "$user_name")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-user operation failed.$response"
  return 1
fi

iecho "delete-user response:$response"
iecho

return 0
}

```

- For API details, see [DeleteUser](#) in *AWS CLI Command Reference*.

DetachRolePolicy

The following code example shows how to use DetachRolePolicy.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
  printf "%s\n" "$*" 1>&2
}

```

```
#####
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a role.
#
# Parameters:
#   -n role_name -- The name of the IAM role.
#   -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#   0 - If successful.
#   1 - If it fails.
#####
function iam_detach_role_policy() {
    local role_name policy_arn response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_detach_role_policy"
        echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
role."
        echo "  -n role_name    The name of the IAM role."
        echo "  -p policy_ARN -- The IAM policy document ARN."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "n:p:h" option; do
        case "${option}" in
            n) role_name="${OPTARG}" ;;
            p) policy_arn="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1
}
```

```
if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
fi

response=$(aws iam detach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
    return 1
fi

echo "$response"

return 0
}
```

- For API details, see [DetachRolePolicy](#) in *AWS CLI Command Reference*.

GetUser

The following code example shows how to use GetUser.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
# (IAM) user already exists.
#
# Parameters:
#     $1 - The name of the IAM user to check.
#
# Returns:
#     0 - If the user already exists.
#     1 - If the user doesn't exist.
#####
function iam_user_exists() {
    local user_name
    user_name=$1

    # Check whether the IAM user already exists.
    # We suppress all output - we're interested only in the return code.

    local errors
    errors=$(aws iam get-user \
        --user-name "$user_name" 2>&1 >/dev/null)

    local error_code=${?}
```

```

if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
else
    if [[ $errors != *"error"*(NoSuchEntity)* ]]; then
        aws_cli_error_log $error_code
        errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
fi
}

```

- For API details, see [GetUser](#) in *AWS CLI Command Reference*.

ListAccessKeys

The following code example shows how to use ListAccessKeys.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function iam_list_access_keys
#
# This function lists the access keys for the specified user.
#

```

```

# Parameters:
#     -u user_name -- The name of the IAM user.
#
# Returns:
#     access_key_ids
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function iam_list_access_keys() {

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_list_access_keys"
        echo "Lists the AWS Identity and Access Management (IAM) access key IDs for the
specified user."
        echo "  -u user_name  The name of the IAM user."
        echo ""
    }

    local user_name response
    local option OPTARG # Required to use getopt command in a function.
    # Retrieve the calling parameters.
    while getopt "u:h" option; do
        case "${option}" in
            u) user_name="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    if [[ -z "$user_name" ]]; then
        errecho "ERROR: You must provide a username with the -u parameter."
        usage
        return 1
    fi
}

```



```

response=$(aws iam list-access-keys \
  --user-name "$user_name" \
  --output text \
  --query 'AccessKeyMetadata[].AccessKeyId')

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports list-access-keys operation failed.$response"
  return 1
fi

echo "$response"

return 0
}

```

- For API details, see [ListAccessKeys](#) in *AWS CLI Command Reference*.

ListUsers

The following code example shows how to use ListUsers.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
  printf "%s\n" "$*" 1>&2
}

```

```
#####
# function iam_list_users
#
# List the IAM users in the account.
#
# Returns:
#     The list of users names
# And:
#     0 - If the user already exists.
#     1 - If the user doesn't exist.
#####
function iam_list_users() {
    local option OPTARG # Required to use getopt command in a function.
    local error_code
    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_list_users"
        echo "Lists the AWS Identity and Access Management (IAM) user in the account."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "h" option; do
        case "${option}" in
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    local response

    response=$(aws iam list-users \
        --output text \
        --query "Users[].UserName")
    error_code=${?}
}
```

```
if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-users operation failed.$response"
    return 1
fi

echo "$response"

return 0
}
```

- For API details, see [ListUsers](#) in *AWS CLI Command Reference*.

Scenarios

Create a user and assume a role

The following code example shows how to create a user and assume a role.

Warning

To avoid security risks, don't use IAM users for authentication when developing purpose-built software or working with real data. Instead, use federation with an identity provider such as [AWS IAM Identity Center](#).

- Create a user with no permissions.
- Create a role that grants permission to list Amazon S3 buckets for the account.
- Add a policy to let the user assume the role.
- Assume the role and list S3 buckets using temporary credentials, then clean up resources.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function iam_create_user_assume_role
#
# Scenario to create an IAM user, create an IAM role, and apply the role to the
# user.
#
# "IAM access" permissions are needed to run this code.
# "STS assume role" permissions are needed to run this code. (Note: It might be
# necessary to
# create a custom policy).
#
# Returns:
# 0 - If successful.
# 1 - If an error occurred.
#####
function iam_create_user_assume_role() {
    {
        if [ "$IAM_OPERATIONS_SOURCED" != "True" ]; then

            source ./iam_operations.sh
        fi
    }

    echo_repeat "*" 88
    echo "Welcome to the IAM create user and assume role demo."
    echo
    echo "This demo will create an IAM user, create an IAM role, and apply the role to
the user."
    echo_repeat "*" 88
    echo

    echo -n "Enter a name for a new IAM user: "
    get_input
    user_name=$get_input_result

    local user_arn
    user_arn=$(iam_create_user -u "$user_name")

    # shellcheck disable=SC2181
    if [[ ${?} == 0 ]]; then
        echo "Created demo IAM user named $user_name"
    else
        errecho "$user_arn"
    fi
}

```

```
    errecho "The user failed to create. This demo will exit."
    return 1
fi

local access_key_response
access_key_response=$(iam_create_user_access_key -u "$user_name")
# shellcheck disable=SC2181
if [[ ${?} != 0 ]]; then
    errecho "The access key failed to create. This demo will exit."
    clean_up "$user_name"
    return 1
fi

IFS=$'\t ' read -r -a access_key_values <<<"$access_key_response"
local key_name=${access_key_values[0]}
local key_secret=${access_key_values[1]}

echo "Created access key named $key_name"

echo "Wait 10 seconds for the user to be ready."
sleep 10
echo_repeat "*" 88
echo

local iam_role_name
iam_role_name=$(generate_random_name "test-role")
echo "Creating a role named $iam_role_name with user $user_name as the principal."

local assume_role_policy_document="{
  \"Version\": \"2012-10-17\",
  \"Statement\": [{
    \"Effect\": \"Allow\",
    \"Principal\": {\"AWS\": \"$user_arn\"},
    \"Action\": \"sts:AssumeRole\"
  }]
}"

local role_arn
role_arn=$(iam_create_role -n "$iam_role_name" -p "$assume_role_policy_document")

# shellcheck disable=SC2181
if [ ${?} == 0 ]; then
    echo "Created IAM role named $iam_role_name"
else
```

```
errecho "The role failed to create. This demo will exit."
clean_up "$user_name" "$key_name"
return 1
fi

local policy_name
policy_name=$(generate_random_name "test-policy")
local policy_document="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [{
        \"Effect\": \"Allow\",
        \"Action\": \"s3:ListAllMyBuckets\",
        \"Resource\": \"arn:aws:s3:::*\"}]}"

local policy_arn
policy_arn=$(iam_create_policy -n "$policy_name" -p "$policy_document")
# shellcheck disable=SC2181
if [[ ${?} == 0 ]]; then
    echo "Created IAM policy named $policy_name"
else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name"
    return 1
fi

if (iam_attach_role_policy -n "$iam_role_name" -p "$policy_arn"); then
    echo "Attached policy $policy_arn to role $iam_role_name"
else
    errecho "The policy failed to attach."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
    return 1
fi

local assume_role_policy_document="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [{
        \"Effect\": \"Allow\",
        \"Action\": \"sts:AssumeRole\",
        \"Resource\": \"$role_arn\"}]}"

local assume_role_policy_name
assume_role_policy_name=$(generate_random_name "test-assume-role-")

# shellcheck disable=SC2181
```

```
local assume_role_policy_arn
assume_role_policy_arn=$(iam_create_policy -n "$assume_role_policy_name" -p
"$assume_role_policy_document")
# shellcheck disable=SC2181
if [ ${?} == 0 ]; then
    echo "Created IAM policy named $assume_role_policy_name for sts assume role"
else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
    return 1
fi

echo "Wait 10 seconds to give AWS time to propagate these new resources and
connections."
sleep 10
echo_repeat "*" 88
echo

echo "Try to list buckets without the new user assuming the role."
echo_repeat "*" 88
echo

# Set the environment variables for the created user.
# bashsupport disable=BP2001
export AWS_ACCESS_KEY_ID=$key_name
# bashsupport disable=BP2001
export AWS_SECRET_ACCESS_KEY=$key_secret

local buckets
buckets=$(s3_list_buckets)

# shellcheck disable=SC2181
if [ ${?} == 0 ]; then
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. This should not have
happened."
else
    errecho "Because the role with permissions has not been assumed, listing buckets
failed."
fi

echo
echo_repeat "*" 88
```

```
echo "Now assume the role $iam_role_name and list the buckets."
echo_repeat "*" 88
echo

local credentials

credentials=$(sts_assume_role -r "$role_arn" -n "AssumeRoleDemoSession")
# shellcheck disable=SC2181
if [ ${?} == 0 ]; then
    echo "Assumed role $iam_role_name"
else
    errecho "Failed to assume role."
    export AWS_ACCESS_KEY_ID=""
    export AWS_SECRET_ACCESS_KEY=""
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
"$assume_role_policy_arn"
    return 1
fi

IFS=$'\t ' read -r -a credentials <<<"$credentials"

export AWS_ACCESS_KEY_ID=${credentials[0]}
export AWS_SECRET_ACCESS_KEY=${credentials[1]}
# bashsupport disable=BP2001
export AWS_SESSION_TOKEN=${credentials[2]}

buckets=$(s3_list_buckets)

# shellcheck disable=SC2181
if [ ${?} == 0 ]; then
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. Listing buckets succeeded
because of "
    echo "the assumed role."
else
    errecho "Failed to list buckets. This should not happen."
    export AWS_ACCESS_KEY_ID=""
    export AWS_SECRET_ACCESS_KEY=""
    export AWS_SESSION_TOKEN=""
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
"$assume_role_policy_arn"
    return 1
fi
```



```

local result=0
export AWS_ACCESS_KEY_ID=""
export AWS_SECRET_ACCESS_KEY=""

echo
echo_repeat "*" 88
echo "The created resources will now be deleted."
echo_repeat "*" 88
echo

clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
"$assume_role_policy_arn"

# shellcheck disable=SC2181
if [[ ${?} -ne 0 ]]; then
    result=1
fi

return $result
}

```

The IAM functions used in this scenario.

```

#####
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
# (IAM) user already exists.
#
# Parameters:
#     $1 - The name of the IAM user to check.
#
# Returns:
#     0 - If the user already exists.
#     1 - If the user doesn't exist.
#####
function iam_user_exists() {
    local user_name
    user_name=$1

    # Check whether the IAM user already exists.

```

```

# We suppress all output - we're interested only in the return code.

local errors
errors=$(aws iam get-user \
  --user-name "$user_name" 2>&1 >/dev/null)

local error_code=${?}

if [[ $error_code -eq 0 ]]; then
  return 0 # 0 in Bash script means true.
else
  if [[ $errors != *"error"*(NoSuchEntity)* ]]; then
    aws_cli_error_log $error_code
    errecho "Error calling iam get-user $errors"
  fi

  return 1 # 1 in Bash script means false.
fi
}

#####
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#   -u user_name -- The name of the user to create.
#
# Returns:
#   The ARN of the user.
#   And:
#   0 - If successful.
#   1 - If it fails.
#####
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopt command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an WS Identity and Access Management (IAM) user. You must supply a
username:"

```

```
    echo "  -u user_name    The name of the user. It must be unique within the
account."
    echo ""
}

# Retrieve the calling parameters.
while getopts "u:h" option; do
    case "${option}" in
        u) user_name="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
fi

iecho "Parameters:\n"
iecho "  User name:  $user_name"
iecho ""

# If the user already exists, we don't want to try to create it.
if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
fi

response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

local error_code=${?}
```

```

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
fi

echo "$response"

return 0
}

#####
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#     -u user_name -- The name of the IAM user.
#     [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#     [access_key_id access_key_secret]
#     And:
#     0 - If successful.
#     1 - If it fails.
#####
function iam_create_user_access_key() {
    local user_name file_name response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_create_user_access_key"
        echo "Creates an AWS Identity and Access Management (IAM) key pair."
        echo "  -u user_name    The name of the IAM user."
        echo "  [-f file_name]  Optional file name for the access key output."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "u:f:h" option; do
        case "${option}" in
            u) user_name="${OPTARG}" ;;
            f) file_name="${OPTARG}" ;;
        esac
    done

```

```
h)
  usage
  return 0
  ;;
\?)
  echo "Invalid parameter"
  usage
  return 1
  ;;
esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

response=$(aws iam create-access-key \
  --user-name "$user_name" \
  --output text)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-access-key operation failed.$response"
  return 1
fi

if [[ -n "$file_name" ]]; then
  echo "$response" >"$file_name"
fi

local key_id key_secret
# shellcheck disable=SC2086
key_id=$(echo $response | cut -f 2 -d ' ')
# shellcheck disable=SC2086
key_secret=$(echo $response | cut -f 4 -d ' ')

echo "$key_id $key_secret"

return 0
```

```

}

#####
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#   -n role_name -- The name of the IAM role.
#   -p policy_json -- The assume role policy document.
#
# Returns:
#   The ARN of the role.
#   And:
#   0 - If successful.
#   1 - If it fails.
#####
function iam_create_role() {
    local role_name policy_document response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_create_user_access_key"
        echo "Creates an AWS Identity and Access Management (IAM) role."
        echo "  -n role_name  The name of the IAM role."
        echo "  -p policy_json -- The assume role policy document."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "n:p:h" option; do
        case "${option}" in
            n) role_name="${OPTARG}" ;;
            p) policy_document="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
}

```

```

    esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
fi

response=$(aws iam create-role \
    --role-name "$role_name" \
    --assume-role-policy-document "$policy_document" \
    --output text \
    --query Role.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
fi

echo "$response"

return 0
}

#####
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#     -n policy_name -- The name of the IAM policy.
#     -p policy_json -- The policy document.
#

```

```
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function iam_create_policy() {
    local policy_name policy_document response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_create_policy"
        echo "Creates an AWS Identity and Access Management (IAM) policy."
        echo "  -n policy_name  The name of the IAM policy."
        echo "  -p policy_json -- The policy document."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "n:p:h" option; do
        case "${option}" in
            n) policy_name="${OPTARG}" ;;
            p) policy_document="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    if [[ -z "$policy_name" ]]; then
        errecho "ERROR: You must provide a policy name with the -n parameter."
        usage
        return 1
    fi

    if [[ -z "$policy_document" ]]; then
        errecho "ERROR: You must provide a policy document with the -p parameter."
        usage
    fi
}
```



```

    return 1
fi

response=$(aws iam create-policy \
  --policy-name "$policy_name" \
  --policy-document "$policy_document" \
  --output text \
  --query Policy.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-policy operation failed.\n$response"
  return 1
fi

echo "$response"
}

#####
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a role.
#
# Parameters:
#   -n role_name -- The name of the IAM role.
#   -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#   0 - If successful.
#   1 - If it fails.
#####
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopt command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
role."
    echo "  -n role_name  The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
  }

```

```
    echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
    case "${option}" in
        n) role_name="${OPTARG}" ;;
        p) policy_arn="${OPTARG}" ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
fi

response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
fi
```

```

    echo "$response"

    return 0
}

#####
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a role.
#
# Parameters:
#     -n role_name -- The name of the IAM role.
#     -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function iam_detach_role_policy() {
    local role_name policy_arn response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_detach_role_policy"
        echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
role."
        echo "  -n role_name    The name of the IAM role."
        echo "  -p policy_ARN -- The IAM policy document ARN."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "n:p:h" option; do
        case "${option}" in
            n) role_name="${OPTARG}" ;;
            p) policy_arn="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"

```

```

        usage
        return 1
    ;;
esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
fi

if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
fi

response=$(aws iam detach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
    return 1
fi

echo "$response"

return 0
}

#####
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#     -n policy_arn -- The name of the IAM policy arn.
#

```

```

# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function iam_delete_policy() {
    local policy_arn response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_delete_policy"
        echo "Deletes an WS Identity and Access Management (IAM) policy"
        echo "  -n policy_arn -- The name of the IAM policy arn."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "n:h" option; do
        case "${option}" in
            n) policy_arn="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    if [[ -z "$policy_arn" ]]; then
        errecho "ERROR: You must provide a policy arn with the -n parameter."
        usage
        return 1
    fi

    iecho "Parameters:\n"
    iecho "  Policy arn: $policy_arn"
    iecho ""

    response=$(aws iam delete-policy \

```

```

    --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
fi

iecho "delete-policy response:$response"
iecho

return 0
}

#####
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#     -n role_name -- The name of the IAM role.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function iam_delete_role() {
    local role_name response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_delete_role"
        echo "Deletes an WS Identity and Access Management (IAM) role"
        echo "  -n role_name -- The name of the IAM role."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "n:h" option; do
        case "${option}" in
            n) role_name="${OPTARG}" ;;

```

```
h)
  usage
  return 0
  ;;
\?)
  echo "Invalid parameter"
  usage
  return 1
  ;;
esac
done
export OPTIND=1

echo "role_name:$role_name"
if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "  Role name: $role_name"
iecho ""

response=$(aws iam delete-role \
  --role-name "$role_name")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-role operation failed.\n$response"
  return 1
fi

iecho "delete-role response:$response"
iecho

return 0
}

#####
# function iam_delete_access_key
#
```

```

# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#     -u user_name  -- The name of the user.
#     -k access_key -- The access key to delete.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function iam_delete_access_key() {
    local user_name access_key response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_delete_access_key"
        echo "Deletes an WS Identity and Access Management (IAM) access key for the
specified IAM user"
        echo "  -u user_name    The name of the user."
        echo "  -k access_key    The access key to delete."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "u:k:h" option; do
        case "${option}" in
            u) user_name="${OPTARG}" ;;
            k) access_key="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    if [[ -z "$user_name" ]]; then
        errecho "ERROR: You must provide a username with the -u parameter."
    fi
}

```



```

    usage
    return 1
fi

if [[ -z "$access_key" ]]; then
    errecho "ERROR: You must provide an access key with the -k parameter."
    usage
    return 1
fi

iecho "Parameters:\n"
iecho "    Username:  $user_name"
iecho "    Access key:  $access_key"
iecho ""

response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
fi

iecho "delete-access-key response:$response"
iecho

return 0
}

#####
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#     -u user_name  -- The name of the user to create.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.

```

```
#####
function iam_delete_user() {
    local user_name response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_delete_user"
        echo "Deletes an WS Identity and Access Management (IAM) user. You must supply a
username:"
        echo "  -u user_name    The name of the user."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "u:h" option; do
        case "${option}" in
            u) user_name="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
    export OPTIND=1

    if [[ -z "$user_name" ]]; then
        errecho "ERROR: You must provide a username with the -u parameter."
        usage
        return 1
    fi

    iecho "Parameters:\n"
    iecho "  User name:  $user_name"
    iecho ""

    # If the user does not exist, we don't want to try to delete it.
    if (! iam_user_exists "$user_name"); then
        errecho "ERROR: A user with that name does not exist in the account."
    fi
}
#####
```

```
    return 1
fi

response=$(aws iam delete-user \
  --user-name "$user_name")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-user operation failed.$response"
  return 1
fi

iecho "delete-user response:$response"
iecho

return 0
}
```

- For API details, see the following topics in *AWS CLI Command Reference*.
 - [AttachRolePolicy](#)
 - [CreateAccessKey](#)
 - [CreatePolicy](#)
 - [CreateRole](#)
 - [CreateUser](#)
 - [DeleteAccessKey](#)
 - [DeletePolicy](#)
 - [DeleteRole](#)
 - [DeleteUser](#)
 - [DeleteUserPolicy](#)
 - [DetachRolePolicy](#)
 - [PutUserPolicy](#)

Amazon S3 examples using AWS CLI with Bash script

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Bash script with Amazon S3.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)
- [Scenarios](#)

Actions

CopyObject

The following code example shows how to use CopyObject.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####  
# function errecho  
#  
# This function outputs everything sent to it to STDERR (standard error output).  
#####  
function errecho() {
```

```

    printf "%s\n" "$*" 1>&2
}

#####
# function copy_item_in_bucket
#
# This function creates a copy of the specified file in the same bucket.
#
# Parameters:
#     $1 - The name of the bucket to copy the file from and to.
#     $2 - The key of the source file to copy.
#     $3 - The key of the destination file.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function copy_item_in_bucket() {
    local bucket_name=$1
    local source_key=$2
    local destination_key=$3
    local response

    response=$(aws s3api copy-object \
        --bucket "$bucket_name" \
        --copy-source "$bucket_name/$source_key" \
        --key "$destination_key")

    # shellcheck disable=SC2181
    if [[ $? -ne 0 ]]; then
        errecho "ERROR: AWS reports s3api copy-object operation failed.\n$response"
        return 1
    fi
}

```

- For API details, see [CopyObject](#) in *AWS CLI Command Reference*.

CreateBucket

The following code example shows how to use CreateBucket.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#####
function iecho() {
    if [[ $VERBOSE == true ]]; then
        echo "$@"
    fi
}

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function create-bucket
#
# This function creates the specified bucket in the specified AWS Region, unless
# it already exists.
#
# Parameters:
#     -b bucket_name -- The name of the bucket to create.
#     -r region_code -- The code for an AWS Region in which to
#                       create the bucket.
#
# Returns:
#     The URL of the bucket that was created.
```

```

# And:
# 0 - If successful.
# 1 - If it fails.
#####
function create_bucket() {
    local bucket_name region_code response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function create_bucket"
        echo "Creates an Amazon S3 bucket. You must supply a bucket name:"
        echo "  -b bucket_name    The name of the bucket. It must be globally unique."
        echo "  [-r region_code]    The code for an AWS Region in which the bucket is
created."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "b:r:h" option; do
        case "${option}" in
            b) bucket_name="${OPTARG}" ;;
            r) region_code="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done

    if [[ -z "$bucket_name" ]]; then
        errecho "ERROR: You must provide a bucket name with the -b parameter."
        usage
        return 1
    fi

    local bucket_config_arg
    # A location constraint for "us-east-1" returns an error.
    if [[ -n "$region_code" ]] && [[ "$region_code" != "us-east-1" ]]; then

```

```
    bucket_config_arg="--create-bucket-configuration LocationConstraint=
$region_code"
fi

iecho "Parameters:\n"
iecho "    Bucket name:  $bucket_name"
iecho "    Region code:  $region_code"
iecho ""

# If the bucket already exists, we don't want to try to create it.
if (bucket_exists "$bucket_name"); then
    errecho "ERROR: A bucket with that name already exists. Try again."
    return 1
fi

# shellcheck disable=SC2086
response=$(aws s3api create-bucket \
    --bucket "$bucket_name" \
    $bucket_config_arg)

# shellcheck disable=SC2181
if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports create-bucket operation failed.\n$response"
    return 1
fi
}
```

- For API details, see [CreateBucket](#) in *AWS CLI Command Reference*.

DeleteBucket

The following code example shows how to use DeleteBucket.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).


```
#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function delete_bucket
#
# This function deletes the specified bucket.
#
# Parameters:
#     $1 - The name of the bucket.

# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function delete_bucket() {
    local bucket_name=$1
    local response

    response=$(aws s3api delete-bucket \
        --bucket "$bucket_name")

    # shellcheck disable=SC2181
    if [[ $? -ne 0 ]]; then
        errecho "ERROR: AWS reports s3api delete-bucket failed.\n$response"
        return 1
    fi
}

```

- For API details, see [DeleteBucket](#) in *AWS CLI Command Reference*.

DeleteObject

The following code example shows how to use DeleteObject.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function delete_item_in_bucket
#
# This function deletes the specified file from the specified bucket.
#
# Parameters:
#     $1 - The name of the bucket.
#     $2 - The key (file name) in the bucket to delete.

# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function delete_item_in_bucket() {
    local bucket_name=$1
    local key=$2
    local response

    response=$(aws s3api delete-object \
        --bucket "$bucket_name" \
        --key "$key")

    # shellcheck disable=SC2181
    if [[ $? -ne 0 ]]; then
        errecho "ERROR: AWS reports s3api delete-object operation failed.\n$response"
    fi
}

```

```

    return 1
  fi
}

```

- For API details, see [DeleteObject](#) in *AWS CLI Command Reference*.

DeleteObjects

The following code example shows how to use DeleteObjects.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#####
# function delete_items_in_bucket
#
# This function deletes the specified list of keys from the specified bucket.
#
# Parameters:
#   $1 - The name of the bucket.
#   $2 - A list of keys in the bucket to delete.

# Returns:
#   0 - If successful.
#   1 - If it fails.
#####

```

```
function delete_items_in_bucket() {
    local bucket_name=$1
    local keys=$2
    local response

    # Create the JSON for the items to delete.
    local delete_items
    delete_items="{\"Objects\":["
    for key in $keys; do
        delete_items="$delete_items{\"Key\": \"$key\"},"
    done
    delete_items=${delete_items%?} # Remove the final comma.
    delete_items="$delete_items]"

    response=$(aws s3api delete-objects \
        --bucket "$bucket_name" \
        --delete "$delete_items")

    # shellcheck disable=SC2181
    if [[ $? -ne 0 ]]; then
        errecho "ERROR: AWS reports s3api delete-object operation failed.\n$response"
        return 1
    fi
}
```

- For API details, see [DeleteObjects](#) in *AWS CLI Command Reference*.

GetObject

The following code example shows how to use GetObject.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
```

```

# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function download_object_from_bucket
#
# This function downloads an object in a bucket to a file.
#
# Parameters:
#     $1 - The name of the bucket to download the object from.
#     $2 - The path and file name to store the downloaded bucket.
#     $3 - The key (name) of the object in the bucket.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function download_object_from_bucket() {
    local bucket_name=$1
    local destination_file_name=$2
    local object_name=$3
    local response

    response=$(aws s3api get-object \
        --bucket "$bucket_name" \
        --key "$object_name" \
        "$destination_file_name")

    # shellcheck disable=SC2181
    if [[ ${?} -ne 0 ]]; then
        errecho "ERROR: AWS reports put-object operation failed.\n$response"
        return 1
    fi
}

```

- For API details, see [GetObject](#) in *AWS CLI Command Reference*.

HeadBucket

The following code example shows how to use HeadBucket.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function bucket_exists
#
# This function checks to see if the specified bucket already exists.
#
# Parameters:
#     $1 - The name of the bucket to check.
#
# Returns:
#     0 - If the bucket already exists.
#     1 - If the bucket doesn't exist.
#####
function bucket_exists() {
    local bucket_name
    bucket_name=$1

    # Check whether the bucket already exists.
    # We suppress all output - we're interested only in the return code.

    if aws s3api head-bucket \
        --bucket "$bucket_name" \
        >/dev/null 2>&1; then
        return 0 # 0 in Bash script means true.
    else
        return 1 # 1 in Bash script means false.
    fi
}
```

- For API details, see [HeadBucket](#) in *AWS CLI Command Reference*.

ListObjectsV2

The following code example shows how to use ListObjectsV2.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function list_items_in_bucket
#
# This function displays a list of the files in the bucket with each file's
# size. The function uses the --query parameter to retrieve only the key and
# size fields from the Contents collection.
#
# Parameters:
#     $1 - The name of the bucket.
#
# Returns:
#     The list of files in text format.
#
# And:
#     0 - If successful.
#     1 - If it fails.
#####
function list_items_in_bucket() {
    local bucket_name=$1
    local response

    response=$(aws s3api list-objects \
        --bucket "$bucket_name" \
```

```

--output text \
--query 'Contents[].{Key: Key, Size: Size}')
```

```

# shellcheck disable=SC2181
if [[ ${?} -eq 0 ]]; then
    echo "$response"
else
    errecho "ERROR: AWS reports s3api list-objects operation failed.\n$response"
    return 1
fi
}
```

- For API details, see [ListObjectsV2](#) in *AWS CLI Command Reference*.

PutObject

The following code example shows how to use PutObject.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function copy_file_to_bucket
#
# This function creates a file in the specified bucket.
#
```



```

# Parameters:
#     $1 - The name of the bucket to copy the file to.
#     $2 - The path and file name of the local file to copy to the bucket.
#     $3 - The key (name) to call the copy of the file in the bucket.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function copy_file_to_bucket() {
    local response bucket_name source_file destination_file_name
    bucket_name=$1
    source_file=$2
    destination_file_name=$3

    response=$(aws s3api put-object \
        --bucket "$bucket_name" \
        --body "$source_file" \
        --key "$destination_file_name")

    # shellcheck disable=SC2181
    if [[ ${?} -ne 0 ]]; then
        errecho "ERROR: AWS reports put-object operation failed.\n$response"
        return 1
    fi
}

```

- For API details, see [PutObject](#) in *AWS CLI Command Reference*.

Scenarios

Get started with buckets and objects

The following code example shows how to:

- Create a bucket and upload a file to it.
- Download an object from a bucket.
- Copy an object to a subfolder in a bucket.
- List the objects in a bucket.
- Delete the bucket objects and the bucket.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function s3_getting_started
#
# This function creates, copies, and deletes S3 buckets and objects.
#
# Returns:
#     0 - If successful.
#     1 - If an error occurred.
#####
function s3_getting_started() {
    {
        if [ "$BUCKET_OPERATIONS_SOURCED" != "True" ]; then
            cd bucket-lifecycle-operations || exit

            source ./bucket_operations.sh
            cd ..
        fi
    }

    echo_repeat "*" 88
    echo "Welcome to the Amazon S3 getting started demo."
    echo_repeat "*" 88

    local bucket_name
    bucket_name=$(generate_random_name "doc-example-bucket")

    local region_code
    region_code=$(aws configure get region)

    if create_bucket -b "$bucket_name" -r "$region_code"; then
        echo "Created demo bucket named $bucket_name"
    else
        errecho "The bucket failed to create. This demo will exit."
        return 1
    fi
}
```

```
fi

local file_name
while [ -z "$file_name" ]; do
    echo -n "Enter a file you want to upload to your bucket: "
    get_input
    file_name=$get_input_result

    if [ ! -f "$file_name" ]; then
        echo "Could not find file $file_name. Are you sure it exists?"
        file_name=""
    fi
done

local key
key="$(basename "$file_name")"

local result=0
if copy_file_to_bucket "$bucket_name" "$file_name" "$key"; then
    echo "Uploaded file $file_name into bucket $bucket_name with key $key."
else
    result=1
fi

local destination_file
destination_file="$file_name.download"
if yes_no_input "Would you like to download $key to the file $destination_file?
(y/n) "; then
    if download_object_from_bucket "$bucket_name" "$destination_file" "$key"; then
        echo "Downloaded $key in the bucket $bucket_name to the file
$destination_file."
    else
        result=1
    fi
fi

if yes_no_input "Would you like to copy $key a new object key in your bucket? (y/
n) "; then
    local to_key
    to_key="demo/$key"
    if copy_item_in_bucket "$bucket_name" "$key" "$to_key"; then
        echo "Copied $key in the bucket $bucket_name to the $to_key."
    else
        result=1
    fi
fi
```

```

    fi
fi

local bucket_items
bucket_items=$(list_items_in_bucket "$bucket_name")

# shellcheck disable=SC2181
if [[ $? -ne 0 ]]; then
    result=1
fi

echo "Your bucket contains the following items."
echo -e "Name\t\tSize"
echo "$bucket_items"

if yes_no_input "Delete the bucket, $bucket_name, as well as the objects in it?
(y/n) "; then
    bucket_items=$(echo "$bucket_items" | cut -f 1)

    if delete_items_in_bucket "$bucket_name" "$bucket_items"; then
        echo "The following items were deleted from the bucket $bucket_name"
        echo "$bucket_items"
    else
        result=1
    fi

    if delete_bucket "$bucket_name"; then
        echo "Deleted the bucket $bucket_name"
    else
        result=1
    fi
fi

return $result
}

```

The Amazon S3 functions used in this scenario.

```

#####
# function create-bucket
#
# This function creates the specified bucket in the specified AWS Region, unless

```

```
# it already exists.
#
# Parameters:
#     -b bucket_name  -- The name of the bucket to create.
#     -r region_code  -- The code for an AWS Region in which to
#                       create the bucket.
#
# Returns:
#     The URL of the bucket that was created.
#     And:
#     0 - If successful.
#     1 - If it fails.
#####
function create_bucket() {
    local bucket_name region_code response
    local option OPTARG # Required to use getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function create_bucket"
        echo "Creates an Amazon S3 bucket. You must supply a bucket name:"
        echo "  -b bucket_name    The name of the bucket. It must be globally unique."
        echo "  [-r region_code]  The code for an AWS Region in which the bucket is
created."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "b:r:h" option; do
        case "${option}" in
            b) bucket_name="${OPTARG}" ;;
            r) region_code="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
                ;;
        esac
    done
```

```

if [[ -z "$bucket_name" ]]; then
    errecho "ERROR: You must provide a bucket name with the -b parameter."
    usage
    return 1
fi

local bucket_config_arg
# A location constraint for "us-east-1" returns an error.
if [[ -n "$region_code" ]] && [[ "$region_code" != "us-east-1" ]]; then
    bucket_config_arg="--create-bucket-configuration LocationConstraint=
$region_code"
fi

iecho "Parameters:\n"
iecho "    Bucket name:    $bucket_name"
iecho "    Region code:    $region_code"
iecho ""

# If the bucket already exists, we don't want to try to create it.
if (bucket_exists "$bucket_name"); then
    errecho "ERROR: A bucket with that name already exists. Try again."
    return 1
fi

# shellcheck disable=SC2086
response=$(aws s3api create-bucket \
    --bucket "$bucket_name" \
    $bucket_config_arg)

# shellcheck disable=SC2181
if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports create-bucket operation failed.\n$response"
    return 1
fi
}

#####
# function copy_file_to_bucket
#
# This function creates a file in the specified bucket.
#
# Parameters:
#     $1 - The name of the bucket to copy the file to.
#     $2 - The path and file name of the local file to copy to the bucket.

```

```

#      $3 - The key (name) to call the copy of the file in the bucket.
#
# Returns:
#      0 - If successful.
#      1 - If it fails.
#####
function copy_file_to_bucket() {
    local response bucket_name source_file destination_file_name
    bucket_name=$1
    source_file=$2
    destination_file_name=$3

    response=$(aws s3api put-object \
        --bucket "$bucket_name" \
        --body "$source_file" \
        --key "$destination_file_name")

    # shellcheck disable=SC2181
    if [[ ${?} -ne 0 ]]; then
        errecho "ERROR: AWS reports put-object operation failed.\n$response"
        return 1
    fi
}

#####
# function download_object_from_bucket
#
# This function downloads an object in a bucket to a file.
#
# Parameters:
#      $1 - The name of the bucket to download the object from.
#      $2 - The path and file name to store the downloaded bucket.
#      $3 - The key (name) of the object in the bucket.
#
# Returns:
#      0 - If successful.
#      1 - If it fails.
#####
function download_object_from_bucket() {
    local bucket_name=$1
    local destination_file_name=$2
    local object_name=$3
    local response

```

```

response=$(aws s3api get-object \
  --bucket "$bucket_name" \
  --key "$object_name" \
  "$destination_file_name")

# shellcheck disable=SC2181
if [[ ${?} -ne 0 ]]; then
  errecho "ERROR: AWS reports put-object operation failed.\n$response"
  return 1
fi
}

#####
# function copy_item_in_bucket
#
# This function creates a copy of the specified file in the same bucket.
#
# Parameters:
#   $1 - The name of the bucket to copy the file from and to.
#   $2 - The key of the source file to copy.
#   $3 - The key of the destination file.
#
# Returns:
#   0 - If successful.
#   1 - If it fails.
#####
function copy_item_in_bucket() {
  local bucket_name=$1
  local source_key=$2
  local destination_key=$3
  local response

  response=$(aws s3api copy-object \
    --bucket "$bucket_name" \
    --copy-source "$bucket_name/$source_key" \
    --key "$destination_key")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR: AWS reports s3api copy-object operation failed.\n$response"
    return 1
  fi
}

```



```
#####
# function list_items_in_bucket
#
# This function displays a list of the files in the bucket with each file's
# size. The function uses the --query parameter to retrieve only the key and
# size fields from the Contents collection.
#
# Parameters:
#     $1 - The name of the bucket.
#
# Returns:
#     The list of files in text format.
#     And:
#     0 - If successful.
#     1 - If it fails.
#####
function list_items_in_bucket() {
    local bucket_name=$1
    local response

    response=$(aws s3api list-objects \
        --bucket "$bucket_name" \
        --output text \
        --query 'Contents[].{Key: Key, Size: Size}')

    # shellcheck disable=SC2181
    if [[ ${?} -eq 0 ]]; then
        echo "$response"
    else
        errecho "ERROR: AWS reports s3api list-objects operation failed.\n$response"
        return 1
    fi
}

#####
# function delete_items_in_bucket
#
# This function deletes the specified list of keys from the specified bucket.
#
# Parameters:
#     $1 - The name of the bucket.
#     $2 - A list of keys in the bucket to delete.
#
# Returns:
```

```

#      0 - If successful.
#      1 - If it fails.
#####
function delete_items_in_bucket() {
    local bucket_name=$1
    local keys=$2
    local response

    # Create the JSON for the items to delete.
    local delete_items
    delete_items="{\"Objects\":["
    for key in $keys; do
        delete_items="$delete_items{\"Key\": \"$key\"},"
    done
    delete_items=${delete_items%?} # Remove the final comma.
    delete_items="$delete_items]"

    response=$(aws s3api delete-objects \
        --bucket "$bucket_name" \
        --delete "$delete_items")

    # shellcheck disable=SC2181
    if [[ $? -ne 0 ]]; then
        errecho "ERROR: AWS reports s3api delete-object operation failed.\n$response"
        return 1
    fi
}

#####
# function delete_bucket
#
# This function deletes the specified bucket.
#
# Parameters:
#      $1 - The name of the bucket.
#
# Returns:
#      0 - If successful.
#      1 - If it fails.
#####
function delete_bucket() {
    local bucket_name=$1
    local response

```

```
response=$(aws s3api delete-bucket \  
  --bucket "$bucket_name")  
  
# shellcheck disable=SC2181  
if [[ $? -ne 0 ]]; then  
  errecho "ERROR: AWS reports s3api delete-bucket failed.\n$response"  
  return 1  
fi  
}
```

- For API details, see the following topics in *AWS CLI Command Reference*.
 - [CopyObject](#)
 - [CreateBucket](#)
 - [DeleteBucket](#)
 - [DeleteObjects](#)
 - [GetObject](#)
 - [ListObjectsV2](#)
 - [PutObject](#)

AWS STS examples using AWS CLI with Bash script

The following code examples show you how to perform actions and implement common scenarios by using the AWS Command Line Interface with Bash script with AWS STS.

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

Scenarios are code examples that show you how to accomplish a specific task by calling multiple functions within the same service.

Each example includes a link to GitHub, where you can find instructions on how to set up and run the code in context.

Topics

- [Actions](#)

Actions

AssumeRole

The following code example shows how to use AssumeRole.

AWS CLI with Bash script

Note

There's more on GitHub. Find the complete example and learn how to set up and run in the [AWS Code Examples Repository](#).

```
#####
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#####
function iecho() {
    if [[ $VERBOSE == true ]]; then
        echo "$@"
    fi
}

#####
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#####
function errecho() {
    printf "%s\n" "$*" 1>&2
}

#####
# function sts_assume_role
#
# This function assumes a role in the AWS account and returns the temporary
# credentials.
#
# Parameters:
#     -n role_session_name -- The name of the session.
```

```

#     -r role_arn -- The ARN of the role to assume.
#
# Returns:
#     [access_key_id, secret_access_key, session_token]
#     And:
#     0 - If successful.
#     1 - If an error occurred.
#####
function sts_assume_role() {
    local role_session_name role_arn response
    local option OPTARG # Required to use getopt command in a function.

# bashsupport disable=BP5008
function usage() {
    echo "function sts_assume_role"
    echo "Assumes a role in the AWS account and returns the temporary credentials:"
    echo "  -n role_session_name -- The name of the session."
    echo "  -r role_arn -- The ARN of the role to assume."
    echo ""
}

while getopt n:r:h option; do
    case "${option}" in
        n) role_session_name=${OPTARG} ;;
        r) role_arn=${OPTARG} ;;
        h)
            usage
            return 0
            ;;
        \?)
            echo "Invalid parameter"
            usage
            return 1
            ;;
    esac
done

response=$(aws sts assume-role \
    --role-session-name "$role_session_name" \
    --role-arn "$role_arn" \
    --output text \
    --query "Credentials.[AccessKeyId, SecretAccessKey, SessionToken]")

local error_code=${?}

```

```
if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-role operation failed.\n$response"
  return 1
fi

echo "$response"

return 0
}
```

- For API details, see [AssumeRole](#) in *AWS CLI Command Reference*.

Security in the AWS Command Line Interface

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The [shared responsibility model](#) describes this as security *of* the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the [AWS Compliance Programs](#). To learn about the compliance programs that apply to AWS Command Line Interface, see [AWS Services in Scope by Compliance Program](#).
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using the AWS Command Line Interface (AWS CLI). The following topics show you how to configure the AWS CLI to meet your security and compliance objectives. You also learn how to use the AWS CLI to help you to monitor and secure your AWS resources.

Topics

- [Data protection in the AWS CLI](#)
- [Identity and Access Management](#)
- [Compliance Validation for this AWS Product or Service](#)
- [Resilience for this AWS Product or Service](#)
- [Infrastructure Security for this AWS Product or Service](#)
- [Enforce a minimum version of TLS](#)

Data protection in the AWS CLI

The AWS [shared responsibility model](#) applies to data protection in AWS Command Line Interface. As described in this model, AWS is responsible for protecting the global infrastructure that runs all

of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. You are also responsible for the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the [Data Privacy FAQ](#). For information about data protection in Europe, see the [AWS Shared Responsibility Model and GDPR](#) blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual users with AWS IAM Identity Center or AWS Identity and Access Management (IAM). That way, each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We require TLS 1.2 and recommend TLS 1.3.
- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.
- If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see [Federal Information Processing Standard \(FIPS\) 140-2](#).

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form text fields such as a **Name** field. This includes when you work with AWS CLI or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form text fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

Data encryption

A key feature of any secure service is that information is encrypted when it is not being actively used.

Encryption at rest

The AWS CLI does not itself store any customer data other than the credentials it needs to interact with the AWS services on the user's behalf.

If you use the AWS CLI to invoke an AWS service that transmits customer data to your local computer for storage, then refer to the Security & Compliance chapter in that service's User Guide for information on how that data is stored, protected, and encrypted.

Encryption in transit

By default, all data transmitted from the client computer running the AWS CLI and AWS service endpoints is encrypted by sending everything through a HTTPS/TLS connection.

You don't need to do anything to enable the use of HTTPS/TLS. It is always enabled unless you explicitly disable it for an individual command by using the `--no-verify-ssl` command line option.

Identity and Access Management

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use AWS resources. IAM is an AWS service that you can use with no additional charge.

Topics

- [Audience](#)
- [Authenticating with identities](#)
- [Managing access using policies](#)
- [How AWS services work with IAM](#)
- [Troubleshooting AWS identity and access](#)

Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work that you do in AWS.

Service user – If you use AWS services to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more AWS features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in AWS, see [Troubleshooting AWS identity and access](#) or the user guide of the AWS service you are using.

Service administrator – If you're in charge of AWS resources at your company, you probably have full access to AWS. It's your job to determine which AWS features and resources your service users should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM. To learn more about how your company can use IAM with AWS, see the user guide of the AWS service you are using.

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to AWS. To view example AWS identity-based policies that you can use in IAM, see the user guide of the AWS service you are using.

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. You must be *authenticated* (signed in to AWS) as the AWS account root user, as an IAM user, or by assuming an IAM role.

You can sign in to AWS as a federated identity by using credentials provided through an identity source. AWS IAM Identity Center (IAM Identity Center) users, your company's single sign-on authentication, and your Google or Facebook credentials are examples of federated identities. When you sign in as a federated identity, your administrator previously set up identity federation using IAM roles. When you access AWS by using federation, you are indirectly assuming a role.

Depending on the type of user you are, you can sign in to the AWS Management Console or the AWS access portal. For more information about signing in to AWS, see [How to sign in to your AWS account](#) in the *AWS Sign-In User Guide*.

If you access AWS programmatically, AWS provides a software development kit (SDK) and a command line interface (CLI) to cryptographically sign your requests by using your credentials. If you don't use AWS tools, you must sign requests yourself. For more information about using the recommended method to sign requests yourself, see [Signing AWS API requests](#) in the *IAM User Guide*.

Regardless of the authentication method that you use, you might be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see [Multi-factor authentication](#) in the *AWS IAM Identity Center User Guide* and [Using multi-factor authentication \(MFA\) in AWS](#) in the *IAM User Guide*.

AWS account root user

When you create an AWS account, you begin with one sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you don't use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform the tasks that only the root user can perform. For the complete list of tasks that require you to sign in as the root user, see [Tasks that require root user credentials](#) in the *IAM User Guide*.

Federated identity

As a best practice, require human users, including users that require administrator access, to use federation with an identity provider to access AWS services by using temporary credentials.

A *federated identity* is a user from your enterprise user directory, a web identity provider, the AWS Directory Service, the Identity Center directory, or any user that accesses AWS services by using credentials provided through an identity source. When federated identities access AWS accounts, they assume roles, and the roles provide temporary credentials.

For centralized access management, we recommend that you use AWS IAM Identity Center. You can create users and groups in IAM Identity Center, or you can connect and synchronize to a set of users and groups in your own identity source for use across all your AWS accounts and applications. For information about IAM Identity Center, see [What is IAM Identity Center?](#) in the *AWS IAM Identity Center User Guide*.

IAM users and groups

An [IAM user](#) is an identity within your AWS account that has specific permissions for a single person or application. Where possible, we recommend relying on temporary credentials instead of creating IAM users who have long-term credentials such as passwords and access keys. However, if you have specific use cases that require long-term credentials with IAM users, we recommend that you rotate access keys. For more information, see [Rotate access keys regularly for use cases that require long-term credentials](#) in the *IAM User Guide*.

An [IAM group](#) is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named *IAMAdmins* and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see [When to create an IAM user \(instead of a role\)](#) in the *IAM User Guide*.

IAM roles

An [IAM role](#) is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. You can temporarily assume an IAM role in the AWS Management Console by [switching roles](#). You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see [Using IAM roles](#) in the *IAM User Guide*.

IAM roles with temporary credentials are useful in the following situations:

- **Federated user access** – To assign permissions to a federated identity, you create a role and define permissions for the role. When a federated identity authenticates, the identity is associated with the role and is granted the permissions that are defined by the role. For information about roles for federation, see [Creating a role for a third-party Identity Provider](#) in the *IAM User Guide*. If you use IAM Identity Center, you configure a permission set. To control what your identities can access after they authenticate, IAM Identity Center correlates the permission set to a role in IAM. For information about permission sets, see [Permission sets](#) in the *AWS IAM Identity Center User Guide*.
- **Temporary IAM user permissions** – An IAM user or role can assume an IAM role to temporarily take on different permissions for a specific task.
- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see [Cross account resource access in IAM](#) in the *IAM User Guide*.
- **Cross-service access** – Some AWS services use features in other AWS services. For example, when you make a call in a service, it's common for that service to run applications in Amazon EC2 or store objects in Amazon S3. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.
- **Forward access sessions (FAS)** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. When you use some services, you might perform an action that then initiates another action in a different service. FAS uses the permissions of the

principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. FAS requests are only made when a service receives a request that requires interactions with other AWS services or resources to complete. In this case, you must have permissions to perform both actions. For policy details when making FAS requests, see [Forward access sessions](#).

- **Service role** – A service role is an [IAM role](#) that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see [Creating a role to delegate permissions to an AWS service](#) in the *IAM User Guide*.
- **Service-linked role** – A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.
- **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see [Using an IAM role to grant permissions to applications running on Amazon EC2 instances](#) in the *IAM User Guide*.

To learn whether to use IAM roles or IAM users, see [When to create an IAM role \(instead of a user\)](#) in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to AWS identities or resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when a principal (user, root user, or role session) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see [Overview of JSON policies](#) in the *IAM User Guide*.

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

By default, users and roles have no permissions. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see [Creating IAM policies](#) in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see [Choosing between managed policies and inline policies](#) in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must [specify a principal](#) in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

Access control lists (ACLs)

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see [Access control list \(ACL\) overview](#) in the *Amazon Simple Storage Service Developer Guide*.

Other policy types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of an entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see [Permissions boundaries for IAM entities](#) in the *IAM User Guide*.
- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see [How SCPs work](#) in the *AWS Organizations User Guide*.
- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see [Session policies](#) in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see [Policy evaluation logic](#) in the *IAM User Guide*.

How AWS services work with IAM

To get a high-level view of how AWS services work with most IAM features, see [AWS services that work with IAM](#) in the *IAM User Guide*.

To learn how to use a specific AWS service with IAM, see the security section of the relevant service's User Guide.

Troubleshooting AWS identity and access

Use the following information to help you diagnose and fix common issues that you might encounter when working with AWS and IAM.

Topics

- [I am not authorized to perform an action in AWS](#)
- [I am not authorized to perform iam:PassRole](#)
- [I want to allow people outside of my AWS account to access my AWS resources](#)

I am not authorized to perform an action in AWS

If you receive an error that you're not authorized to perform an action, your policies must be updated to allow you to perform the action.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a fictional `my-example-widget` resource but doesn't have the fictional `aws:GetWidget` permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
aws:GetWidget on resource: my-example-widget
```

In this case, the policy for the `mateojackson` user must be updated to allow access to the `my-example-widget` resource by using the `aws:GetWidget` action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I am not authorized to perform iam:PassRole

If you receive an error that you're not authorized to perform the `iam:PassRole` action, your policies must be updated to allow you to pass a role to AWS.

Some AWS services allow you to pass an existing role to that service instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in AWS. However, the action requires the service to have permissions that are granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
iam:PassRole
```

In this case, Mary's policies must be updated to allow her to perform the `iam:PassRole` action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I want to allow people outside of my AWS account to access my AWS resources

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.

To learn more, consult the following:

- To learn whether AWS supports these features, see [How AWS services work with IAM](#).
- To learn how to provide access to your resources across AWS accounts that you own, see [Providing access to an IAM user in another AWS account that you own](#) in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party AWS accounts, see [Providing access to AWS accounts owned by third parties](#) in the *IAM User Guide*.
- To learn how to provide access through identity federation, see [Providing access to externally authenticated users \(identity federation\)](#) in the *IAM User Guide*.

- To learn the difference between using roles and resource-based policies for cross-account access, see [Cross account resource access in IAM](#) in the *IAM User Guide*.

Compliance Validation for this AWS Product or Service

To learn whether an AWS service is within the scope of specific compliance programs, see [AWS services in Scope by Compliance Program](#) and choose the compliance program that you are interested in. For general information, see [AWS Compliance Programs](#).

You can download third-party audit reports using AWS Artifact. For more information, see [Downloading Reports in AWS Artifact](#).

Your compliance responsibility when using AWS services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- [Security and Compliance Quick Start Guides](#) – These deployment guides discuss architectural considerations and provide steps for deploying baseline environments on AWS that are security and compliance focused.
- [Architecting for HIPAA Security and Compliance on Amazon Web Services](#) – This whitepaper describes how companies can use AWS to create HIPAA-eligible applications.

Note

Not all AWS services are HIPAA eligible. For more information, see the [HIPAA Eligible Services Reference](#).

- [AWS Compliance Resources](#) – This collection of workbooks and guides might apply to your industry and location.
- [AWS Customer Compliance Guides](#) – Understand the shared responsibility model through the lens of compliance. The guides summarize the best practices for securing AWS services and map the guidance to security controls across multiple frameworks (including National Institute of Standards and Technology (NIST), Payment Card Industry Security Standards Council (PCI), and International Organization for Standardization (ISO)).
- [Evaluating Resources with Rules](#) in the *AWS Config Developer Guide* – The AWS Config service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.

- [AWS Security Hub](#) – This AWS service provides a comprehensive view of your security state within AWS. Security Hub uses security controls to evaluate your AWS resources and to check your compliance against security industry standards and best practices. For a list of supported services and controls, see [Security Hub controls reference](#).
- [Amazon GuardDuty](#) – This AWS service detects potential threats to your AWS accounts, workloads, containers, and data by monitoring your environment for suspicious and malicious activities. GuardDuty can help you address various compliance requirements, like PCI DSS, by meeting intrusion detection requirements mandated by certain compliance frameworks.
- [AWS Audit Manager](#) – This AWS service helps you continuously audit your AWS usage to simplify how you manage risk and compliance with regulations and industry standards.

This AWS product or service follows the [shared responsibility model](#) through the specific Amazon Web Services (AWS) services it supports. For AWS service security information, see the [AWS service security documentation page](#) and [AWS services that are in scope of AWS compliance efforts by compliance program](#).

Resilience for this AWS Product or Service

The AWS global infrastructure is built around AWS Regions and Availability Zones.

AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking.

With Availability Zones, you can design and operate applications and databases that automatically fail over between zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see [AWS Global Infrastructure](#).

This AWS product or service follows the [shared responsibility model](#) through the specific Amazon Web Services (AWS) services it supports. For AWS service security information, see the [AWS service security documentation page](#) and [AWS services that are in scope of AWS compliance efforts by compliance program](#).

Infrastructure Security for this AWS Product or Service

This AWS product or service uses managed services, and therefore is protected by the AWS global network security. For information about AWS security services and how AWS protects

infrastructure, see [AWS Cloud Security](#). To design your AWS environment using the best practices for infrastructure security, see [Infrastructure Protection](#) in *Security Pillar AWS Well-Architected Framework*.

You use AWS published API calls to access this AWS Product or Service through the network. Clients must support the following:

- Transport Layer Security (TLS). We require TLS 1.2 and recommend TLS 1.3.
- Cipher suites with perfect forward secrecy (PFS) such as DHE (Ephemeral Diffie-Hellman) or ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the [AWS Security Token Service](#) (AWS STS) to generate temporary security credentials to sign requests.

This AWS product or service follows the [shared responsibility model](#) through the specific Amazon Web Services (AWS) services it supports. For AWS service security information, see the [AWS service security documentation page](#) and [AWS services that are in scope of AWS compliance efforts by compliance program](#).

Enforce a minimum version of TLS

To add increased security when communicating with AWS services, you should use TLS 1.2 or later. When you use the AWS CLI, Python is used to set the TLS version.

AWS CLI version 2 uses an internal Python script that's compiled to use a minimum of TLS 1.2 when the service it's talking to supports it. As long as you use version 2 of the AWS CLI, no further steps are needed to enforce this minimum.

Troubleshoot AWS CLI errors

This section covers common errors and troubleshooting steps to follow to resolve your issue. We suggest following the [general troubleshooting](#) first.

Contents

- [General troubleshooting to try first](#)
 - [Check your AWS CLI command formatting](#)
 - [Check the AWS Region your AWS CLI command is using](#)
 - [Confirm that you're running a recent version of the AWS CLI](#)
 - [Use the --debug option](#)
 - [Enable and review the AWS CLI command history logs](#)
 - [Confirm that your AWS CLI is configured](#)
- [Command not found errors](#)
- [The "aws --version" command returns a different version than you installed](#)
- [The "aws --version" command returns a version after uninstalling the AWS CLI](#)
- [The AWS CLI processed a command with an incomplete parameter name](#)
- [Access denied errors](#)
- [Invalid credentials and key errors](#)
- [Signature does not match errors](#)
- [SSL certificate errors](#)
- [Invalid JSON errors](#)
- [Additional resources](#)

General troubleshooting to try first

If you receive an error or encounter an issue with the AWS CLI, we suggest the following general tips to help you troubleshoot.

[Back to top](#)

Check your AWS CLI command formatting

If you receive an error that indicates that a command doesn't exist, or that it doesn't recognize a parameter (`Parameter validation failed`) that the documentation says is available, then your command might be formatted incorrectly. We suggest that you check the following:

- Check your command for spelling and formatting errors.
- Confirm all [quotes and escaping appropriate for your terminal](#) is correct in your command.
- Generate an [AWS CLI skeleton](#) to confirm your command structure.
- For JSON, see the additional [troubleshooting for JSON values](#). If you're having issues with your terminal processing JSON formatting, we suggest skipping past the terminal's quoting rules by using [Blobs to pass JSON data directly to the AWS CLI](#).

For more information on how a specific command should be structured, see the [AWS CLI version 2 reference guide](#).

[Back to top](#)

Check the AWS Region your AWS CLI command is using

Note

You must specify an AWS Region when using the AWS CLI, either explicitly or by setting a default Region. For a list of all of the AWS Regions that you can specify, see [AWS Regions and Endpoints](#) in the *Amazon Web Services General Reference*. The AWS Region designators used by the AWS CLI are the same names that you see in AWS Management Console URLs and service endpoints.

Errors or unexpected results might occur if an AWS service isn't available for your specified AWS Region or your resources are located in a different AWS Region. In order of precedence, the AWS Region is set in the following ways:

- The `--region` command line option.
- The SDK compatible [AWS_REGION](#) environment variable.
- The [AWS_DEFAULT_REGION](#) environment variable.
- The [region](#) profile setting.

Confirm you're using the correct AWS Region for your resources.

[Back to top](#)

Confirm that you're running a recent version of the AWS CLI

If you receive an error that indicates that a command doesn't exist, or that it doesn't recognize a parameter that the [AWS CLI version 2 reference guide](#) says is available, first confirm that your command is correctly formatted. If the formatting is correct, then we recommend that you upgrade to the most recent version of the AWS CLI. Updated versions of the AWS CLI are released almost every business day. New AWS services, features, and parameters are introduced in those new versions of the AWS CLI. The only way to get access to those new services, features, or parameters is to upgrade to a version that was released after that element was first introduced.

How you update your version of the AWS CLI depends on how you originally installed it as described in [the section called "Install/Update"](#).

If you used one of the bundled installers, you might need to remove the existing installation before you download and install the latest version for your operating system.

[Back to top](#)

Use the --debug option

When the AWS CLI reports an error that you don't immediately understand, or produces results that you don't expect, you can get more detail about the error by running the command again with the --debug option. With this option, the AWS CLI outputs details about every step it takes to process your command. The details in the output can help you to determine when the error occurs and provides clues about where it started.

You can send the output to a text file for later review, or to send to AWS Support when asked for it.

When you include the --debug option, some of the details include:

- Looking for credentials
- Parsing the provided parameters
- Constructing the request sent to AWS servers
- The contents of the request sent to AWS
- The contents of the raw response
- The formatted output

Here's an example of a command run with and without the `--debug` option.

```
$ aws iam list-groups --profile MyTestProfile
{
  "Groups": [
    {
      "Path": "/",
      "GroupName": "MyTestGroup",
      "GroupId": "AGPA0123456789EXAMPLE",
      "Arn": "arn:aws:iam::123456789012:group/MyTestGroup",
      "CreateDate": "2019-08-12T19:34:04Z"
    }
  ]
}
```

```
$ aws iam list-groups --profile MyTestProfile --debug
2019-08-12 12:36:18,305 - MainThread - awscli.clidriver - DEBUG - CLI version: aws-
cli/1.16.215 Python/3.7.3 Linux/4.14.133-113.105.amzn2.x86_64 botocore/1.12.205
2019-08-12 12:36:18,305 - MainThread - awscli.clidriver - DEBUG - Arguments entered to
CLI: ['iam', 'list-groups', '--debug']
2019-08-12 12:36:18,305 - MainThread - botocore.hooks - DEBUG - Event session-
initialized: calling handler <function add_scalar_parsers at 0x7fdf173161e0>
2019-08-12 12:36:18,305 - MainThread - botocore.hooks - DEBUG - Event session-
initialized: calling handler <function register_uri_param_handler at 0x7fdf17dec400>
2019-08-12 12:36:18,305 - MainThread - botocore.hooks - DEBUG - Event session-
initialized: calling handler <function inject_assume_role_provider_cache at
0x7fdf17da9378>
2019-08-12 12:36:18,307 - MainThread - botocore.credentials - DEBUG - Skipping
environment variable credential check because profile name was explicitly set.
2019-08-12 12:36:18,307 - MainThread - botocore.hooks - DEBUG - Event session-
initialized: calling handler <function attach_history_handler at 0x7fdf173ed9d8>
2019-08-12 12:36:18,308 - MainThread - botocore.loaders - DEBUG - Loading JSON
file: /home/ec2-user/venv/lib/python3.7/site-packages/botocore/data/iam/2010-05-08/
service-2.json
2019-08-12 12:36:18,317 - MainThread - botocore.hooks - DEBUG - Event building-command-
table.iam: calling handler <function add_waiters at 0x7fdf1731a840>
2019-08-12 12:36:18,320 - MainThread - botocore.loaders - DEBUG - Loading JSON
file: /home/ec2-user/venv/lib/python3.7/site-packages/botocore/data/iam/2010-05-08/
waiters-2.json
2019-08-12 12:36:18,321 - MainThread - awscli.clidriver - DEBUG - OrderedDict([('path-
prefix', <awscli.arguments.CLIArument object at 0x7fdf171ac780>), ('marker',
<awscli.arguments.CLIArument object at 0x7fdf171b09e8>), ('max-items',
<awscli.arguments.CLIArument object at 0x7fdf171b09b0>)])
```



```
2019-08-12 12:36:18,322 - MainThread - botocore.hooks - DEBUG - Event building-argument-table.iam.list-groups: calling handler <function add_streaming_output_arg at 0x7fdf17316510>
2019-08-12 12:36:18,322 - MainThread - botocore.hooks - DEBUG - Event building-argument-table.iam.list-groups: calling handler <function add_cli_input_json at 0x7fdf17da9d90>
2019-08-12 12:36:18,322 - MainThread - botocore.hooks - DEBUG - Event building-argument-table.iam.list-groups: calling handler <function unify_paging_params at 0x7fdf17328048>
2019-08-12 12:36:18,326 - MainThread - botocore.loaders - DEBUG - Loading JSON file: /home/ec2-user/venv/lib/python3.7/site-packages/botocore/data/iam/2010-05-08/paginators-1.json
2019-08-12 12:36:18,326 - MainThread - awscli.customizations.paginate - DEBUG - Modifying paging parameters for operation: ListGroups
2019-08-12 12:36:18,326 - MainThread - botocore.hooks - DEBUG - Event building-argument-table.iam.list-groups: calling handler <function add_generate_skeleton at 0x7fdf1737eae8>
2019-08-12 12:36:18,326 - MainThread - botocore.hooks - DEBUG - Event before-building-argument-table-parser.iam.list-groups: calling handler <bound method OverrideRequiredArgsArgument.override_required_args of <awscli.customizations.cliinputjson.CliInputJSONArgument object at 0x7fdf171b0a58>>
2019-08-12 12:36:18,327 - MainThread - botocore.hooks - DEBUG - Event before-building-argument-table-parser.iam.list-groups: calling handler <bound method GenerateCliSkeletonArgument.override_required_args of <awscli.customizations.generatecliskeleton.GenerateCliSkeletonArgument object at 0x7fdf171c5978>>
2019-08-12 12:36:18,327 - MainThread - botocore.hooks - DEBUG - Event operation-args-parsed.iam.list-groups: calling handler functools.partial(<function check_should_enable_pagination at 0x7fdf17328158>, ['marker', 'max-items'], {'max-items': <awscli.arguments.CLIArgument object at 0x7fdf171b09b0>}, OrderedDict([('path-prefix', <awscli.arguments.CLIArgument object at 0x7fdf171ac780>), ('marker', <awscli.arguments.CLIArgument object at 0x7fdf171b09e8>), ('max-items', <awscli.customizations.paginate.PageArgument object at 0x7fdf171c58d0>), ('cli-input-json', <awscli.customizations.cliinputjson.CliInputJSONArgument object at 0x7fdf171b0a58>), ('starting-token', <awscli.customizations.paginate.PageArgument object at 0x7fdf171b0a20>), ('page-size', <awscli.customizations.paginate.PageArgument object at 0x7fdf171c5828>), ('generate-cli-skeleton', <awscli.customizations.generatecliskeleton.GenerateCliSkeletonArgument object at 0x7fdf171c5978>)]))
2019-08-12 12:36:18,328 - MainThread - botocore.hooks - DEBUG - Event load-cli-arg.iam.list-groups.path-prefix: calling handler <awscli.paramfile.URIArgumentHandler object at 0x7fdf1725c978>
```

```
2019-08-12 12:36:18,328 - MainThread - botocore.hooks - DEBUG - Event load-cli-arg.iam.list-groups.marker: calling handler <awscli.paramfile.URIArgumentHandler object at 0x7fdf1725c978>
2019-08-12 12:36:18,328 - MainThread - botocore.hooks - DEBUG - Event load-cli-arg.iam.list-groups.max-items: calling handler <awscli.paramfile.URIArgumentHandler object at 0x7fdf1725c978>
2019-08-12 12:36:18,328 - MainThread - botocore.hooks - DEBUG - Event load-cli-arg.iam.list-groups.cli-input-json: calling handler <awscli.paramfile.URIArgumentHandler object at 0x7fdf1725c978>
2019-08-12 12:36:18,328 - MainThread - botocore.hooks - DEBUG - Event load-cli-arg.iam.list-groups.starting-token: calling handler <awscli.paramfile.URIArgumentHandler object at 0x7fdf1725c978>
2019-08-12 12:36:18,328 - MainThread - botocore.hooks - DEBUG - Event load-cli-arg.iam.list-groups.page-size: calling handler <awscli.paramfile.URIArgumentHandler object at 0x7fdf1725c978>
2019-08-12 12:36:18,328 - MainThread - botocore.hooks - DEBUG - Event load-cli-arg.iam.list-groups.generate-cli-skeleton: calling handler <awscli.paramfile.URIArgumentHandler object at 0x7fdf1725c978>
2019-08-12 12:36:18,329 - MainThread - botocore.hooks - DEBUG - Event calling-command.iam.list-groups: calling handler <bound method CliInputJSONArgument.add_to_call_parameters of <awscli.customizations.cliinputjson.CliInputJSONArgument object at 0x7fdf171b0a58>>
2019-08-12 12:36:18,329 - MainThread - botocore.hooks - DEBUG - Event calling-command.iam.list-groups: calling handler <bound method GenerateCliSkeletonArgument.generate_json_skeleton of <awscli.customizations.generatecliskeleton.GenerateCliSkeletonArgument object at 0x7fdf171c5978>>
2019-08-12 12:36:18,329 - MainThread - botocore.credentials - DEBUG - Looking for credentials via: assume-role
2019-08-12 12:36:18,329 - MainThread - botocore.credentials - DEBUG - Looking for credentials via: assume-role-with-web-identity
2019-08-12 12:36:18,329 - MainThread - botocore.credentials - DEBUG - Looking for credentials via: shared-credentials-file
2019-08-12 12:36:18,329 - MainThread - botocore.credentials - INFO - Found credentials in shared credentials file: ~/.aws/credentials
2019-08-12 12:36:18,330 - MainThread - botocore.loaders - DEBUG - Loading JSON file: /home/ec2-user/venv/lib/python3.7/site-packages/botocore/data/endpoints.json
2019-08-12 12:36:18,334 - MainThread - botocore.hooks - DEBUG - Event choose-service-name: calling handler <function handle_service_name_alias at 0x7fdf1898eb70>
2019-08-12 12:36:18,337 - MainThread - botocore.hooks - DEBUG - Event creating-client-class.iam: calling handler <function add_generate_presigned_url at 0x7fdf18a028c8>
2019-08-12 12:36:18,337 - MainThread - botocore.regions - DEBUG - Using partition endpoint for iam, us-west-2: aws-global
```

```
2019-08-12 12:36:18,337 - MainThread - botocore.args - DEBUG - The s3 config key is not
a dictionary type, ignoring its value of: None
2019-08-12 12:36:18,340 - MainThread - botocore.endpoint - DEBUG - Setting iam timeout
as (60, 60)
2019-08-12 12:36:18,341 - MainThread - botocore.loaders - DEBUG - Loading JSON file: /
home/ec2-user/venv/lib/python3.7/site-packages/botocore/data/_retry.json
2019-08-12 12:36:18,341 - MainThread - botocore.client - DEBUG - Registering retry
handlers for service: iam
2019-08-12 12:36:18,342 - MainThread - botocore.hooks - DEBUG - Event before-
parameter-build.iam.ListGroups: calling handler <function generate_idempotent_uuid at
0x7fdf189b10d0>
2019-08-12 12:36:18,342 - MainThread - botocore.hooks - DEBUG - Event before-
call.iam.ListGroups: calling handler <function inject_api_version_header_if_needed at
0x7fdf189b2a60>
2019-08-12 12:36:18,343 - MainThread - botocore.endpoint - DEBUG - Making
request for OperationModel(name=ListGroups) with params: {'url_path': '/',
'query_string': '', 'method': 'POST', 'headers': {'Content-Type': 'application/x-
www-form-urlencoded; charset=utf-8', 'User-Agent': 'aws-cli/1.16.215 Python/3.7.3
Linux/4.14.133-113.105.amzn2.x86_64 botocore/1.12.205'}, 'body': {'Action':
'ListGroups', 'Version': '2010-05-08'}, 'url': 'https://iam.amazonaws.com/',
'context': {'client_region': 'aws-global', 'client_config': <botoconfig.Config
object at 0x7fdf16e9a4a8>, 'has_streaming_input': False, 'auth_type': None}}
2019-08-12 12:36:18,343 - MainThread - botocore.hooks - DEBUG - Event request-
created.iam.ListGroups: calling handler <bound method RequestSigner.handler of
<botoconfig.signers.RequestSigner object at 0x7fdf16e9a470>>
2019-08-12 12:36:18,343 - MainThread - botocore.hooks - DEBUG - Event choose-
signer.iam.ListGroups: calling handler <function set_operation_specific_signer at
0x7fdf18996f28>
2019-08-12 12:36:18,343 - MainThread - botocore.auth - DEBUG - Calculating signature
using v4 auth.
2019-08-12 12:36:18,343 - MainThread - botocore.auth - DEBUG - CanonicalRequest:
POST
/

content-type:application/x-www-form-urlencoded; charset=utf-8
host:iam.amazonaws.com
x-amz-date:20190812T193618Z

content-type;host;x-amz-date
5f776d91EXAMPLE9b8cb5eb5d6d4a787a33ae41c8cd6eEXAMPLEca69080e1e1f
2019-08-12 12:36:18,344 - MainThread - botocore.auth - DEBUG - StringToSign:
AWS4-HMAC-SHA256
20190812T193618Z
20190812/us-east-1/iam/aws4_request
```

```

ab7e367eEXAMPLE2769f178ea509978cf8bfa054874b3EXAMPLE8d043fab6cc9
2019-08-12 12:36:18,344 - MainThread - botocore.auth - DEBUG - Signature:
d85a0EXAMPLEeb40164f2f539cdc76d4f294fe822EXAMPLE18ad1ddf58a1a3ce7
2019-08-12 12:36:18,344 - MainThread - botocore.endpoint - DEBUG - Sending
http request: <AWSPreparedRequest stream_output=False, method=POST,
url=https://iam.amazonaws.com/, headers={'Content-Type': b'application/
x-www-form-urlencoded; charset=utf-8', 'User-Agent': b'aws-cli/1.16.215
Python/3.7.3 Linux/4.14.133-113.105.amzn2.x86_64 botocore/1.12.205',
'X-Amz-Date': b'20190812T193618Z', 'Authorization': b'AWS4-HMAC-SHA256
Credential=AKIA01234567890EXAMPLE-east-1/iam/aws4_request, SignedHeaders=content-
type;host;x-amz-date, Signature=d85a07692aceb401EXAMPLEa1b18ad1ddf58a1a3ce7EXAMPLE',
'Content-Length': '36'}>
2019-08-12 12:36:18,344 - MainThread - urllib3.util.retry - DEBUG - Converted retries
value: False -> Retry(total=False, connect=None, read=None, redirect=0, status=None)
2019-08-12 12:36:18,344 - MainThread - urllib3.connectionpool - DEBUG - Starting new
HTTPS connection (1): iam.amazonaws.com:443
2019-08-12 12:36:18,664 - MainThread - urllib3.connectionpool - DEBUG - https://
iam.amazonaws.com:443 "POST / HTTP/1.1" 200 570
2019-08-12 12:36:18,664 - MainThread - botocore.parsers - DEBUG - Response headers:
{'x-amzn-RequestId': '74c11606-bd38-11e9-9c82-559da0adb349', 'Content-Type': 'text/
xml', 'Content-Length': '570', 'Date': 'Mon, 12 Aug 2019 19:36:18 GMT'}
2019-08-12 12:36:18,664 - MainThread - botocore.parsers - DEBUG - Response body:
b'<ListGroupResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">\n
<ListGroupResult>\n  <IsTruncated>>false</IsTruncated>\n  <Groups>\n
  <member>\n    <Path>/</Path>\n    <GroupName>MyTestGroup</GroupName>
\n    <Arn>arn:aws:iam::123456789012:group/MyTestGroup</Arn>\n
  <GroupId>AGPA1234567890EXAMPLE</GroupId>\n    <CreateDate>2019-08-12T19:34:04Z</
CreateDate>\n  </member>\n  </Groups>\n </ListGroupResult>\n
  <ResponseMetadata>\n    <RequestId>74c11606-bd38-11e9-9c82-559da0adb349</RequestId>\n
  </ResponseMetadata>\n</ListGroupResponse>\n'
```

```

2019-08-12 12:36:18,665 - MainThread - botocore.hooks - DEBUG - Event needs-
retry.iam.ListGroups: calling handler <botocore.retryhandler.RetryHandler object at
0x7fdf16e9a780>
2019-08-12 12:36:18,665 - MainThread - botocore.retryhandler - DEBUG - No retry needed.
2019-08-12 12:36:18,665 - MainThread - botocore.hooks - DEBUG - Event after-
call.iam.ListGroups: calling handler <function json_decode_policies at 0x7fdf189b1d90>
{
  "Groups": [
    {
      "Path": "/",
      "GroupName": "MyTestGroup",
      "GroupId": "AGPA123456789012EXAMPLE",
      "Arn": "arn:aws:iam::123456789012:group/MyTestGroup",
      "CreateDate": "2019-08-12T19:34:04Z"
    }
  ]
}

```

```
    }  
  ]  
}
```

[Back to top](#)

Enable and review the AWS CLI command history logs

You can enable the AWS CLI command history logs using the [cli_history](#) file setting. After enabling this setting, the AWS CLI records the history of `aws` commands.

You can list your history using the `aws history list` command, and use the resulting `command_ids` in the `aws history show` command for details. For more information see [aws history](#) in the *AWS CLI reference guide*.

When you include the `--debug` option, some of the details include:

- API calls made to botocore
- Status codes
- HTTP responses
- Headers
- Return codes

You can use this information to confirm parameter data and API calls are behaving in the way you expect, and can then deduce at what step in the process your command is failing.

[Back to top](#)

Confirm that your AWS CLI is configured

Various errors can occur if your `config` and `credentials` files or your IAM user or role is not configured correctly. For more information on resolving errors with `config` and `credentials` files or your IAM user or roles, see [the section called "Access denied errors"](#) and [the section called "Invalid credentials and key errors"](#).

[Back to top](#)

Command not found errors

This error means that the operating system can't find the AWS CLI command. The installation might be incomplete or requires updating.

Possible cause: You're trying to use an AWS CLI feature newer than your installed version, or have incorrect formatting

Example error text:

```
$ aws s3 copy
usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:

    aws help
    aws <command> help
    aws <command> <subcommand> help
aws: error: argument subcommand: Invalid choice, valid choices are:

ls                | website
cp                | mv
....
```

Various errors can occur if your command is formatted incorrectly or you are using an earlier version from before the feature was released. For more information on resolving errors around these two issues, see [the section called “Check your AWS CLI command formatting”](#) and [the section called “Confirm that you're running a recent version of the AWS CLI”](#).

[Back to top](#)

Possible cause: The terminal needs to be restarted after installation

Example error text:

```
$ aws --version
command not found: aws
```

If the aws command cannot be found after first installing or updating the AWS CLI, you might need to restart your terminal for it to recognize any PATH updates.

[Back to top](#)

Possible cause: The AWS CLI did not fully install

Example error text:

```
$ aws --version
command not found: aws
```

If the `aws` command cannot be found after first installing or updating the AWS CLI, it might not have been fully installed. Try reinstalling by following the steps for your platform in [the section called "Install/Update"](#).

[Back to top](#)

Possible cause: The AWS CLI does not have permissions (Linux)

If the `aws` command cannot be found after first installing or updating the AWS CLI on Linux, it might not have execute permissions for the folder it installed in. Run the following command with the PATH to your AWS CLI installation, to provide [chmod](#) permissions to the AWS CLI:

```
$ sudo chmod -R 755 /usr/local/aws-cli/
```

[Back to top](#)

Possible cause: The operating system PATH was not updated during installation

Example error text:

```
$ aws --version
command not found: aws
```

You might need to add the `aws` executable to your operating system's PATH environment variable. To add the AWS CLI to your PATH, use the following instructions for your operating system.

Linux and macOS

1. Find your shell's profile script in your user directory. If you're not sure which shell you have, run `echo $SHELL`.

```
$ ls -a ~
```

```
. .. .bash_logout .bash_profile .bashrc Desktop Documents Downloads
```

- **Bash** – `.bash_profile`, `.profile`, or `.bash_login`
 - **Zsh** – `.zshrc`
 - **Tcsh** – `.tcshrc`, `.cshrc`, or `.login`
2. Add an export command to your profile script. The following command adds your local bin to the current PATH variable.

```
export PATH=/usr/local/bin:$PATH
```

3. Reload the updated profile into your current session.

```
$ source ~/.bash_profile
```

Windows

1. In a Windows Command Prompt, use the `where` command with the `/R` *path* parameter to find the aws file location. The results return all folders containing aws.

```
C:\> where /R c:\ aws  
c:\Program Files\Amazon\AWSCLIV2\aws.exe  
...
```

By default, the AWS CLI version 2 is located in:

```
c:\Program Files\Amazon\AWSCLIV2\aws.exe
```

2. Press the Windows key and enter **environment variables**.
3. From the list of suggestions, choose **Edit environment variables for your account**.
4. Choose **PATH**, and then choose **Edit**.
5. Add the path you found in the first step into the **Variable value** field, for example, **C:
*\Program Files\Amazon\AWSCLIV2\aws.exe***.
6. Choose **OK** twice to apply the new settings.
7. Close any running command prompts and reopen the command prompt window.

[Back to top](#)

The "aws --version" command returns a different version than you installed

Your terminal might be returning a different PATH for the AWS CLI than you expect.

Possible cause: The terminal needs to be restarted after installing

If the aws command shows the wrong version, you might need to restart your terminal for it to recognize any PATH updates. All open terminals needs to be closed, not just your active terminal.

[Back to top](#)

Possible cause: The system needs to be restarted after installing

If the aws command shows the wrong version and restarting the terminal didn't work, you might need to restart your system for it to recognize your PATH updates.

[Back to top](#)

Possible cause: You have multiple versions of the AWS CLI

If you updated the AWS CLI and used a different install method than your pre-existing installation, it might cause multiple versions to be installed. For example, if on Linux or macOS you used pip for your current install, but tried to update using the .pkg install file, this could cause some conflicts especially with your PATH pointing to the old version.

To resolve this, [uninstall all versions of the AWS CLI](#) and perform a clean install.

After uninstalling all versions, follow instructions appropriate for your operating system to install your desired version of the [AWS CLI version 1](#) or [AWS CLI version 2](#).

Note

If this is happening after you installed the AWS CLI version 2 with a pre-existing install of AWS CLI version 1, follow the migration instructions in [the section called "Migration instructions"](#).

[Back to top](#)

The "aws --version" command returns a version after uninstalling the AWS CLI

This often occurs when there is still an AWS CLI installed somewhere on your system.

Possible cause: The terminal needs to be restarted after uninstalling

If the `aws --version` command still works, you might need to restart your terminal for it to recognize any terminal updates.

[Back to top](#)

Possible cause: You have multiple versions of the AWS CLI on your system, or didn't use the same uninstall method that you used to originally install the AWS CLI

The AWS CLI might not uninstall correctly if you uninstalled the AWS CLI using a different method than you used to install it, or if you installed multiple versions. For example, if you used `pip` for your current install, you must use `pip` to uninstall it. To resolve this, uninstall AWS CLI using the same method that you used to install it.

1. Follow the instructions appropriate for your operating system and your original installation method to uninstall the [AWS CLI version 1](#) and [AWS CLI version 2](#).
2. Close all terminals you have open.
3. Open your preferred terminal, enter in the following command and confirm that no version is returned.

```
$ aws --version
command not found: aws
```

If you still have a version listed in the output, the AWS CLI was most likely installed using a different method or there are multiple versions. If you don't know which method you installed the AWS CLI, follow the instructions for each uninstall method for the [AWS CLI version 1](#) and [AWS CLI version 2](#) appropriate to your operating system until no version output is received.

Note

If you used a package manager to install the AWS CLI (`pip`, `apt`, `brew`, etc.), you must use the same package manager to uninstall it. Be sure to follow the

instructions provided by the package manager on how to uninstall all versions of a package.

[Back to top](#)

The AWS CLI processed a command with an incomplete parameter name

Possible cause: You used a recognized abbreviation of the AWS CLI parameter

Since the AWS CLI is built using Python, the AWS CLI uses the Python `argparse` library, including the [allow_abbrev](#) argument. Abbreviations of parameters are recognized by the AWS CLI and processed.

The following [create-change-set](#) command example changes the CloudFormation stack name. The parameter `--change-set-n` is recognized as an abbreviation of `--change-set-name`, and the AWS CLI processes the command.

```
$ aws cloudformation create-change-set --stack-name my-stack --change-set-n my-change-set
```

When your abbreviation could be multiple commands, the parameter will not be recognized as an abbreviation.

The following [create-change-set](#) command example changes the CloudFormation stack name. The parameter `--change-set-` is **not** recognized as an abbreviation, as there are multiple parameters it could be an abbreviation of, such as `--change-set-name` and `--change-set-type`. Therefore the AWS CLI does **not** process the command.

```
$ aws cloudformation create-change-set --stack-name my-stack --change-set- my-change-set
```

Warning

Do not purposefully use parameter abbreviations. They are unreliable and are not backwards compatible. If any new parameters are added to a command that confuse your abbreviations, it will break your commands.

Additionally, if the parameter is a single-value argument, it can cause unexpected behavior with your commands. If multiple instances of a single-value argument is passed, only the last instance will run. In the following example, the parameter `--filters` is a single-valued argument. The parameters `--filters` and `--filter` are specified. The `--filter` parameter is an abbreviation of `--filters`. This cause two instances of `--filters` being applied and only the last `--filter` argument applies.

```
$ aws ec2 describe-vpc-peering-connections \  
  --filters Name=tag:TagName,Values=VpcPeeringConnection \  
  --filter Name=status-code,Values=active
```

Confirm you are using valid parameters before running a command to prevent unexpected behavior.

[Back to top](#)

Access denied errors

Possible cause: The AWS CLI program file doesn't have "run" permission

On Linux or macOS, make sure that the `aws` program has run permissions for the calling user. Typically, the permissions are set to 755.

To add run permission for your user, run the following command, substituting `~/.local/bin/aws` with the path to the program on your computer.

```
$ chmod +x ~/.local/bin/aws
```

[Back to top](#)

Possible cause: Your IAM identity doesn't have permission to perform the operation

Example error text:

```
$ aws s3 ls  
An error occurred (AccessDenied) when calling the ListBuckets operation: Access  
denied.
```

When you run a AWS CLI command, AWS operations are performed on your behalf, using credentials that associate you with an IAM account or role. The policies attached must grant you permission to call the API actions that correspond to the commands that you run with the AWS CLI.

Most commands call a single action with a name that matches the command name. However, custom commands like `aws s3 sync` call multiple APIs. You can see which APIs a command calls by using the `--debug` option.

If you are sure that the user or role has the proper permissions assigned by policy, make sure that your AWS CLI command is using the credentials you expect. See the [next section about credentials](#) to verify that the credentials the AWS CLI is using are the ones that you expect.

For information about assigning IAM permissions, see [Overview of Access Management: Permissions and Policies](#) in the *IAM User Guide*.

[Back to top](#)

Invalid credentials and key errors

Example error text:

```
$ aws s3 ls
```

```
An error occurred (InvalidAccessKeyId) when calling the ListBuckets operation: The AWS
Access Key Id
you provided does not exist in our records.
```

```
$ aws s3 ls
```

```
An error occurred (InvalidClientTokenId) when calling the ListBuckets operation: The
security token
included in the request is invalid.
```

Possible cause: The AWS CLI is reading incorrect credentials or from an unexpected location

The AWS CLI might be reading credentials from a different location than you expect, or your key pair information is incorrect. You can run `aws configure list` to confirm which credentials are used.

The following example shows how to check the credentials used for the default profile.

```
$ aws configure list
  Name                Value                Type    Location
  ----                -
  profile              <not set>           None    None
  access_key           *****XYVA         shared-credentials-file
  secret_key           *****ZAGY         shared-credentials-file
  region               us-west-2           config-file  ~/.aws/config
```

The following example shows how to check the credentials of a named profile.

```
$ aws configure list --profile saanvi
  Name                Value                Type    Location
  ----                -
  profile              saanvi              manual  --profile
  access_key           *****             shared-credentials-file
  secret_key           *****             shared-credentials-file
  region               us-west-2           config-file  ~/.aws/config
```

To confirm your key pair details, check your config and credentials files. For more information on config and credentials files, see [the section called “Configuration and credential file settings”](#). For more information on credentials and authentication, including credentials precedence, see [Authentication and access credentials](#).

[Back to top](#)

Possible cause: Your computer's clock is out of sync

If you are using valid credentials, your clock might be out of sync. On Linux or macOS, run `date` to check the time.

```
$ date
```

If your system clock is not correct within a few minutes, use `ntpd` to sync it.

```
$ sudo service ntpd stop
$ sudo ntpdate time.nist.gov
$ sudo service ntpd start
$ ntpstat
```

On Windows, use the date and time options in the Control Panel to configure your system clock.

[Back to top](#)

Signature does not match errors

Example error text:

```
$ aws s3 ls
```

```
An error occurred (SignatureDoesNotMatch) when calling the ListBuckets operation: The request signature we calculated does not match the signature you provided. Check your key and signing method.
```

When the AWS CLI runs a command, it sends an encrypted request to the AWS servers to perform the appropriate AWS service operations. Your credentials (the access key and secret key) are involved in the encryption and enable AWS to authenticate the person making the request. There are several things that can interfere with the correct operation of this process, as follows.

Possible cause: Your clock is out of sync with the AWS servers

To help protect against [replay attacks](#), the current time can be used during the encryption/decryption process. If the time of the client and server disagree by more than the allowed amount, the process can fail and the request is rejected. This can also happen when you run a command in a virtual machine whose clock is out of sync with the host machine's clock. One possible cause is when the virtual machine hibernates and takes some time after waking up to sync the clock with the host machine.

On Linux or macOS, run `date` to check the time.

```
$ date
```

If your system clock is not correct within a few minutes, use `ntpd` to sync it.

```
$ sudo service ntpd stop
$ sudo ntpdate time.nist.gov
$ sudo service ntpd start
$ ntpstat
```

On Windows, use the date and time options in the Control Panel to configure your system clock.

[Back to top](#)

Possible cause: Your operating system is mishandling AWS keys that contain certain special characters

If your AWS keys include certain special characters, such as -, +, /, or %, some operating system variants process the string improperly and cause the key string to be interpreted incorrectly.

If you process your keys using other tools or scripts, such as tools that build the credentials file on a new instance as part of its creation, those tools and scripts might have their own handling of special characters that causes them to be transformed into something that AWS no longer recognizes.

We suggest regenerating the secret key to get one that does not include the special character causing issues.

[Back to top](#)

SSL certificate errors

Possible cause: The AWS CLI doesn't trust your proxy's certificate

Example error text:

```
$ aws s3 ls
[SSL: CERTIFICATE_ VERIFY_FAILED] certificate verify failed
```

When you use a AWS CLI command, you receive an [SSL: CERTIFICATE_ VERIFY_FAILED] certificate verify failed error message. This is caused by the AWS CLI not trusting your proxy's certificate due to factors such as your proxy's certificate being self-signed, with your company set as the Certification Authority (CA). This prevents the AWS CLI from finding your companies CA root certificate in the local CA registry.

To fix this, instruct the AWS CLI where to find your companies .pem file using the [ca_bundle](#) configuration file setting, [--ca-bundle](#) command line option, or the [AWS_CA_BUNDLE](#) environment variable.

[Back to top](#)

Possible cause: Your configuration isn't pointing to the correct CA root certificate location

Example error text:


```
$ aws s3 ls
SSL validation failed for regionname [Errno 2] No such file or directory
```

This is caused by your Certification Authority (CA) bundle file location being configured incorrectly in the AWS CLI. To fix this, confirm where your companies .pem file is located and update the AWS CLI configuration by using the [ca_bundle](#) configuration file setting, [--ca-bundle](#) command line option, or the [AWS_CA_BUNDLE](#) environment variable.

[Back to top](#)

Possible cause: Your configuration isn't using the correct AWS Region

Example error text:

```
$ aws s3 ls
[SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed
```

Errors or unexpected results might occur if an AWS service isn't available for your specified AWS Region or your resources are located in a different AWS Region. For troubleshooting steps, see [the section called "Check the AWS Region your AWS CLI command is using"](#).

[Back to top](#)

Possible cause: Your TLS version needs to be updated

Example error text:

```
$ aws s3 ls
[SSL: UNSAFE_LEGACY_RENEGOTIATION_DISABLED] unsafe legacy renegotiation disabled
```

The AWS service is using a version of TLS that is incompatible with your device's TLS version. To resolve this issue, update to a supported TLS version. For more information, see [the section called "Enforcing a minimum TLS version"](#).

[Back to top](#)

Invalid JSON errors

Example error text:

```
$ aws dynamodb update-table \  
  --provisioned-throughput '{"ReadCapacityUnits':15,WriteCapacityUnits':10}' \  
  --table-name MyDDBTable  
Error parsing parameter '--provisioned-throughput': Invalid JSON: Expecting property  
name enclosed in  
double quotes: line 1 column 25 (char 24)  
JSON received: {"ReadCapacityUnits":15,WriteCapacityUnits":10}
```

When you use an AWS CLI command, you receive a "Invalid JSON" error message. This is usually an error seen when you enter a command with an expected JSON format and the AWS CLI cannot read your JSON correctly.

Possible cause: You did not enter valid JSON for the AWS CLI to use

Confirm you have valid JSON entered for your command. We suggest using a JSON validator for JSON you're having issues formatting.

For more advanced JSON usage in the command line, consider using a command line JSON processor, like `jq`, to create JSON strings. For more information on `jq`, see the [jq repository](#) on *GitHub*.

[Back to top](#)

Possible cause: Your terminal's quoting rules are preventing valid JSON being sent to the AWS CLI

Before the AWS CLI receives anything from a command, your terminal processes the command using its own quoting and escaping rules. Due to a terminal's formatting rules, some of your JSON content may be stripped before the command is passed to the AWS CLI. When formulating commands, be sure to use your [terminal's quoting rules](#).

To troubleshoot, use the `echo` command to see how the shell is handling your parameters:

```
$ echo {"ReadCapacityUnits":15,"WriteCapacityUnits":10}  
ReadCapacityUnits:15 WriteCapacityUnits:10
```

```
$ echo '{"ReadCapacityUnits":15,"WriteCapacityUnits":10}'  
{"ReadCapacityUnits":15,"WriteCapacityUnits":10}
```

Modify your command until your until valid JSON is returned.

For more in-depth troubleshooting, use the `--debug` parameter to view the debug logs as they'll display exactly what got passed to the AWS CLI:

```
$ aws dynamodb update-table \  
  --provisioned-throughput '{"ReadCapacityUnits":15,WriteCapacityUnits":10}' \  
  --table-name MyDDBTable \  
  --debug  
2022-07-19 22:25:07,741 - MainThread - awscli.clidriver - DEBUG - CLI version: aws-  
cli/1.18.147  
Python/2.7.18 Linux/5.4.196-119.356.amzn2int.x86_64 boto3/1.18.6  
2022-07-19 22:25:07,741 - MainThread - awscli.clidriver - DEBUG - Arguments entered  
to CLI:  
['dynamodb', 'update-table', '--provisioned-throughput',  
'{"ReadCapacityUnits":15,WriteCapacityUnits":10}',  
 '--table-name', 'MyDDBTable', '--debug']
```

Use your terminal's quoting rules to fix any issues your JSON input has when being sent to the AWS CLI. For more information on quoting rules, see [the section called “Quotes with Strings”](#).

Note

If you're having issues with getting valid JSON to the AWS CLI, we recommend to bypass a terminal's quoting rules for JSON data input by using Blobs to pass your JSON data directly to the AWS CLI. For more information on Blobs, see [Blob](#).

[Back to top](#)

Additional resources

For additional help with your AWS CLI issues, visit the [AWS CLI community](#) on *GitHub* or the [AWS re:Post community](#).

[Back to top](#)

Migrate from AWS CLI version 1 to version 2

This section contains instructions for updating the AWS CLI version 1 to AWS CLI version 2. The AWS CLI version 2 builds on AWS CLI version 1 and includes features and enhancements based on community feedback.

Before you migrate to version 2, [learn about the differences between the versions](#). The AWS CLI version 2 includes new features and changes that might require you to update your scripts or commands for backwards compatibility.

AWS CLI versions 1 and 2 use the same `aws` command name. If you have both versions installed, your computer uses the first one found in your search path.

If you previously installed AWS CLI version 1, follow our [migration instructions to start using version 2](#).

If you have not previously installed AWS CLI version 1, follow the instructions in [Get started](#).

Topics

- [New features and changes in AWS CLI version 2](#)
- [AWS CLI version 2 migration instructions](#)

New features and changes in AWS CLI version 2

This topic describes new features and changes in behavior between AWS CLI version 1 and AWS CLI version 2. These changes might require you to update your scripts or commands to get the same behavior in version 2 as you did in version 1.

Topics

- [AWS CLI version 2 new features](#)
- [Breaking changes between AWS CLI version 1 and AWS CLI version 2](#)

AWS CLI version 2 new features

The AWS CLI version 2 is the most recent major version of the AWS CLI and supports all of the latest features. Some features introduced in version 2 are not backported to version 1 and you must upgrade to access those features. These features include the following:

Python interpreter not needed

The AWS CLI version 2 doesn't need a separate install of Python. It includes an embedded version.

[Wizards](#)

You can use a wizard with the AWS CLI version 2. The wizard guides you through constructing certain commands.

[IAM Identity Center authentication](#)

If your organization uses AWS IAM Identity Center (IAM Identity Center), your users can sign in to Active Directory, a built-in IAM Identity Center directory, or [another IdP connected to IAM Identity Center](#). Then, they are mapped to an AWS Identity and Access Management (IAM) role that allows you to run AWS CLI commands.

[Auto-prompt](#)

When enabled, the AWS CLI version 2 can prompt you for commands, parameters, and resources when you run an `aws` command.

[Run the AWS CLI from the official Amazon ECR Public or Docker images](#)

The official Docker image for the AWS CLI provides isolation, portability, and security that AWS directly supports and maintains. This way, you can use the AWS CLI version 2 in a container-based environment without having to manage the installation yourself.

[Client-side pager](#)

The AWS CLI version 2 provides the use of a client-side pager program for output. By default, this feature is turned on and returns all output through your operating system's default pager program.

[aws configure import](#)

Import `.csv` credentials generated from the AWS Management Console. A `.csv` file is imported with the profile name matching the IAM user name.

[aws configure list-profiles](#)

Lists the names of all profiles you have configured.

[the section called “YAML stream output format”](#)

The `yaml` and `yaml-stream` format takes advantage of the [YAML](#) format while providing more responsive viewing of large datasets by streaming the data to you. You can start viewing and using YAML data before the entire query downloads.

[New high-level ddb commands for DynamoDB](#)

The AWS CLI version 2 has the high-level Amazon DynamoDB commands [ddb put](#) and [ddb select](#). These commands provide a simplified interface for putting items in DynamoDB tables and searching in a DynamoDB table or index.

[aws logs tail](#)

The AWS CLI version 2 has a custom `aws logs tail` command that tails the logs for an Amazon CloudWatch Logs group. By default, the command returns logs from all associated CloudWatch Logs streams during the past ten minutes.

[Added metadata support for high-level s3 commands](#)

The AWS CLI version 2 adds the `--copy-props` parameter to the high-level `s3` commands. With this parameter, you can configure additional metadata and tags for Amazon Simple Storage Service (Amazon S3).

[AWS_REGION](#)

The AWS CLI version 2 has an AWS SDK-compatible environment variable called `AWS_REGION`. This variable specifies the AWS Region to send requests to. It overrides the `AWS_DEFAULT_REGION` environment variable, which is only applicable in the AWS CLI.

Breaking changes between AWS CLI version 1 and AWS CLI version 2

This section describes all of the changes in behavior between AWS CLI version 1 and AWS CLI version 2. These changes might require you to update your scripts or commands to get the same behavior in version 2 as you did in version 1.

Topics

- [Environment variable added to set text file encoding](#)
- [Binary parameters are passed as base64-encoded strings by default](#)
- [Improved Amazon S3 handling of file properties and tags for multipart copies](#)
- [No automatic retrieval of http:// or https:// URLs for parameters](#)

- [Pager used for all output by default](#)
- [Timestamp output values are standardized to ISO 8601 format](#)
- [Improved handling of CloudFormation deployments that result in no changes](#)
- [Changed default behavior for Regional Amazon S3 endpoint for us-east-1 Region](#)
- [Changed default behavior for Regional AWS STS endpoints](#)
- [ecr get-login removed and replaced with ecr get-login-password](#)
- [AWS CLI version 2 support for plugins is changing](#)
- [Hidden alias support removed](#)
- [The api_versions configuration file setting is not supported](#)
- [AWS CLI version 2 uses only Signature v4 to authenticate Amazon S3 requests](#)
- [AWS CLI version 2 is more consistent with paging parameters](#)
- [AWS CLI version 2 provides more consistent return codes across all commands](#)

Environment variable added to set text file encoding

By default, text files for [the section called “Blob”](#) use the same encoding as the installed locale. Because the AWS CLI version 2 uses an embedded version of Python, the PYTHONUTF8 and PYTHONIOENCODING environment variables are not supported. To set encoding for text files to be different from the locale, use the AWS_CLI_FILE_ENCODING environment variable. The following example sets the AWS CLI to open text files using UTF-8 on Windows.

```
AWS_CLI_FILE_ENCODING=UTF-8
```

For more information, see [Environment variables to configure the AWS CLI](#) .

Binary parameters are passed as base64-encoded strings by default

In the AWS CLI, some commands required [base64](#)-encoded strings, while others required UTF-8-encoded byte strings. In the AWS CLI version 1, passing data between two encoded string types often required some intermediate processing. The AWS CLI version 2 makes handling binary parameters more consistent, which helps pass values from one command to another more reliably.

By default, the AWS CLI version 2 passes all binary input and binary output parameters as the base64-encoded string b1obs (binary large object). For more information, see [the section called “Blob”](#).

To revert to the AWS CLI version 1 behavior, use the [cli_binary_format](#) file configuration or the [--cli-binary-format](#) parameter.

Improved Amazon S3 handling of file properties and tags for multipart copies

When you use the AWS CLI version 1 commands in the `aws s3` namespace to copy a file from one S3 bucket location to another, and that operation uses [multipart copy](#), no file properties from the source object are copied to the destination object.

By default, the corresponding commands in the AWS CLI version 2 transfer all tags and some of the properties from the source to the destination copy. Compared to the AWS CLI version 1, this can result in more AWS API calls being made to the Amazon S3 endpoint. To change the default behavior for `s3` commands in AWS CLI version 2, use the `--copy-props` parameter.

For more information, see [the section called "File properties and tags in multipart copies"](#).

No automatic retrieval of `http://` or `https://` URLs for parameters

The AWS CLI version 2 does not perform a GET operation when a parameter value begins with `http://` or `https://`, and does not use the returned content as the parameter value. As a result, the associated command line option `cli_follow_urlparam` is removed from the AWS CLI version 2.

If you need to retrieve a URL and pass the URL contents into a parameter value, we recommend that you use `curl` or a similar tool to download the contents of the URL to a local file. Then, use the `file://` syntax to read the contents of that file and use it as the parameter value.

For example, the following command no longer tries to retrieve the contents of the page found at `http://www.example.com` and pass those contents as the parameter. Instead, it passes the literal text string `https://example.com` as the parameter.

```
$ aws ssm put-parameter \  
  --value http://www.example.com \  
  --name prod.microservice1.db.secret \  
  --type String 2
```

If you need to retrieve and use the contents of a web URL as a parameter, you can do the following in version 2.

```
$ curl https://my.example.com/mypolicyfile.json -o mypolicyfile.json
```



```
$ aws iam put-role-policy \  
  --policy-document file:///./mypolicyfile.json \  
  --role-name MyRole \  
  --policy-name MyReadOnlyPolicy
```

In the preceding example, the `-o` parameter tells `curl` to save the file in the current folder with the same name as the source file. The second command retrieves the content of that downloaded file and passes the content as the value of `--policy-document`.

Pager used for all output by default

By default, the AWS CLI version 2 returns all output through your operating system's default pager program. This program is the [less](#) program on Linux or macOS, and the [more](#) program on Windows. This can help you navigate a large amount of output from a service by displaying that output one page at a time.

You can configure the AWS CLI version 2 to use a different paging program or none at all. For more information, see [the section called "Client-side pager"](#).

Timestamp output values are standardized to ISO 8601 format

By default, the AWS CLI version 2 returns all timestamp response values in the [ISO 8601 format](#). In AWS CLI version 1, commands returned timestamp values in whatever format was returned by the HTTP API response, which could vary from service to service.

To see timestamps in the format returned by the HTTP API response, use the `wire` value in your config file. For more information, see [cli_timestamp_format](#).

Improved handling of CloudFormation deployments that result in no changes

By default in the AWS CLI version 1, if you deploy a AWS CloudFormation template that results in no changes, the AWS CLI returns a failed error code. This causes problems if you don't consider that to be an error and you want your script to continue. You can work around this in the AWS CLI version 1 by adding the flag `--no-fail-on-empty-changeset`, which returns `0`.

Since this is a common use case, the AWS CLI version 2 defaults to returning a successful exit code of `0` when there is no change caused by a deployment and the operation returns an empty changeset.

To revert to the original behavior, add the flag `--fail-on-empty-changeset`.

Changed default behavior for Regional Amazon S3 endpoint for us-east-1 Region

When you configure the AWS CLI version 1 to use the us-east-1 Region, the AWS CLI uses the global `s3.amazonaws.com` endpoint that is physically hosted in the us-east-1 Region. The AWS CLI version 2 uses the true Regional endpoint `s3.us-east-1.amazonaws.com` when that Region is specified. To force the AWS CLI version 2 to use the global endpoint, you can set the Region for a command to `aws-global`.

Changed default behavior for Regional AWS STS endpoints

By default, the AWS CLI version 2 sends all AWS Security Token Service (AWS STS) API requests to the Regional endpoint for the currently configured AWS Region.

By default, the AWS CLI version 1 sends AWS STS requests to the global AWS STS endpoint. You can control this default behavior in version 1 by using the [sts_regional_endpoints](#) setting.

`aws ecr get-login` removed and replaced with `aws ecr get-login-password`

The AWS CLI version 2 replaces the command `aws ecr get-login` with the `aws ecr get-login-password` command that improves automated integration with container authentication.

The `aws ecr get-login-password` command reduces the risk of exposing your credentials in the process list, shell history, or other log files. It also improves compatibility with the `docker login` command for better automation.

The `aws ecr get-login-password` command is available in the AWS CLI version 1.17.10 and later, and the AWS CLI version 2. The earlier `aws ecr get-login` command is still available in the AWS CLI version 1 for backward compatibility.

With the `aws ecr get-login-password` command, you can replace the following code that retrieves a password.

```
$ (aws ecr get-login --no-include-email)
```

To reduce the risk of exposing the password to the shell history or logs, use the following example command instead. In this example, the password is piped directly to the `docker login` command, where it is assigned to the password parameter by the `--password-stdin` option.

```
$ aws ecr get-login-password | docker login --username AWS --password-stdin MY-REGISTRY-URL
```

For more information, see [aws ecr get-login-password](#) in the *AWS CLI version 2 Reference Guide*.

AWS CLI version 2 support for plugins is changing

Plugin support in the AWS CLI version 2 is completely provisional and is intended to help users migrate from AWS CLI version 1 until a stable, updated plugin interface is released. There are no guarantees that a particular plugin or even the AWS CLI plugin interface will be supported in future versions of the AWS CLI version 2. If you rely on plugins, be sure to lock into a particular version of the AWS CLI and test the functionality of your plugin when you do upgrade.

To enable plugin support, create a [plugins] section in your `~/.aws/config`.

```
[plugins]
cli_legacy_plugin_path = <path-to-plugins>/python3.7/site-packages
<plugin-name> = <plugin-module>
```

In the [plugins] section, define the `cli_legacy_plugin_path` variable and set its value to the Python site packages path where your plugin module is. Then, you can configure a plugin by providing a name for the plugin (`plugin-name`) and the file name of the Python module (`plugin-module`) that contains the source code for your plugin. The AWS CLI loads each plugin by importing its `plugin-module` and calling its `awscli_initialize` function.

Hidden alias support removed

AWS CLI version 2 no longer supports the following hidden aliases that were supported in version 1.

In the following table, the first column displays the service, command, and parameter that work in all versions, including the AWS CLI version 2. The second column displays the alias that no longer works in the AWS CLI version 2.

Working service, command, and parameter	Obsolete alias
cognito-identity create-identity-pool open-id-connect-provider-arns	open-id-connect-provider-arns

Working service, command, and parameter	Obsolete alias
storagegateway describe-tapes tape-arns	tape-ar-ns
storagegateway.describe-tape-archives.tape-arns	tape-ar-ns
storagegateway.describe-vtl-devices.vtl-device-arns	vtl-device-ar-ns
storagegateway.describe-cached-iscsi-volumes.volume-arns	volume-ar-ns
storagegateway.describe-stored-iscsi-volumes.volume-arns	volume-ar-ns
route53domains.view-billing.start-time	start
deploy.create-deployment-group.ec2-tag-set	ec-2-tag-set
deploy.list-application-revisions.s3-bucket	s-3-bucket
deploy.list-application-revisions.s3-key-prefix	s-3-key-prefix
deploy.update-deployment-group.ec2-tag-set	ec-2-tag-set
iam.enable-mfa-device.authentication-code1	authentication-code-1
iam.enable-mfa-device.authentication-code2	authentication-code-2
iam.resync-mfa-device.authentication-code1	authentication-code-1
iam.resync-mfa-device.authentication-code2	authentication-code-2
importexport.get-shipping-label.street1	street-1
importexport.get-shipping-label.street2	street-2
importexport.get-shipping-label.street3	street-3
lambda.publish-version.code-sha256	code-sha-256
lightsail.import-key-pair.public-key-base64	public-key-base-64
opsworks.register-volume.ec2-volume-id	ec-2-volume-id

The `api_versions` configuration file setting is not supported

The AWS CLI version 2 doesn't support calling earlier versions of AWS service APIs by using the `api_versions` configuration file setting. All AWS CLI commands now call the latest version of the service APIs that are currently supported by the endpoint.

AWS CLI version 2 uses only Signature v4 to authenticate Amazon S3 requests

The AWS CLI version 2 doesn't support earlier signature algorithms to cryptographically authenticate service requests sent to Amazon S3 endpoints. This signing happens automatically with every Amazon S3 request and only the [Signature Version 4 Signing Process](#) is supported. You can't configure the signature version. All Amazon S3 bucket presigned URLs now use only SigV4 and have a maximum expiration duration of one week.

AWS CLI version 2 is more consistent with paging parameters

In the AWS CLI version 1, if you specify pagination parameters on the command line, then automatic pagination is turned off as expected. However, when you specify pagination parameters by using a file with the `--cli-input-json` parameter, automatic pagination was not turned off, which could result in unexpected output. The AWS CLI version 2 turns off automatic pagination regardless of how you provide the parameters.

AWS CLI version 2 provides more consistent return codes across all commands

The AWS CLI version 2 is more consistent across all commands and properly returns an appropriate exit code compared to the AWS CLI version 1. We also added exit codes 252, 253, and 254. For more information on exit codes, see [the section called "Return Codes"](#).

If you have a dependency on how the AWS CLI version 1 uses return code values, we recommend checking the exit codes to make sure that you're getting the values you expect.

AWS CLI version 2 migration instructions

This topic provides instructions for migrating from AWS CLI version 1 to AWS CLI version 2.

AWS CLI versions 1 and 2 use the same `aws` command name. If you have both versions installed, your computer uses the first one found in your search path. If you previously installed AWS CLI version 1, we recommend that you do one of the following to use AWS CLI version 2:

- **Recommended** – [Uninstall AWS CLI version 1 and use only AWS CLI version 2.](#)

- [To have both version installed](#), use your operating system's ability to create a symbolic link (symlink) or alias with a different name for one of the two aws commands.

For information on breaking changes between version 1 and version 2, see [the section called "New features and changes"](#).

Replacing version 1 with version 2

Perform the following steps to replace AWS CLI version 1 with AWS CLI version 2.

To replace AWS CLI version 1 with AWS CLI version 2

1. Prepare any existing scripts you have for the migration by confirming any breaking changes between version 1 and version 2 in [the section called "New features and changes"](#).
2. Uninstall the AWS CLI version 1 by following the uninstall instructions for your operating system in [Installing, updating, and uninstalling the AWS CLI version 1](#).
3. Confirm that the AWS CLI is completely uninstalled by using the following command.

```
$ aws --version
```

Complete one of the following based on the output:

- **No version returned:** You've successfully uninstalled the AWS CLI version 1 and can proceed to the next step.
 - **A version is returned:** You still have an install of the AWS CLI version 1. For troubleshooting steps, see [the section called "The "aws --version" command returns a version after uninstalling the AWS CLI"](#). Perform troubleshooting steps until no version output is received.
4. Install the AWS CLI version 2 by following the appropriate install instructions for your operating system in [Install or update to the latest version of the AWS CLI](#).

Side-by-side install

To have both versions installed, use your operating system's ability to create a symbolic link (symlink) or alias with a different name for one of the two aws commands.

1. Install the AWS CLI version 2 by following the appropriate install instructions for your operating system in [Install or update to the latest version of the AWS CLI](#).

2. Use your operating system's ability to create a symlink or alias with a different name for one of the two aws commands, such as using *aws2* for AWS CLI version 2. The following are symlink examples for AWS CLI version 2. Replace the *PATH* with your install location.

Linux and macOS

You can use a [symbolic link](#) or [alias](#) on Linux and macOS.

```
$ alias aws2='PATH'
```

Windows command prompt

[DOSKEY](#) on Windows.

```
C:\> doskey aws2=PATH
```

Uninstall the AWS CLI version 2

This topic describes how to uninstall the AWS Command Line Interface version 2 (AWS CLI version 2).

AWS CLI version 2 uninstallation instructions:

Linux

To uninstall the AWS CLI version 2, run the following commands.

1. Locate the symlink and install paths.

- Use the `which` command to find the symlink. This shows the path you used with the `--bin-dir` parameter.

```
$ which aws
/usr/local/bin/aws
```

- Use the `ls` command to find the directory that the symlink points to. This gives you the path you used with the `--install-dir` parameter.

```
$ ls -l /usr/local/bin/aws
lrwxrwxrwx 1 ec2-user ec2-user 49 Oct 22 09:49 /usr/local/bin/aws -> /usr/local/
aws-cli/v2/current/bin/aws
```

2. Delete the two symlinks in the `--bin-dir` directory. If your user has write permission to these directories, you don't need to use `sudo`.

```
$ sudo rm /usr/local/bin/aws
$ sudo rm /usr/local/bin/aws_completer
```

3. Delete the `--install-dir` directory. If your user has write permission to this directory, you don't need to use `sudo`.

```
$ sudo rm -rf /usr/local/aws-cli
```

4. **(Optional)** Remove the shared AWS SDK and AWS CLI settings information in the `.aws` folder.

⚠ Warning

These configuration and credentials settings are shared across all AWS SDKs and the AWS CLI. If you remove this folder, they cannot be accessed by any AWS SDKs that are still on your system.

The default location of the `.aws` folder differs between platforms, by default the folder is located in `~/.aws/`. If your user has write permission to this directory, you don't need to use `sudo`.

```
$ sudo rm -rf ~/.aws/
```

macOS

To uninstall the AWS CLI version 2, run the following commands, substituting the paths you used to install. The example commands use the default installation paths.

1. Find the folder that contains the symlinks to the main program and the completer.

```
$ which aws
/usr/local/bin/aws
```

2. Using that information, run the following command to find the installation folder that the symlinks point to.

```
$ ls -l /usr/local/bin/aws
lrwxrwxrwx 1 ec2-user ec2-user 49 Oct 22 09:49 /usr/local/bin/aws -> /usr/local/
aws-cli/aws
```

3. Delete the two symlinks in the first folder. If your user already has write permission to these folders, you don't need to use `sudo`.

```
$ sudo rm /usr/local/bin/aws
$ sudo rm /usr/local/bin/aws_completer
```

4. Delete the main installation folder. Use `sudo` to gain write access to the `/usr/local` folder.

```
$ sudo rm -rf /usr/local/aws-cli
```

5. **(Optional)** Remove the shared AWS SDK and AWS CLI settings information in the `.aws` folder.

Warning

These configuration and credentials settings are shared across all AWS SDKs and the AWS CLI. If you remove this folder, they cannot be accessed by any AWS SDKs that are still on your system.

The default location of the `.aws` folder differs between platforms, by default the folder is located in `~/.aws/`. If your user has write permission to this directory, you don't need to use `sudo`.

```
$ sudo rm -rf ~/.aws/
```

Windows

1. Open **Programs and Features** by doing one of the following:
 - Open the **Control Panel**, and then choose **Programs and Features**.
 - Open a command prompt, and then enter the following command.

```
C:\> appwiz.cpl
```

2. Select the entry named **AWS Command Line Interface**, and then choose **Uninstall** to launch the uninstaller.
3. Confirm that you want to uninstall the AWS CLI.
4. **(Optional)** Remove the shared AWS SDK and AWS CLI settings information in the `.aws` folder.

⚠ Warning

These configuration and credentials settings are shared across all AWS SDKs and the AWS CLI. If you remove this folder, they cannot be accessed by any AWS SDKs that are still on your system.

The default location of the `.aws` folder differs between platforms, by default the folder is located in `%UserProfile%\.aws`.

```
$ rmdir %UserProfile%\.aws
```

Troubleshooting AWS CLI install and uninstall errors

If you come across issues after installing or uninstalling the AWS CLI, see [Troubleshoot errors](#) for troubleshooting steps. For the most relevant troubleshooting steps, see [the section called "Command not found errors"](#), [the section called "The "aws --version" command returns a different version than you installed"](#), and [the section called "The "aws --version" command returns a version after uninstalling the AWS CLI"](#).

AWS CLI user guide document history

The following table describes important additions to the *AWS Command Line Interface User Guide*, beginning in January 2019. For notification about updates to this documentation, you can subscribe to the RSS feed.

Change	Description	Date
Updated credential and authentication information.	Updated credential and authentication method instructions and examples. This includes updating relevant Getting started pages and configuration pages. To accommodate this increase in documentation, relevant credential topics were moved to the new Authentication and access credentials section.	March 31, 2023
Token provider configuration with automatic authentication refresh for AWS IAM Identity Center added	The new process to configure the AWS CLI to authenticate users with AWS IAM Identity Center (IAM Identity Center) using the SSO token provider configuration, which can automatically retrieve refreshed authentication tokens.	December 7, 2022
Official Amazon ECR Public image for the AWS CLI version 2 released	The official supported Amazon ECR Public image for the AWS CLI version 2 is released for Linux, macOS, and Windows.	November 18, 2022

Updated the guide for migrating from AWS CLI V1 to V2	Expanded the breaking changes guide to include migration instructions to going from AWS CLI version 1 to the AWS CLI version 2. Includes updates to the Troubleshooting page to help with installation issues.	May 13, 2022
New process to build a AWS CLI installer from source.	New process to install or update from source to the latest release of the AWS CLI on supported operating systems.	February 17, 2022
Content for the AWS CLI V1 and V2 are now separated into their respective guides	For clarity and ease, the AWS CLI version 1 and AWS CLI version 2 content is now separated into their own guides. For AWS CLI version 1, see the AWS CLI version 1 User Guide .	November 2, 2021
Added AWS CLI alias information	Added AWS CLI alias information. Aliases are shortcuts you can create in the AWS Command Line Interface (AWS CLI) to shorten commands or scripts that you frequently use.	March 11, 2021
Updated filter output information	Updated information for filters and moved to their own page.	February 1, 2021
Added information for Wizards	Added AWS CLI version 2 wizard information.	November 20, 2020

Updated auto-prompt	Updated the AWS CLI version 2 auto-prompt information with current features.	November 10, 2020
Added Amazon S3 scripting example	Added an Amazon S3 lifecycle scripting example.	October 15, 2020
Added Amazon EC2 scripting example	Added an Amazon EC2 instance type scripting example.	October 15, 2020
Added retries information	Added a retries page for features and behavior of retries in the AWS CLI.	September 17, 2020
Server-side and client-side pagination page	Updated pagination information and centralized on a single page.	August 17, 2020
Updated s3 commands page	Updated the high-level s3 commands page with new examples and resources.	July 30, 2020
Updated installation information	The install, update, and uninstall information for Linux, macOS, and Windows are updated.	May 19, 2020
Added information for text file encoding on the AWS CLI version 2	By default, AWS CLI version 2 uses the same text file encoding as the local. You can now use environment variables to set text file encoding.	May 14, 2020

Official Docker image for the AWS CLI version 2 released	The official support Docker image for the AWS CLI version 2 is released for all Linux, macOS, and Windows.	March 31, 2020
Added information regarding client-side pagers for AWS CLI version 2	By default, AWS CLI version 2 uses the pager program <code>less</code> for all client-side output.	February 19, 2020
AWS Command Line Interface (AWS CLI) Version 2 is officially released	The AWS CLI version 2 is generally available and is the recommended version for customers to install.	February 10, 2020
macOS installer for AWS CLI version 2 is now an Apple Package installer .pkg file.	The macOS installer for AWS CLI version 2 has been updated from a .zip file with a shell script to full macOS Installer package. This simplifies installation and makes it compatible with the newest macOS releases.	February 3, 2020
Added content for AWS CLI version 2's improved default handling of S3 and STS Regional endpoints	By default, AWS CLI version 2 now directs requests for the Amazon S3 and AWS STS services to the currently configured Regional endpoint instead of the global endpoint.	January 13, 2020
Developer preview release for AWS CLI version 2	Announcing preview release of AWS CLI version 2. Added instructions about installing version 2. Add Migration topic to discuss differences between versions 1 and 2.	November 7, 2019

Added support for AWS IAM Identity Center to AWS CLI named profiles	AWS CLI version 2 adds support for creating a named profile that can directly login to IAM Identity Center and get AWS temporary credentials for use in subsequent AWS CLI commands.	November 7, 2019
New MFA section	Added a new section describing how to access the CLI using multi-factor authentication and roles.	May 3, 2019
Update to "Using the CLI" section	Major improvements and additions to the usage instructions and procedures.	March 7, 2019
Update to "Installing the CLI" section	Major improvements and additions to the AWS CLI installation instructions and procedures.	March 7, 2019
Update to "Configuring the CLI" section	Major improvements and additions to the AWS CLI configuration instructions and procedures.	March 7, 2019

AWS Glossary

For the latest AWS terminology, see the [AWS glossary](#) in the *AWS Glossary Reference*.